Phonology (2022), **39**:2 213–249
doi:10.1017/S0952675723000076

PHONOLOGY

ARTICLE

The atomic properties of stress

Nate Koser

Department of Linguistics, Rutgers, the State University of New Jersey, New Brunswick, NJ, USA.
Email: nate.koser@quantiphi.com

Received: 13 February 2022; **Revised:** 15 August 2022; **Accepted:** 2 September 2022;
First published online: 27 September 2023

Keywords: computational phonology, stress, computational complexity, locality, iterative stress

Abstract

This article posits a theory of iterative stress that separates each facet of the stress map into its constituent parts, or 'atoms'. Through the well-defined notion of complexity provided by Formal Language Theory, it is shown that this division of the stress map results in a more restrictive characterisation of iterative stress than a single-function analysis does. While the single-function approach masks the complexity of the atomic properties present in the pattern, the compositional analysis makes it explicitly clear. It also demonstrates the degree to which, despite what appear to be significant surface differences in the patterns, the calculation of the stress function is largely the same, even between quantity-sensitive and quantity-insensitive patterns. These stress compositions are limited to one output-local function to iterate stress, and a small number of what I call *edge-oriented* functions to provide 'cleanup' when the iteration function alone fails to capture the pattern.

Contents

1. Introduction

This article articulates a restrictive theory of unidirectional iterative stress from a computational perspective. I argue that the best characterisation of iterative stress is one where the 'atoms' of the stress generalisation are composed in sequence. The atoms are defined as functions that implement basic aspects of stress such as basic iteration or non-finality. The article thus joins a vast body of previous work treating surface stress patterns as the sum of individual stress generalisations (Chomsky & Halle 1968; Booij 1983; Halle & Vergnaud 1987; Idsardi 1992; Bailey 1995; Hayes 1995; Gordon 2002; Hyde 2002; Kager 2005; Buckley 2009; Kager 2012; Rogers *et al*. 2013; Heinz 2014), but addresses the question of what a possible stress pattern can be through the lens of computation, analysing these individual stress functions with the tools of formal language theory (FLT). The iteration of stress is shown to be an output strictly local (OSL) function (Chandlee *et al*. 2015; Chandlee & Heinz 2018), while phonological requirements such as non-finality, clash and lapse are encoded with what I refer to as 'edge-oriented' (EO) functions. I define the class of EO functions by their limitation to apply only at or near a word edge, thus representing a novel characterisation of a typologically real property of stress patterns using computational methods. The proposal then is that the observed output of an iterative stress map is the composition of the local functions implementing the relevant stress primitives.

This proposal argues that a compositional analysis provides a better hypothesis for the typology of iterative stress than an analysis using a single function. This is in part because, for some patterns, this single function is properly *subsequential* – that is, it belongs to a class of functions that can describe fundamentally non-local patterns (Mohri 1997; Schützenberger 1977). However, I demonstrate that these iterative stress patterns *are* local, containing just a small set of atoms that each express a local generalisation. Though compositions of local functions can in general describe non-local patterns, the restrictions invoked here ensure that a well-defined notion of locality is preserved. Full subsequential power also overgenerates by predicting pathological parity-counting stress patterns that are non-phonological. Thus, this article connects the FLT complexity classes with substantive elements of phonological theory – though one subsequential function does suffice, it masks the true computational nature of the individual stress atoms such as clash, lapse and the basic iteration of stress. The result is a restrictive theory that makes explicit the computational requirements for iterative stress while embracing its individual substantive elements that have been noted in the literature for decades.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

Whereas previous work from FLT on the composition of stress primitives has focused on stress systems as formal language sets and the intersection of those sets (Rogers *et al*. 2013; Heinz 2014; Rogers & Lambert 2019), here the focus is on the composition of *functions*. The analysis of stress as a mapping from the input through a series of functions to the output resembles classic analyses of stress that view it as a map from an underlying representation to a surface representation. Many of these analyses date back to the earliest days of phonology (Chomsky & Halle 1968; Liberman & Prince 1977; Halle & Vergnaud 1987; Idsardi 1992; Hayes 1995). What the approach taken here contributes is an exact formal characterisation of the individual 'atoms' of stress assignment and their typological consequences based on their computational properties. This contributes a unique perspective to the question of why stress appears in the way that it does on the surface – the computational restrictions on stress mappings determine what is and is not a possible stress generalisation, providing a testable hypothesis for iterative stress. Though the compositional theory of stress presented here is too restrictive for stress in general, it is sufficient to characterise all unidirectional iterative patterns found in the typological studies that were examined (Bailey 1995; Gordon 2002; Heinz 2009).

By examining both *quantity-insensitive* (QI) and *quantity-sensitive* (QS) stress, I show that – despite surface differences – they share fundamental computational properties, including that of iteration of stress by an OSL function. While minor differences appear in the analyses to account for surface patterns, all iterative QI and QS patterns can be captured with this decomposed OSL plus EO analysis. Past work has attempted to account for similarities in QI and QS typology (Prince 1983; Kager 1992), and the analyses below highlight and add additional perspective to the computational similarities as well.

This article also contributes to our knowledge of function composition generally, as the compositional analyses in this article draw from the novel EO class of functions. The OSL function iterates stress through a string. Where this fails to capture the correct input–output map with regard to the specific language, a following EO function provides 'cleanup' for the OSL function. In intuitive terms, EO functions are limited to apply only within a fixed distance from a word edge, and so make a limited number of alterations to the string – the cleanup provided is always local. Limitation to EO prevents the abuse of markup, where the first function leaves information that the second function can exploit. Avoiding this constrains overgeneration and, I argue, is a property of bounded stress systems in general.

The remainder of the article is structured as follows: §2 gives relevant background on stress. §3 defines the EO class and situates it within the FLT complexity hierarchy. §4 presents analyses of iterative stress patterns. §5 discusses the results and outlines areas for future work. §6 concludes.

2. Background

2.1 Stress

Stress is realised as acoustic prominence on one or more syllables in a word. The study of stress is as old as the study of phonology in generative grammar. Because of the wide variety of patterns in the typology of stress, it makes for a lucrative testing ground for

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

formal theories of linguistics. For some previous work that engages large tracts of the stress typology, see Hyman (1977), Booij (1983), Halle & Vergnaud (1987), Dresher & Kaye (1990), Bailey (1995), Hayes (1995); Gordon (2002); and Heinz (2009).

2.1.1 Previous work

This article follows in the tradition of work on stress that examines the question of what a possible stress pattern is. The topic has been approached from many different angles, including rule-based and parametric approaches (Chomsky & Halle 1968; Booij 1983; Halle & Vergnaud 1987; Idsardi 1992; Hayes 1995) as well as constraint-based approaches (Bailey 1995; Gordon 2002; Hyde 2002; Kager 2005; Buckley 2009; Kager 2012). On top of the theoretical devices that have been used to analyse stress, such as feet or metrical grids (Liberman & Prince 1977; Halle & Vergnaud 1978; Selkirk 1980; Prince 1983; Hammond 1984; Halle & Vergnaud 1987; Idsardi 1992; Hayes 1995), previous work often proposes special mechanisms to account for the patterns we observe and their particular details, such as rules of extrametricality (Liberman & Prince 1977; Hayes 1981) or the NONFINALITY constraint of OT grammars (Prince & Smolensky 1993). This productive body of earlier work on stress thus shares the intuition of this article that the surface patterns we observe are the sum of more fine-grained, primitive stress generalisations.

Previous work on stress from the computational perspective offers an additional perspective to the issue of what constitutes a possible stress pattern (Heinz 2007a; Rogers *et al*. 2013; Heinz 2014; Baek 2018; Hao & Andersson 2019; Rogers & Lambert 2019; Koser & Jardine 2020b). However, most of this previous work differs in that it studies stress patterns as phonotactics, rather than a mapping from an input to an output. Analysis of stress as a function rather than as well-formed stringsets, as in this article, provides a computational characterisation of stress patterns that is more in line with how phonological grammar is typically conceived.

2.1.2 Representation

In this article, stress is studied as a string-to-string mapping, where a series of functions apply to an input of bare syllables and return an output of syllables marked with stress according to the properties of the function. For example, an 'initial stress' function would provide the following mapping:

(1) σσσσσ ↦ σ́σσσσ

The analyses presented here make no reference to foot structure or the metrical grid (Liberman & Prince 1977; Halle & Vergnaud 1978; Selkirk 1980; Prince 1983; Hammond 1984; Halle & Vergnaud 1987; Idsardi 1992; Hayes 1995). This choice is not a denial of hierarchical structure for stress – in fact, the mappings described below are congruent with the designation of heads in such serial analyses, where syllables are marked for stress and thus promoted to the next level on the tree or grid. As such, in addition to the conclusions drawn with regard to computation, the results of this article can be taken as a formal analysis of an integral step in classic serial accounts of stress.

It should also be acknowledged that the conception of stress assignment in this article is arguably more akin to a grid structure, as it lacks horizontal constituency groups such as feet or prosodic words. So, while this work attempts to abstract away from specific proposals with regard to representation of stress as much as possible, it is

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

not the case that *no* theoretical commitments are made, and the results presented here are indeed tied to those theoretical commitments.[1] For example, the analyses presented here adopt the syllable as a basic unit that is created before stress applies which, while reasonably non-controversial, is still a theoretical commitment.

For the sake of conciseness, the proceeding analyses do not refer to levels of stress, as in primary versus secondary stress. Primary stresses could be incorporated into the machinery – I demonstrate in §5 that any resulting differences are cosmetic and do not alter the conclusions that are drawn with regard to computation.

2.1.3 Stress typology

This article examines iterative stress patterns where stress iterates in a single direction throughout the word, that is, they are unidirectional. This excludes what I refer to as 'multi-directional' iterative patterns, as in Cahuilla (Seiler 1977),[2] and long-distance patterns, such as the default-to-same (DTS) and default-to-opposite (DTO) patterns. The atomic properties of such patterns are a direction for future research that I discuss briefly in §5, otherwise setting them aside.

Unidirectional iterative patterns anchor a main stress at some point in the word near an edge, while further stresses iterate away from the main stress. An example is Murinbata (Street & Mollinjin 1981):

(2) σ́σ, σ́σσ̀, σ́σσ̀σ, σ́σσ̀σσ̀, σ́σσ̀σσ̀σ, ...

Main stress is located on the initial syllable, and secondary stresses are applied recursively to every odd-numbered syllable thereafter. Murinbata is an example of a QI pattern, meaning that determination of stress placement is not affected by the presence of heavy or light syllables.

I also examine QS patterns, where stress is impacted by syllable weight. A classic (non-iterative) example of a QS pattern is Latin, which stresses the penultimate syllable unless it is light, in which case the antepenultimate syllable is stressed instead:

(3)

light penult:	heavy penult:
Ĺ L L	H Ĥ H
Ĥ L L	L Ĥ L
Ĥ L H	L Ĥ H
Ĺ L H	H Ĥ L

Below it is shown that, despite what appears to be an extreme difference in the surface realisation of stress between QI and QS patterns, the computational properties of the stress function in both cases are largely the same.

As highlighted in Heinz (2009), one goal of research in this vein is to determine what basic properties the stress assignment function has, and what separates a possible stress pattern from an impossible one. For example, there are many logically possible stress patterns that are pathological and do not appear in the stress typology. Consider that, while stress systems such as 'stress every other syllable starting from the first' or 'stress every other syllable starting from the penult' appear perfectly reasonable and

[1] I thank an anonymous reviewer for clarifying this point.

[2] Note that here 'multi-directional' does not refer to patterns like Garawa (Furby 1974) which are often given the label of 'bidirectional', but rather to stress that iterates fully through the word in both directions.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

are in fact attested, there is no known stress system that follows a rule like 'stress every other syllable starting from the middle'. Pathological patterns that require locating the middle of the word have been referred to as the Midpoint Pathology (Eisner 1997; Hyde 2008; Kager 2012), shown here:

(4) σ́σ
σσ́σ
σσ́σσ
σσσ́σσ
σσσ́σσσ
σσσσ́σσσ…

Stress is fixed in the middle of the word. Though phonologists would generally agree that such a pattern is unattested and pathological, the formal notion of complexity available in FLT indicates *why* that is the case: the generalisation 'find the middle' exceeds the proposed complexity threshold for phonological functions – it is not even a *regular* function (Eisner 1997).[3] If phonology is at most regular (Johnson 1972; Kaplan & Kay 1994), and stress maps belong to an even more restrictive class than that, it is natural that no stress patterns based on the middle of the word should exist.

However, a sufficiently restrictive theory of iterative stress will enforce even more stringent requirements than being formally regular, as there are clearly pathological patterns that fall within the regular boundary. One example is the following 'sour grapes'-like pattern (SG; Wilson 2003, 2006) for stress (Koser & Jardine 2020a,b). In such patterns, if a condition such as spread of a feature cannot be satisfied to the fullest possible extent, the condition is abandoned instead of applying partially:

(5) σσσ
σ́σ́σ
σσσσσ
σ́σ́σ́σ́σ
σσσσσσσ
σ́σ́σ́σ́σ́σ…

Here, stress only iterates through words of even parity. Though such a pattern is clearly pathological in that it relies on the parity of the entire word, it will be shown that it is not ruled out by the assumption that phonology itself is regular. However, I demonstrate below that if iterative stress maps are restricted to the composition of OSL and EO functions, then this sour-grapes stress pattern *is* eliminated from the predicted typology.

Thus, an important result of this article is a restrictive theory of stress based on its computational properties that makes testable predictions about what we should observe in the typology of natural language stress.

2.2 Finite-state representations of stress

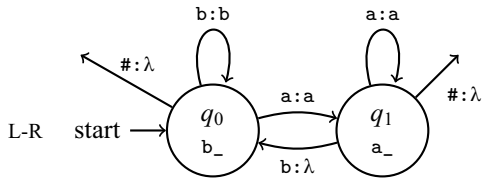
In this article, I study stress as a mapping from an input string of syllables that are unmarked for stress to an output string of syllables that are marked for stress. To study

[3] Stanton (2016) also offers an explanation from learnability.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

the formal properties of these mappings, I represent them using finite-state transducers (FSTs; Kaplan & Kay 1994; Mohri 1997; see Sakarovitch 2006 for an introduction). An FST is a kind of directed graph where a set of states are connected by transitions between those states. FSTs provide a medium to study the abstract computational properties a function has by making it clear what kind of information the function is sensitive to. In this article, the relevant information encoded by a certain state is given as a label appearing on the state. The transitions are labelled with input–output pairs, where the input symbol is given to the left of a colon and the output symbol appears to the right. When moving through a machine, the current state of the transduction encodes information that is relevant to the function it represents. The function represented by (6) deletes an input *b* that appears immediately after an input *a*:

(6)



When the transduction is in q_0, this indicates either the start of the string, or that the symbol most recently read was a *b*: all transitions leading to q_0 have an input *b*. Thus, *b* is the minimal *suffix* leading to q_0, and so state q_0 is labelled b_, indicating that the transducer has just read (if anything) a *b*, and that it is now ready to read the symbol immediately to the right of that *b*. State q_1 is the '*a* state': all transitions leading to q_1 have an input *a*, and so it is labelled a_. When in q_1, if the next input symbol is a *b*, the transition to q_0 is taken and the empty string λ is written to the output. This type of transition with λ can be used to model deletion, or when the function needs to wait for more information before deciding what to output for a given input. The transitions leaving q_0 and q_1 labelled with # indicate reading of the word boundary and the end of the word. As word boundaries have no phonetic content, they contribute λ to the output. Throughout the article, I exclude these where they are irrelevant – that is, when the word boundary does not affect the behaviour of a particular function. The direction in which the transduction applies is noted with 'L-R' or 'R-L' under the transducer. This is for clarity for the reader, and is not part of the formal definition of an FST. A transduction that applies left to right starts by reading the symbol at the left edge of the string, and vice versa. The transduction derives the following outputs for the example inputs:

(7) a. *aaabbb* ↦ *aaabb*
b. *bbbbb* ↦ *bbbbb*
c. *ababab* ↦ *aaa*
d. *bbbaaa* ↦ *bbbaaa*

To demonstrate the application of a transduction to a string, consider the following derivation for input (7a). Diagrams such as the one in (8) are helpful in interpreting transducers, and so feature throughout the article. The input and output lines represent

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

the respective input and output strings. The line of states indicates what transition was taken in the machine. For instance, the sequence $q_0 \rightarrow q_1$ can be interpreted as starting in state q_0 and taking a transition to state q_1, taking the symbol above the arrow as input and outputting the symbol below the arrow. The transducer in (6) reads the string left to right, and so the function starts at the beginning of the word.

(8)

input:		a		a		a		b		b		b
states:	q_0	$\rightarrow$	q_1	$\rightarrow$	q_1	$\rightarrow$	q_1	$\rightarrow$	q_0	$\rightarrow$	q_0	$\rightarrow$ q_0
output:		a		a		a		λ		b		b

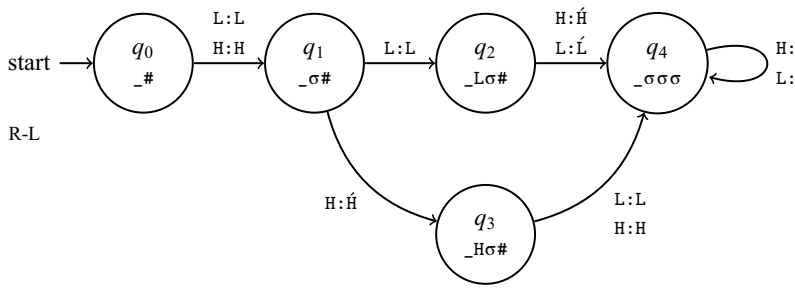
Reading the string left to right, the first input symbol is an *a*, so the transducer outputs a corresponding *a* and moves to q_1, where it will remain until it encounters a *b* or the end of the string. The next two symbols are also *a*, so the transducer remains in q_1. The fourth symbol is a *b*, which causes the transducer to output λ and return to q_0. The remaining input symbols are all *b*, and so the transducer remains in q_0 until it reaches the end of the string. The input *aaabbb* is thus mapped to the output *aaabb*. Note that FSTs such as this one describe not just a single input–output pair, but will map any input to the correct output for a particular pattern.

Studying phonological patterns in this way reveals the abstract computational properties that phonology has. It indicates which patterns are more, less or equally complex in a mathematically defined way. FSTs are thus an invaluable tool in the pursuit of the most restrictive characterisation of linguistic phenomena such as stress. The input–output mappings in (9) exemplify stress assignment according to the Latin pattern in (3):

(9)
a. LLL $\mapsto$ ĹLL
b. HLL $\mapsto$ ĤLL
c. HHH $\mapsto$ HĤH
d. LHL $\mapsto$ LĤL

An FST for Latin is shown in (10):[4]

(10)



[4]It should be noted that the transducers presented here are *minimised* for ease of interpretation, whereas a canonical input strictly local (ISL) or OSL transducer would have a state for each unique suffix. So, for example, in (10), q_1 and q_4 would both be expanded into two different states for L or H inputs. While minimisation means that the transducers are not canonical ISL/OSL transducers as defined in Chandlee (2014), it does not affect the computational properties of the function.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

The transducer reads the input string right to left. The final syllable is never stressed, and so it is output faithfully whether it is light or heavy in the transition from q_0 to q_1. That syllable weight is irrelevant at this point is reflected in the label _σ on q_1, which indicates that the transducer has just read an input heavy *or* light syllable. From there, if a light syllable is seen, it is output as unstressed, as stress avoids light penults. Intuitively, this is the information that is encoded in q_2, as indicated in the label: a light penult has just been read in the input. The next syllable – the antepenult – will then be marked as stressed regardless of its weight. Alternatively, going back to q_1, if a heavy penult is encountered instead, it is stressed in the transition from q_1 to q_3. Once q_4 is reached, all other input symbols are left unchanged, as the end of the three-syllable window has been reached. Interestingly, the state information makes explicit that some QS languages are sensitive to weight in certain positions, rather than the entire word – the penult is the only syllable where weight matters in Latin, for example.

Consider the following derivations for inputs LLL and LHL. This transducer reads the string right to left, and so the function starts at the end of the word:

(11) a.

input:		L		L		L		
states:	q_4	←	q_2	←	q_1	←	q_0	
output:		Ĺ		L		L		

b.

input:		L		H		L		
states:	q_4	←	q_3	←	q_1	←	q_0	
output:		L		Ĥ		L		

It should be noted that finite-state analyses of linguistic patterns are not an assertion that the phonological grammar as it is instantiated in the brain is a series of finite-state machines. Rather, FSTs are merely a useful analytical tool for representing phonological functions. This is because the properties of the transducers tell us about the properties of the functions themselves with regard to computational complexity. For the purposes of this article, the most important property is what kind of information determines movement through the states of the transducer. Is it local to the input or output, or is it non-local information? Does it apply in any part of the word or is it tethered to an edge? These different classes of transducers play distinct roles in the theory of iterative stress presented here, and are discussed in detail here.

3. The proposal and computational complexity

FLT provides a well-defined measure of complexity in the form of complexity classes. The nested hierarchy of complexity classes divides the space of possible functions based on the expressive power of those functions. As applied to natural language, the study of FLT complexity delineates function classes that are relevant to natural language processes, helping to establish testable hypotheses about what a possible linguistic generalisation is. One important claim from this area of research is that phonological processes do not exceed the power of the *regular* class (Johnson 1972; Kaplan & Kay 1994). Intuitively, this is because phonological functions are computed using a finite amount of memory and a finite alphabet. Further research has shown that the vast majority of phonological processes are *subregular,* belonging to some more restrictive subclass of the regular functions (Rogers *et al*. 2013; Heinz 2018). The study of the relation of phonological processes to subregular complexity classes

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

is an ongoing program that delivers precise, mathematically explicit characterisations of phonology.

Research on stress in this vein seeks to identify the upper bound of complexity in stress generalisations, and to characterise the computational properties of stress in a way that is maximally restrictive. Thus, a proposal about the complexity of stress or some subcategory of stress patterns is a hypothesis that all patterns of that type fall within that complexity boundary. In this way, computational studies of stress invite a mutually beneficial relationship with work in other formalisms to provide a more holistic understanding of the nature of stress. For example, if some pathological pattern is not ruled out by a system of rules or constraints, but examination of its complexity indicates that it exceeds the hypothesised boundary for complexity in stress, then this offers an explanation to the pattern's absence, and may give insight into what restrictions can be implemented to remove it from the predicted typology. The reverse relationship can also obtain: hypotheses about complexity do not by themselves rule out all inaccurate typological predictions. For example, a stress pattern that stresses every syllable in the word is maximally simple from a computational perspective, but is unattested and a bad prediction. In such cases, an explanation that makes use of substantive aspects of phonological theory can and should be pursued.

Joining the body of work on stress from a computational perspective, the proposal with regard to iterative stress here uses independently motivated categories from the theory of computation to provide a well-defined notion of the formal nature of mechanisms such as non-finality, and to identify the level of computational power they require. Proposing that iterative stress patterns adhere to certain computational restrictions thus offers a hypothesis not only about what iterative stress maps ought to look like, but *why* they take the form that they do – they are subject to certain computational restrictions that can be expressed in the terms of FLT. It also contributes a precise computational characterisation of the individual 'atoms' present in stress typology and demonstrates that these properties only become apparent when stress is broken down into a series of steps. That a generalisation like non-finality fits within the hypothesised computational limits for iterative stress thus lends support to such substantive mechanisms that have been previously proposed in the stress literature, but does so from the novel perspective of computational complexity that FLT offers.

I start here with the definition of the proposed EO class and then provide descriptions of relevant existing complexity classes, as well as explanations of their relationships with each other.

3.1 EO functions

In this section, I introduce the EO functions, which implement the cleanup functions necessary when iteration of stress alone fails to capture all aspects of an iterative pattern. Intuitively, EO functions are those for which any changes made to the input string only occur in a fixed window at the edge of the word. More formally, in an FST for an EO function, only a finite number of strings can reach a non-identity transition. Therefore, given a string of sufficient length, the transduction always reaches a state where no further changes can be made to the word. The transitions from these states

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

are self-loops that provide identity mappings – the restriction prevents a return to a state with a non-identity transition. In graph-theoretic terms, these properties mean that the largest subgraph of an EO transducer that can reach a non-identity transition is a directed acyclic graph.

(12) **Definition – EO functions:** EO functions are those describable with an FST for which only a finite number of strings arrive in a state with a non-identity transition exiting the state.

Requiring that only a finite number of input sequences can be followed by a change ensures that EO functions can only make alterations to their input in a fixed window at the word edge. Returning to the Latin transducer in (10), for example, L#, H#, LL# and LH# are the only input sequences that can reach a state with a non-identity transition: q_1 or q_2. No transitions return to earlier states, as this would permit an in principle unlimited number of changes to apply to an input, eliminating a necessary property of EO functions that ensures their restrictiveness. In (10), arriving in q_4 indicates that the necessary span to determine the behaviour of the function has been read. The function does not 'care' about any further input symbols – they are always left unchanged.[5] In the EO transducers throughout this article, such states are labelled with the maximal span of input symbols leading to them, making explicit the size of the window that determines the behaviour of the function. For the Latin transducer, q_4 is labelled _σσσ, as this three-syllable window at the right word edge is the maximal span needed to determine the correct output of the Latin stress pattern.

The iterative stress patterns that we observe on the surface are conditioned by a number of atomic stress generalisations that are separate from the iteration of stress itself, such as non-finality, clash and lapse. EO functions encode these atomic properties of iterative stress in a straightforward and restrictive manner, as they are always tethered to an edge in attested iterative patterns. In a non-finality function, for example, the only string leading to a non-identity transition is the right word boundary, # (remembering that the function reads the string from right to left). The next input syllable is output as unstressed, and the rest of the word is left unchanged. The behaviour of the function is determined by a small, local window at the word edge. EO functions provide a direct characterisation of this property of stress atoms. Prioritising the word edge is a property of stress patterns in general: single stress patterns (final stress, penultimate stress, etc.) never place stress more than three syllables away from an edge, nor do iterative patterns begin farther from an edge than that (Hyman 1977; Gordon 2002). EO thus provides a testable hypothesis for iterative stress – a substantive claim couched in computational terms that goes beyond categorisation of surface patterns to explain *why* iterative patterns appear the way that they do.

3.2 OSL functions

Another relevant class of functions is the OSL class (Chandlee 2014; Chandlee *et al.* 2015; Chandlee & Heinz 2018). Intuitively, an OSL function is a function that is calculated based entirely on information in the output string. Processes where application

[5] This is reminiscent of the *definite* class of formal languages (Salomaa 1969).

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

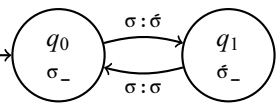
of a rule is iterative, such as spreading of a feature, are in general OSL (Chandlee 2014; Dolatian *et al*. 2021). OSL functions are separated into left and right subclasses (L-OSL and R-OSL). L-OSL functions can express different generalisations than R-OSL functions, as they process the string in opposite directions and so describe iterative patterns only in their respective directions. This output centredness is reflected in OSL transducers, where the states in the machine correspond to sequences of symbols read in the output that are relevant to the changes imposed by the function.

An example of an L-OSL function is left-to-right iteration of stress, as found in Murinbata (Street & Mollinjin 1981):

(13) a. σσ ↦ σ́σ
b. σσσ ↦ σ́σσ́
c. σσσσ ↦ σ́σσ́σ
d. σσσσσ ↦ σ́σσ́σσ́
e. σσσσσσ ↦ σ́σσ́σσ́σ
…

Placement of iterative stress depends on the output string. Intuitively, this is because the next application of stress depends on where it was last placed, and so this information can only be located in the output; the input contains no stresses. The states of an OSL transducer thus correspond to the most recent output:

(14) L-R



The transducer outputs the first syllable it encounters with stress, moving to state q_1. The next syllable is output as unstressed, and the transduction moves back to q_0. In other words, the current state in an OSL transducer encodes information about the *output* string. Reliance on the output is a defining feature of OSL transducers. Formally, a process is called OSL_k if there is some k such that any two strings with the same $k - 1$ suffixes in the output string arrive in the same state. The function represented in (14) is thus OSL_2: any string for which the transduction has just output σ́ arrives in q_1, and any string for which the transduction has just output σ arrives in q_0. The process repeats for the remainder of the word, applying stress iteratively:

(15) a. *Applying (14) to a five-syllable input*

input:	σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1
output:	σ́		σ		σ́		σ		σ́		

b. *Applying (14) to a six-syllable input*

input:	σ		σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1	→	q_0
output:	σ́		σ		σ́		σ		σ́		σ	

Any output string ending in σ will land in q_0, whereas any output string ending in σ́ will be in state q_1. This makes the output-oriented nature of OSL functions explicit:

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

the states do not encode any information about input syllables, but all output strings ending in σ or σ́ will land in their respective state. This is seen in (15), where the five-syllable form ends in q_1, whereas the six-syllable form ends in state q_0.

A consequence of this is that OSL transducers cannot have a 'waiting' transition between two states in which the empty string λ is written to the output, unlike the transducer in (18), presented below in §3.3. This configuration in a transducer means that there are infinite pairs of strings with the same output suffix that will land in different states – q_2 or q_3 in (18) – which is an explicit indication that the function itself is not OSL.

OSL functions can generate some iterative stress patterns, but not all. As will be shown below, for stress assignment where iteration is the only factor, an OSL mapping is sufficient. Placement of further stresses in an iterative chain depends on the previous stress in the output – the input of bare syllables is not enough information. However, if there are other stress phenomena in play, such as clash or non-finality, an OSL function alone is insufficient. This is because such patterns require a small amount of lookahead – information other than the most recent output, which is not available to OSL functions. This article proposes that in such cases, the stress map is the composition of an OSL and EO function, where the first handles the iteration of stress and the second accounts for other factors in the pattern. This provides a unified account of iterative stress patterns that is based on properties of their computation.

Alongside the OSL class is the ISL class (Chandlee 2014; Chandlee & Heinz 2018). The input-centred counterpart of OSL functions, ISL functions are calculated based entirely on information in the input string. The ISL class includes many common phonological processes such as deletion and epenthesis, and so may seem a natural hypothesis for the cleanup functions of iterative stress. However, I demonstrate below that ISL functions do not provide the correct restriction for the cleanup functions in iterative stress maps.

3.3 Subsequential functions

Another relevant class of functions is the subsequential class (Schützenberger 1977; Mohri 1997). Subsequential functions are those that are computable deterministically on their input in a single direction. Subsequential functions are more powerful than EO, ISL or OSL functions, but are still important in the study of phonology. They are subregular, have well-understood automata-theoretic and learnability properties and represent an important division in the space of possible functions that includes many phonological patterns and excludes many non-phonological ones (Oncina *et al*. 1993; Heinz & Lai 2013; Chandlee 2014; Jardine 2016; Luo 2017; Payne 2017). Though there are notable exceptions (Jardine 2016; Hao & Andersson 2019; McCollum *et al*. 2020; Koser & Jardine 2020b), most phonological processes are subsequential.

An example of a subsequential stress pattern comes from the *default to opposite* (DTO) languages. These languages stress a right/leftmost heavy syllable, or the first/last syllable in the opposite direction if no heavies are present. Kwakw'ala exhibits a 'leftmost heavy or right' (LHOR) pattern (Bach 1975; Hayes 1995):

(16) a. LLLL ↦ LLLĹ
b. HHHH ↦ ĤHHH

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

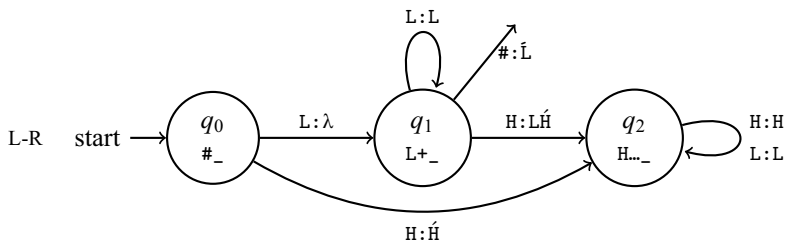
c. LHLLLL ↦ LH́LLLL
d. LHLLHL ↦ LH́LLHL

Reading the string left to right, the first heavy syllable encountered is stressed. If no heavy syllables are encountered, the final light syllable is stressed instead. This type of pattern loses the property of locality that is characteristic of EO, ISL or OSL classes. This is because, to determine if any given H or a final L should be stressed, the function must keep track of the lack or presence of heavy syllables for the entire length of the word up to that point (arrows indicate tracking of information through the string):

(17) LL[Ĥ]LHL LLLLLL[Ĺ]

The transducer in (18) will generate the LHOR stress pattern:

(18)



The transduction proceeds left to right. The transition from q_0 to q_1 takes an input L and 'waits' with λ – no output can be written until more input symbols are seen. If the word is light syllables only, the transduction loops in q_1, outputting Ĺ when the word ends, as seen in the exit transition on input #. If at any point a heavy is seen, it is output with stress and the transduction moves to q_2, where no further changes are made. In intuitive terms, being in q_1 means that only light syllables have been read, whereas being in q_2 means that at least one heavy has been read. Note that this is not local information; LHOR is a long-distance pattern. That the transducer encodes this information about an arbitrary previous number of symbols it has seen makes the function properly subsequential. The following derivations for LLLL and LLHL demonstrate the LHOR mapping:

(19) a. *Applying (18) to LLLL*

input:	L	L	L	L	
states:	q_0 →	q_1 →	q_1 →	q_1 →	q_1
output:	λ	L	L	L	Ĺ

b. *Applying (18) to LLHL*

input:	L	L	H	L	
states:	q_0 →	q_1 →	q_1 →	q_2 →	q_2
output:	λ	L	LH́	L	

When evaluating whether any position in the string, light or heavy, should surface with stress, the function depends on information about the – in principle unbounded – sequence of symbols preceding the current one. The loop in q_1 allows for an arbitrarily

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

long sequence of L inputs before stress is ultimately applied. State q_1 corresponds to only having read input L for the duration of the transduction, whereas q_2 corresponds to having seen at least one H somewhere in the string. The pattern cannot be determined locally, and so it is not OSL, ISL or EO.

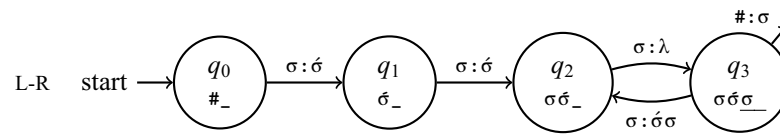
One property of properly subsequential functions that separates their expressive power from OSL functions is lookahead. Subsequential functions may contain lookahead – the 'waiting' as in (18) for some coming input before making a decision on what to output. OSL functions, however, must write a symbol to the output whenever there is a transition from one state to another. They cannot provide lookahead.

As a consequence of this, some patterns that appear local in an intuitive sense require a properly subsequential function to describe. One example is the stress pattern of Pintupi (Hansen & Hansen 1969):

(20) σ́σ, σ́σσ, σ́σσ́σ, σ́σσ́σσ, σ́σσ́σσ́σ, ...

Stress iterates left to right starting from the initial syllable and avoids the final syllable. The pattern combines iteration, demonstrated to be OSL below, with the addition of a non-finality requirement. Though this may seem inconsequential in terms of the computation of the pattern, this non-finality requirement necessitates a properly subsequential function because it requires lookahead. In other words, a single-function analysis of Pintupi is not OSL. In the trisyllable, for example, the third syllable is unstressed. However, in the four-syllable form, the third syllable *does* bear stress. This means that a transducer for the pattern cannot simply alternate between outputting stressed and unstressed syllables, because the same syllable may be output differently if it is final or not. The transducer is shown in (21):

(21)



The transducer must wait whenever it takes an odd syllable (other than the first) as its input. This waiting behaviour was also observed for LHOR stress in (18). In order to know what to output for a given odd syllable, the stress function needs to know whether there is another syllable after it. State q_2 thus corresponds to an output σ́σ, but q_3 is a waiting state – an 'odd syllable' state. It corresponds to some odd-numbered input syllable, and does not correspond to any additional output beyond that encoded in q_2. This means that information other than recent local outputs is needed to compute the function, and so it is not OSL:

(22) a. *Applying (21) to a five-syllable input*

input:	σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_2	→	q_3	→	q_2	→	q_3
output:	σ́		σ		λ		σ́σ		λ		σ

b. *Applying (21) to a six-syllable input*

input:	σ		σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_2	→	q_3	→	q_2	→	q_3	→	q_2
output:	σ́		σ		λ		σ́σ		λ		σ́σ		

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

An important question, then, is whether or not this properly subsequential characterisation of Pintupi is appropriate. It suggests that the subsequential boundary bisects the typology of iterative stress patterns and that adding a non-finality generalisation to basic iteration of stress somehow makes the pattern 'long-distance', like the LHOR pattern. It also implies that any subsequential pattern may appear in the typology of iterative stress. However, it is not the case that any subsequential function is also a possible stress pattern. For example, a pattern that stresses every odd heavy syllable in a word is definable with a subsequential function:

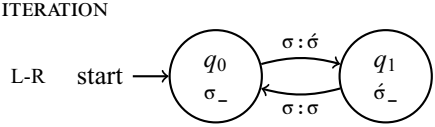
(23) a. HHLLLHH ↦ ÁHLLLÁHH
b. HHHHHH ↦ ÁHÁHÁHH
c. LLLLHHH ↦ LLLLÁHÁ
d. LÁHLLLLÁL ↦ LÁHLLLLÁL

Such a pattern is clearly pathological and should not be included in the predicted typology of stress patterns. To provide a better hypothesis regarding the typology of iterative stress, I argue that patterns like Pintupi are fundamentally local in a way that properly subsequential patterns are not. To make this property of stress computation explicit, it is proposed that the best characterisation of iterative stress is as the composition of an OSL and EO function into a combined map. The apparent need for lookahead in some iterative patterns suggests subsequential functions, but the compositional analysis shows that reference to this level of power is unnecessary: iterative patterns are merely the composition of different stress primitives that are formally local. I demonstrate that the OSL plus EO composed map is a better hypothesis for iterative stress than one that claims subsequential power, as it excludes patterns like (23).

3.4 Function composition

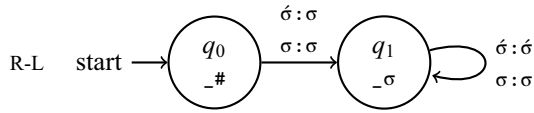
When two functions are composed, the output of the first function becomes the input for the second function. Application of successive rules in a derivational phonological analysis, such as in SPE, is a kind of function composition (Johnson 1972; Kaplan & Kay 1994). I argue that iterative stress mappings are best expressed as the composition of one OSL function and some (small) number of EO functions providing 'cleanup' that act in place of lookahead to derive the correct output. Thus, the phonological grammar is composed of these individual stress atoms, and the grammar knows which order they must apply in. To demonstrate, consider the pattern of Pintupi in (20). Instead of using a single properly subsequential function as in (21), here it is separated into an OSL function dubbed ITERATION, which achieves the basic iteration of stress, and an EO function dubbed NON-FINALITY, which removes a final stress when one is present. Here and for the remainder of the article, where there are multiple transducers, they apply in the order of their alphabetical label:

(24) a. ITERATION



https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

b. NON-FINALITY



The L-OSL function ITERATION in (24a) applies alternating stress, beginning with the initial syllable. The EO function NON-FINALITY in (24b) takes the output of the first function as its input. If the first syllable it encounters is stressed, it removes the stress. Otherwise, it outputs syllables faithfully. Since the first symbol it reads is really the last syllable, this amounts to deletion of stress on a final syllable, encoding the non-finality property of Pintupi. The only input substring that leads to a non-identity transition is the word boundary, # (which implicitly leads to the start state in (24b)), and so NON-FINALITY is EO. Reaching q_1 indicates that the relevant span of input symbols (one) to calculate the pattern has been seen, and so the only transitions leaving q_1 are identity-mapping loops back to q_1.

Examples of how the two functions interact are given here for five- and six-syllable forms. These diagrams should be read top to bottom; note how the output of ITERATION lines up vertically with and matches the input for NON-FINALITY:

(25) a. *Applying (24) to a five-syllable input*

ITERATION:	input:		σ		σ		σ		σ	
	states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1
	output:		σ́		σ		σ́		σ		σ́	
NON-FINALITY:	input:		σ́		σ		σ́		σ		σ́	
	states:	q_1	←	q_1	←	q_1	←	q_1	←	q_1	←	q_0
	output:		σ́		σ		σ́		σ		σ	

b. *Applying (24) to a six-syllable input*

ITERATION:	input:		σ		σ		σ		σ		σ	
	states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1	→	q_0
	output:		σ́		σ		σ́		σ		σ́		σ	
NON-FINALITY:	input:		σ́		σ		σ́		σ		σ́		σ	
	states:	q_1	←	q_1	←	q_1	←	q_1	←	q_1	←	q_1	←	q_0
	output:		σ́		σ		σ́		σ		σ́		σ	

ITERATION and NON-FINALITY interact to produce the correct outputs for Pintupi. It is merely the composition of local stress atoms, and requires no reference to a properly subsequential function. This is because, at a fundamental computational level, it is not a long-distance pattern. However, as demonstrated above, a single-function analysis does require use of a properly subsequential function.

The restriction to EO cleanup functions in the composition is a crucial requirement. Composition in general is powerful in that it can result in a mapping that is more expressive than any of its constituent functions. For example, Heinz and Lai (2013) show that sour-grapes harmony (Padgett 1995; Wilson 2003) can be derived from the composition of two subsequential functions working in opposite directions, even though sour grapes itself is not subsequential. McCollum *et al.* (2020) further

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

demonstrate that restrictions on interaction are necessary to avoid overgeneration in the interaction of subsequential functions.

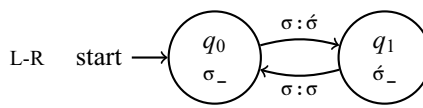
These effects are in part because of how directionality affects function composition. When composing functions, the directionality has a direct impact on the expressive power of the generalisations the composition can express. This is because the first function can essentially provide unbounded lookahead by marking the string in some way that informs the behaviour of the second function. Thus, adherence to unidirectionality in the functions constrains the complexity. For example, it is known that the composition of two unidirectional subsequential functions cannot produce a combined map that is more expressive than one that a single subsequential function could produce (Elgot & Mezei 1965).

However, this is *not* true of compositions of subsequential functions operating in opposite directions. Heterodirectional subsequential functions can describe the stress map for DTS stress patterns, which are not subsequential (Hao & Andersson 2019; Koser & Jardine 2020b). The composition of a left and right OSL function can also describe non-subsequential functions such as the hypothetical sour-grapes stress pattern from (5), repeated in (26):

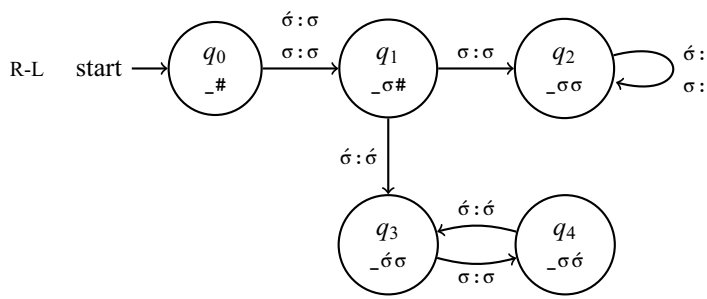
(26) σ́σ
σσσ
σ́σ́σ
σσσσσ
σ́σ́σ́σ́σ
σσσσσσσ
σ́σ́σ́σ́σ́σ́σ…

Stress iterates through the word only if it is of even length. Though parity counting is in general properly regular for stringsets (Heinz 2007b; Rogers *et al*. 2013; Graf 2017), such a stress map can be described as the composition of two heterodirectional OSL functions:

(27) a. ITERATION



b. SOUR GRAPES



https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

(27a) is ITERATION, like (24a). It reads the string from left to right, placing stress on every other syllable throughout the word starting with the first syllable. (27b) is the SOUR GRAPES function, producing the pathological aspect of the combined map. Application of ITERATION means that words of even length will end in an unstressed syllable, whereas words of odd length will end in a syllable bearing stress. SOUR GRAPES makes explicit use of this markup information. Reading the string right to left, the final syllable is always output as unstressed in the transition to q_1. The next syllable, the penult, gives the second function the crucial information it needs. A stressed penult indicates a string of even length, and so the transduction moves to q_3 which, alternating with q_4, faithfully outputs the input unchanged, preserving the stresses. An unstressed penult instead indicates an odd-parity word, and so the transduction moves to q_2, where all stresses are removed. The composition of the two OSL functions correctly derives the pathological stress map:

(28) a. *Applying (27) to a five-syllable input*

i. ITERATION:

input:	σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1
output:	σ́		σ		σ́		σ		σ́	

ii. SOUR GRAPES:

input:	σ́		σ		σ́		σ		σ́
states:	q_2	←	q_2	←	q_2	←	q_2	←	q_1	←	q_0
output:	σ		σ		σ		σ		σ	

b. *Applying (27) to a six-syllable input*

i. ITERATION:

input:	σ		σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1	→	q_0
output:	σ́		σ		σ́		σ		σ́		σ	

ii. SOUR GRAPES:

input:	σ́		σ		σ́		σ		σ́		σ
states:	q_3	←	q_4	←	q_3	←	q_4	←	q_3	←	q_1	←	q_0
output:	σ́		σ		σ́		σ		σ́		σ	

Though this is a logically possible stress map, it is clearly pathological and is disallowed by the theory of iterative stress adopted here, which only permits OSL plus EO compositions. Only one OSL function is needed to achieve the basic iteration of stress, while the EO cleanup functions encode edge-adjacent stress generalisations that complete each individual pattern. SOUR GRAPES is clearly not EO, as it may delete stresses that occur at any point in the word. Thus, allowing multiple OSL functions is not only unnecessary, it allows the generation of pathological patterns such as sour grapes stress. Adherence to OSL plus EO compositions places important constraints on the predicted typology of iterative patterns, which I conjecture removes not only sour-grapes patterns but pathological properly subsequential patterns as in (23) as well. I now turn to an analysis of various iterative stress patterns.

3.5 Relationships between function classes

The classes of functions described above form a nested hierarchy based on their expressive power, that is, the types of generalisations that functions from each class can describe. EO functions are also ISL, as they are local to the word edge in the input.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

Some – but not all – EO functions are OSL.[6] Not every OSL or ISL function is EO. For example, any OSL or ISL function that may apply anywhere in the word and is not tied to a word edge is not EO. The difference emerges partially from that fact that OSL and ISL are defined in terms of output and input substrings, whereas EO is defined in terms of finite distance to the word edge. In other words, they are calculated based on distinct intensional requirements, even when the extensional results of the functions are the same.

EO is partially motivated by the fact that reference to OSL and ISL alone does not provide a sufficiently restrictive characterisation of iterative patterns. This is because the cleanup functions necessary for iterative stress maps are both ISL *and* OSL.[7] For example, a non-finality requirement could be computed input-locally by searching for the substring 'σ́#' (i.e. final stress) in the input. It could also be computed output-locally if the function instead enforces 'σ#' in the output at the end of the word, whether the input was stressed or not. Thus, a limitation to OSL plus ISL fails to achieve the correct restrictiveness, because the particular cleanup functions in this article are both ISL and OSL. With no further restrictions, this is then an implicit hypothesis that cleanup functions may be OSL in general. In §3.4, I demonstrated that OSL plus EO compositions provide better typological predictions for iterative stress than when OSL is composed with OSL or ISL.

It is not difficult to determine that the class of EO functions does not overlap completely with the OSL or ISL class. The OSL iteration function in (14) is not EO, as it applies stress throughout the entire word. For example, given an input of bare syllables σσσσσσ…, an EO function can only stipulate the specific location for stress in a bounded window at the word edge (i.e. stress *n*, stress *n* + 2, stress *n* + 4, etc.) for all cases of stress. Not only is this insufficient to handle words of all lengths in a finite manner, it is not really iteration at all; it is an enumeration of all possible stress locations. Note that this is the same reason that an ISL function cannot describe iterative stress. Similarly, any attempt to use EO to describe an ISL function that may apply anywhere in the word results in an enumeration of the cases as the size of the word grows. In other words, while all EO functions are ISL and some are OSL, their restriction to the word edge means they are not general enough to describe all ISL or OSL patterns. For the cleanup functions in iterative stress maps discussed in this article, this is a desirable property that increases the restrictiveness of the theory while still capturing the target patterns.

4. Analyses

As described above, the OSL functions are functions that rely on information in the output string. This makes the OSL class a good hypothesis for iterative functions in general, including iterative stress, as application of successive stresses depends on the presence of another stress somewhere in the output. However, plain iteration of stress is not the only factor in iterative stress assignment. The phenomena of clash, lapse and

[6]Deletion generalisations relying on a word edge, for example, are EO and ISL, but not OSL. See Chandlee *et al*. (2015) and Chandlee and Heinz (2018) for more on why deletion is not OSL.

[7]I am indebted to an anonymous reviewer for pointing this out.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

non-finality can disrupt iteration of stress. In QS systems, a heavy syllable may disrupt the count for application of stress. Below I show that, despite surface differences, these types of patterns share fundamental computational properties, including OSL iteration of stress. This is true of both QI patterns and QS patterns, suggesting that despite a small difference in alphabet size resulting in surface differences, the core computational properties are the same.[8]

4.1 Iteration of stress

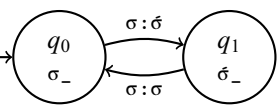
4.1.1 Binary iteration

In some languages, binary iteration is the only feature of stress assignment. This means the stress function in these languages is a simple left or right OSL function. The stress pattern of Murinbata (Street & Mollinjin 1981), given as a QI example in §3.2, is repeated in (29):

(29) σ́σ, σ́σσ́, σ́σσ́σ, σ́σσ́σσ́, σ́σσ́σσ́σ, ...

As described above, a function describing this mapping needs only keep track of what the previous output symbol was. If the previous output was a stressed syllable, then the current output will be unstressed, and vice versa. The transducer from (14) is repeated in (30):

(30) L-R



Reading left to right, the first input syllable is output as stressed, moving to q_1. The next syllable is output with no stress and we return to q_0, where the process repeats until the end of the word. The information encoded by each state makes it clear that the process is output-oriented. State q_1 encodes the fact that a stressed syllable has just been written in the output, whereas q_0 does the same for an unstressed syllable. The input string contains no such information that could be used to iteratively place stress. The function is OSL, as illustrated by the derivations in (15), repeated in (31):

(31) a. *Applying (30) to a five-syllable input*

input:	σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1
output:	σ́		σ		σ́		σ		σ́		

b. *Applying (30) to a six-syllable input*

input:	σ		σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1	→	q_0
output:	σ́		σ		σ́		σ		σ́		σ		

The same generalisation can be seen in QS languages. Fijian stress (Hayes 1995) displays binary iteration with some extra caveats related to the light/heavy syllable

[8]Note that this does abstract away from the details of syllable creation and encoding of weight that ultimately sets up the choice of a QI or QS alphabet.

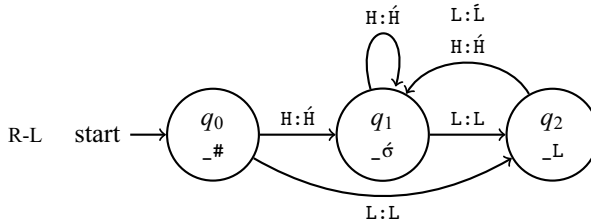
https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

distinction. Main stress falls on a heavy final syllable, or if absent, the penult. Then, additional stress is placed on all heavy syllables and on every other light syllable counting leftwards from the previous stress. This means that a heavy resets the count for iteration of stress, as seen in (32g) and (32h):

(32)
a. ĹL
b. LĹL
c. LĤ
d. ĤĤ
e. LĹLĹL
f. ĹLĤ
g. LĤĹL
h. ĹLĤĹL

If placement of stress were not sensitive to syllable weight, we would expect an initial stress in (32g), and a peninitial stress in (32h). Despite the sensitivity to weight, the stress function in Fijian is still OSL, as placement of stress once again depends on previous stressed outputs:

(33)



Reading right to left, whenever an H is seen, it is output as stressed, landing in q_1. When an L is seen in q_0 or q_1, an unstressed L is output, landing in q_2. When the transducer is in q_2, the next input syllable will be assigned stress, regardless of whether it is heavy or light, and the transducer will revert to q_1. Thus, q_1 represents having just written a stress in the output, resetting the binary count. Example derivations for LLLLL and LLHLL are given in (34):

(34)
a. *Applying (33) to an input with only light syllables, deriving (32e)*

input:	L	L	L	L	L
states:	q_2 ←	q_1 ←	q_2 ←	q_1 ←	q_2 ← q_0
output:	L	Ĺ	L	Ĺ	L

b. *Applying (33) to the input LLHLL, deriving (32h)*

input:	L	L	H	L	L
states:	q_1 ←	q_2 ←	q_1 ←	q_1 ←	q_2 ← q_0
output:	Ĺ	L	Ĥ	Ĺ	L

Note the similarity to Murinbata: q_1 encodes the same information in both (30) and (33), even though Murinbata is QI and Fijian is QS. State q_2 of Fijian is analogous to q_0 of Murinbata, the difference being that only light syllables can be output with no stress.

4.1.2 Ternary iteration

Some languages employ a ternary count for iteration of stress. When this is the only factor, stress assignment is OSL. A QI example is Ioway-Oto (Whitman 1947;

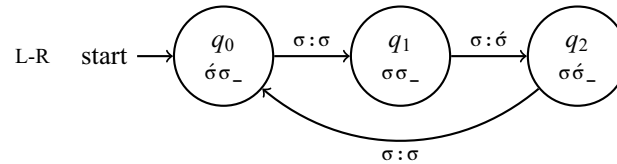
https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

Gordon 2002), which stresses the peninitial syllable and every third syllable to the right of it:

(35) σ́, σσ́, σσσ́, σσσσ́, σσσσσ́, σσσσσσ́, σσσσσσσ́…

The stress function for Ioway-Oto is like Murinbata, but with a slight delay. After each stressed syllable, it outputs two unstressed ones before the next iteration of stress:[9]

(36)



This transducer represents a TERNARY ITERATION function. Reading left to right, the peninitial receives stress in the transition from q_1 to q_2. The next two syllables are output as unstressed, as the transducer returns to q_0 and then q_1. This ternary cycle continues for the length of the word. This captures the ternary iteration of Ioway-Oto and indicates that, just like the binary iteration of Murinbata, it is OSL:

(37) a. σσσσσσσσ ↦ σσ́σσ́σσ
b. σσσσσσσσσ ↦ σσ́σσ́σσ́σ
c. σσσσσσσσσσ ↦ σσ́σσ́σσ́σσ

An example derivation for an eight-syllable input is given in (38):

(38) *Applying (36) to an eight-syllable input*

input:	σ	σ	σ	σ	σ	σ	σ	σ
states:	q_0	q_1	q_2	q_0	q_1	q_2	q_0	q_1	q_2
output:	σ	σ́	σ	σ	σ́	σ	σ	σ́

Ternary iteration of stress occurs in QS systems as well. As with binary iteration, despite descriptive differences, the input–output map remains an OSL function. One example is Sentani (Cowan 1965; Hayes 1995). In Sentani, the final syllable is stressed if it is heavy, as in (39a) and (39d). Otherwise, the penult is stressed instead, as in the other examples in (39). Ternary iteration proceeds leftwards from the stressed penultimate or final syllable, as in (39c) and (39d). Iteration is interrupted by heavy syllables, which receive stress as in (39e) and (39f), unless this would create a clash, as in (39g) and (39h).

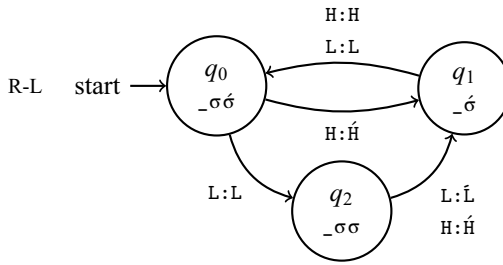
(39) a. LH́
b. H́L
c. ĹLLĹL
d. LLĹLLH́
e. LH́LĹL
f. LH́LH́L
g. HĹL
h. LH́LHH́L

[9]Note that accounting for the monosyllabic form would require a waiting transition with the empty string λ, which by definition would make the function non-OSL. This is true of every QI pattern where main stress is not anchored directly on the left or right word edge. I leave this as a question for future research.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

As with Fijian, the stress function for Sentani is OSL, despite the additional requirements related to weight and clash. Iteration of stress is output-oriented, seen in the transducer as a return to q_1 whenever stress is applied. No such generalisation about the input string is possible:

(40)



Reading right to left, a final heavy syllable is stressed in the transition from q_0 to q_1, or the penult is stressed in the path from q_2 to q_1. From here, ternary iteration of stress applies, with q_1 representing having just seen a stress in the *output* string. The transition from q_0 to q_1 represents the reset of iteration by a heavy syllable. Clash is disallowed, as the transition from q_1 (having just output a stress) to q_0 produces only unstressed syllables. Reaching q_2 indicates that an unstressed syllable and an L were just output, and so the following symbol is stressed regardless of weight, returning again to q_1. The stress map for Sentani is R-OSL:

(41) a. LH $\mapsto$ LH́
b. LLLLL $\mapsto$ ĹLLĹL
c. LHLLL $\mapsto$ LH́LĹL
d. LHLHHL $\mapsto$ LH́LHH́L

Example derivations for LLLLL and LHLHHL are given in (42):

(42) a. *Applying (40) to the input LLLLL, deriving (39c)*

input:	L		L		L		L		L
states:	q_1	$\leftarrow$	q_2	$\leftarrow$	q_0	$\leftarrow$	q_1	$\leftarrow$	q_2	$\leftarrow$	q_0
output:	Ĺ		L		L		Ĺ		L		

b. *Applying (40) to the input LHLHHL, deriving (39h)*

input:	L		H		L		H		H		L
states:	q_0	$\leftarrow$	q_1	$\leftarrow$	q_2	$\leftarrow$	q_0	$\leftarrow$	q_1	$\leftarrow$	q_2	$\leftarrow$	q_0
output:	L		Ĥ		L		H		Ĥ		L		

The analysis undertaken in this section indicates that, in general, iteration of stress is an OSL function, regardless of whether it is binary or ternary, or whether it is QI or QS. Indeed, all patterns analysed below require some version of (30) or (33) to model iterative application of stress. While this by itself is sufficient for languages like Murinbata and Fijian, other types of patterns will require additional machinery: application of an additional function corresponding to the atomic elements of the stress pattern such as non-finality, lapse and clash.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

4.2 Non-finality

In systems with non-finality, iteration of stress is disrupted by a requirement that the final syllable surface as unstressed. As discussed in §3.3, one QI example is Pintupi (Hansen & Hansen 1969), which has the pattern in (20), repeated in (43):

(43) σ́, σ́σσ, σ́σσ́, σ́σσ́σσ, σ́σσ́σσ́, …

The ITERATION function alone does not capture the Pintupi pattern. Unlike Murinbata, the pattern requires a small amount of lookahead that is not available to OSL functions. Rather, a transducer for Pintupi as a single function, as in (21), is properly subsequential, because information other than the most recent output or input is required to compute the function.

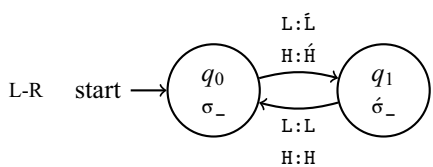
Though this suggests that iteration of stress in some languages is more complex than in other languages, it was demonstrated that separating the limited lookahead from the iteration of stress by dividing the map into two different functions reveals underlying similarities among iterative patterns and provides a more restrictive theory of stress than allowing the full range of subsequential functions. It preserves the formal notion of locality present in OSL and EO functions but absent in properly subsequential functions. Both Pintupi and Murinbata use the OSL ITERATION function, whereas Pintupi requires an additional cleanup step that acts in place of lookahead to enforce non-finality. This comes in the form of the EO function that removes a final stress if present, seen in (24).

A QS version of non-finality appears in Wergaia (Hercus 1986; Hyde 2011). As shown in (44), stress iterates from left to right, starting with the first syllable. An odd-numbered final syllable will be stressed if it is heavy, as in (44d), but stress avoids light final syllables, as in (44e) and (44f).

(44) a. ĹL
b. ĺL
c. ĹLĹL
d. ĹLĺ
e. ĺLL
f. ĹHL

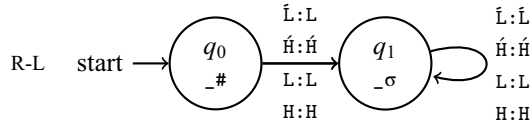
Though this binary weight distinction for non-finality sets Wergaia apart from Pintupi in a descriptive sense, the computation of the stress map is of equal complexity. This is because the decision whether to output an odd-numbered *light* syllable as stressed depends on whether the current position is at the end of the word. Like the single-function analysis for Pintupi, this is not OSL: it is properly subsequential because of the waiting. It is also amenable to the same compositional analysis as Pintupi, separating the iteration of stress from the non-finality requirement:

(45) a. ITERATION



https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

b. QUANTITY-SENSITIVE NON-FINALITY



The L-OSL transducer in (45a) is an ITERATION function, placing stress on every odd-numbered syllable regardless of weight. This means that, despite the different input and output symbols, the application of the function is identical to the QI ITERATION function in (24a). The transducer in (45b) is QS NON-FINALITY. Just as in the QI NON-FINALITY function of Pintupi in (24b), the only input substring leading to a non-identity transition is the word boundary #, and so it is EO. Taking the output of the first function as its input and working from right to left, it removes stress from final light syllables, otherwise outputting the word faithfully and modelling the weight-specific non-finality requirement of Wergaia. This highlights the fact that, despite the different alphabets, the information encoded in the states is identical to that of Pintupi in (24), reinforcing the fact that the computation of the functions in both cases is the same.

(46)

		(45a)		(45b)	
a.	LLLL	↦	ĹLĹL	↦	ĹLĹL
b.	LLH	↦	ĹLḾ	↦	ĹLḾ
c.	HLL	↦	ḾLĹ	↦	ḾLL

Derivations for odd-parity inputs with light and heavy final syllables are shown in (47):

(47) a. *Applying (45) to the input LLLLL*

i. ITERATION:

input:		L		L		L		L		L
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→ q_1
output:		Ĺ		L		Ĺ		L		Ĺ

ii. QS NON-FINALITY:

input:		Ĺ		L		Ĺ		L		Ĺ
states:	q_1	←	q_1	←	q_1	←	q_1	←	q_1	← q_0
output:		Ĺ		L		Ĺ		L		L

b. *Applying (45) to the input LLLLH*

i. ITERATION:

input:		L		L		L		L		H
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→ q_1
output:		Ĺ		L		Ĺ		L		Ḿ

ii. QS NON-FINALITY:

input:		Ĺ		L		Ĺ		L		Ḿ
states:	q_1	←	q_1	←	q_1	←	q_1	←	q_1	← q_0
output:		Ĺ		L		Ĺ		L		Ḿ

Despite the differences in the surface generalisations of Pintupi and Wergaia, the fundamental properties underlying the calculation of stress in the two languages are identical. This kind of compositional analysis makes the computational properties of different stress phenomena explicit: iteration of stress is OSL and non-finality is EO, regardless of weight-sensitivity.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

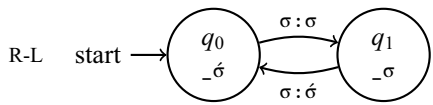
4.3 Internal lapse

Languages that display an internal lapse can also be analysed compositionally in this way. QI examples include Garawa (Furby 1974) and Piro (Matteson 1965). To the best of my knowledge, there are no iterative QS internal-lapse counterparts. The pattern of Garawa is as in (48):

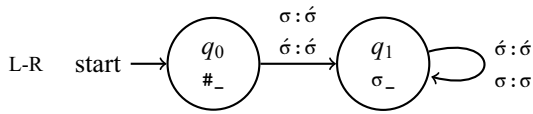
(48) ́σ, ́σσ, ́σσ́σ, ́σσσ́σ, ́σσ́σσ́σ, ́σσσ́σσ́σ …

The initial syllable is always stressed. Stress iterates right to left from the penult, but avoids the peninitial syllable if a stress there would create a clash. Since ITERATION in this case must apply R-L, the OSL function will fail to stress the initial syllable and erroneously stress the peninitial syllable in odd-parity forms. This is again due to the lack of lookahead in the OSL function: a fully subsequential function could provide the necessary lookahead, but an OSL function cannot. This means that two EO cleanup steps are necessary, one placing an initial stress where absent, and a second to resolve clashes that the first would produce:

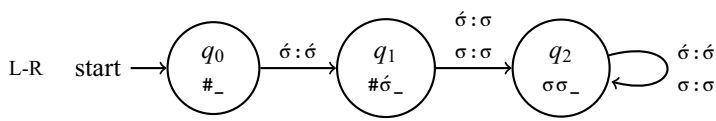
(49) a. ITERATION



b. INITIAL STRESS



c. ANTI-CLASH



The R-OSL ITERATION transducer in (49a) iterates stress right to left starting with the penult. Taking the output of the first function as its input and reading the string left to right, the INITIAL STRESS function in (49b) adds an initial stress if it is absent, but otherwise leaves the string unchanged. Then, the ANTI-CLASH function in (49c) resolves any potential clashes this would create in the transition from q_1 to q_2. For Garawa, this clash resolution results in the characteristic internal lapse in odd-parity forms. The rest of the string is left unchanged. Both INITIAL STRESS and ANTI-CLASH are EO, as a finite number of strings lead to a state that could apply a change to the input. For INITIAL STRESS, this is the word boundary #, and for ANTI-CLASH, the sole input sequence is #σ́. The following examples demonstrate the effect of the composed mapping:

(50)

		(49a)		(49b)		(49c)	
a.	σσσσσσ	↦	́σσ́σσ́σ	↦	́σσ́σσ́σ	↦	́σσ́σσ́σ
b.	σσσσσσσ	↦	σ́σσ́σσ́σ	↦	́σσ́σσ́σσ́	↦	́σσσ́σσ́σ

Even-parity forms, as in (50a), are unchanged by the EO functions: the action of the OSL function alone provides the correct output. Odd-parity forms, as in (50b),

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

have the first two syllables altered by the EO functions, creating an internal lapse. Though Garawa features a pair of EO functions instead of just one as in Pintupi, the generalisations in both languages adhere to the single-OSL restriction with EO cleanup functions acting in place of the lookahead seen in the subsequential transducers. This indicates that along with non-finality, the forces that create internal lapse are also EO when viewed as atomic stress elements. For clarity, the following derivation shown begins after the ITERATION function in (49a) has already applied:

(51) a. *Applying (49) to a five-syllable input*

i. INITIAL STRESS:	input:		σ		σ́		σ		σ́		σ
	states:	q_0	→	q_1	→	q_1	→	q_1	→	q_1	→	q_1
	output:		σ́		σ́		σ		σ́		σ
ii. ANTI-CLASH:	input:		σ́		σ́		σ		σ́		σ
	states:	q_0	→	q_1	→	q_2	→	q_2	→	q_2	→	q_2
	output:		σ́		σ		σ		σ́		σ

b. *Applying (49) to a six-syllable input*

i. INITIAL STRESS:	input:		σ́		σ		σ́		σ		σ́		σ
	states:	q_0	→	q_1	→	q_1	→	q_1	→	q_1	→	q_1	→	q_1
	output:		σ́		σ		σ́		σ		σ́		σ
ii. ANTI-CLASH:	input:		σ́		σ		σ́		σ		σ́		σ
	states:	q_0	→	q_1	→	q_2	→	q_2	→	q_2	→	q_2	→	q_2
	output:		σ́		σ		σ́		σ		σ́		σ

4.4 Clash

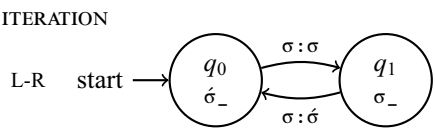
Thus far, the patterns examined have explicitly avoided placing two stresses next to each other. Clash patterns do the opposite, allowing adjacent stresses. A QI example comes from Ojibwe (Kaye 1973), shown here:

(52) σσ́, σσ́σ́, σσ́σσ́, σσ́σσ́σ́, σσ́σσ́σσ́, …

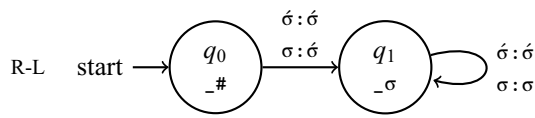
Stress iterates left to right from the peninitial syllable, and the final syllable is always stressed, even if this results in a clash. In odd-parity forms, binary iteration will not add a final stress, and so an EO function does the job instead.

Like Pintupi, this pattern requires bounded lookahead. The decision to stress the third syllable, for example, cannot be made until the next symbol is read. Whereas Pintupi must avoid stressing the final syllable, Ojibwe is required to do so:

(53) a. ITERATION



b. FINALITY



https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

ITERATION in (53a) iterates stress left to right starting from the peninitial. This is identical to the Garawa transducer in (49a), 5but operating in the opposite direction. The function in (53b), dubbed FINALITY, adds a final stress if one is absent, but otherwise leaves the string unchanged. This is the inverse of the NON-FINALITY function seen for Pintupi in (24b). Just like NON-FINALITY, the only input substring leading to a non-identity transition is the word boundary #, and so FINALITY is also EO:

(54)

		(53a)		(53b)	
a.	σσσσσσ	↦	σσσσσσ	↦	σσσσσσ
b.	σσσσσσσ	↦	σσσσσσσ	↦	σσσσσσσ

Once again, the EO function demonstrates that the lookahead required to describe the pattern as a single subsequential function is unnecessary in a compositional analysis. Above, it was shown that non-finality and lapse creation are EO. Now, clash can be added to this group of EO stress functions. Example derivations for five- and six-syllable words are given in (55):

(55) a. *Applying (53) to a five-syllable input*

i. ITERATION:

input:	σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1
output:	σ		σ́		σ		σ́		σ

ii. FINALITY:

input:	σ		σ́		σ		σ́		σ
states:	q_1	←	q_1	←	q_1	←	q_1	←	q_1	←	q_0
output:	σ		σ́		σ		σ́		σ́

b. *Applying (53) to a six-syllable input*

i. ITERATION:

input:	σ		σ		σ		σ		σ		σ
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1	→	q_0
output:	σ		σ́		σ		σ́		σ		σ́

ii. FINALITY:

input:	σ		σ́		σ		σ́		σ		σ́
states:	q_1	←	q_1	←	q_1	←	q_1	←	q_1	←	q_1	←	q_0
output:	σ		σ́		σ		σ́		σ		σ́

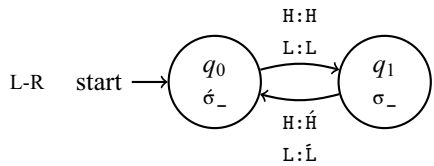
In QS languages, a clash may occur incidentally when heavy syllables fall next to another stress that resulted from binary iteration, if the language requires all heavy syllables to be stressed. This can be seen for Fijian in (32). Another interesting case is that of Cayuga (Foster 1982). In Cayuga, stress iteratively falls on even-numbered syllables counting from the left, with some further requirements. When the penult is even, it receives main stress regardless of weight, as in (56a) and (56b). An odd penult also receives main stress if it is heavy, as in (56c). If the penult is odd and light, however, the antepenult is stressed instead, as in (56d):

(56) a. LĹLĹL
b. HĹLĹHL
c. LĹLĹĹHL
d. LĹLĹLL

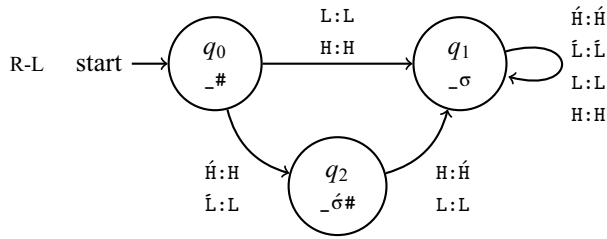
https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

Relying not only on weight but on an apparent even/odd parity distinction, the pattern of Cayuga looks on the surface to be more complex than the others discussed to this point. However, it too breaks down into an OSL iteration function and an EO cleanup function. This is partly because the iterative placement of stress provides a pseudo-parity count on the surface in a way that is local, without explicitly tracking the parity of the entire word or a specific syllable:

(57) a. ITERATION



b. CAYUGA STRESS



The L-OSL function in (57a) is another example of binary ITERATION. It applies stress to every other syllable starting from the peninitial, regardless of weight. The function CAYUGA STRESS in (57b) corrects overapplications of stress. When it encounters an unstressed final syllable, this indicates that the penult is even and stressed, and so the transduction moves to q_1 where everything is output faithfully. If instead it encounters a stressed final syllable, it removes the stress and moves to q_2. From here, a heavy syllable is stressed, whereas a light syllable is left unchanged, after which the transducer moves to q_1.

Despite some conceptual similarities to the SOUR GRAPES function discussed in §3.4, (57b) is EO – the only input substring leading to a non-identity transition is σ# – whereas the SOUR GRAPES function in (27b) is not. Thus, the composed map adheres to the stated single-OSL plus EO restriction:

(58)

		(57a)		(57b)	
a.	LLLLL	↦	LĹLĹL	↦	LĹLĹL
b.	HLLHL	↦	HĹLĤL	↦	HĹLĤL
c.	LLLLHL	↦	LĹLĹHĹ	↦	LĹLĹĤL
d.	LLLLLL	↦	LĹLĹLĹ	↦	LĹLĹLL

Derivations for inputs LLLHL and LLLLLL are given in (59):

(59) a. *Applying (57) to an input with a heavy even penult*

i. ITERATION:

input:	L		L		L		H		L	
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→ q_1
output:	L		Ĺ		L		Ĥ		L	

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

ii. CAYUGA STRESS:

input:	L		Ĺ		L		Ḿ		L
states:	q_1	←	q_1	←	q_1	←	q_1	←	q_1	←	q_0
output:	L		Ĺ		L		Ḿ		L

b. *Applying (57) to an input with a light odd penult*

i. ITERATION:

input:	L		L		L		L		L		L
states:	q_0	→	q_1	→	q_0	→	q_1	→	q_0	→	q_1	→	q_0
output:	L		Ĺ		L		Ĺ		L		Ĺ

ii. CAYUGA STRESS:

input:	L		Ĺ		L		Ĺ		L		Ĺ
states:	q_1	←	q_1	←	q_1	←	q_1	←	q_1	←	q_2	←	q_0
output:	L		Ĺ		L		Ĺ		L		L

So far it has been shown that bounded lookahead is a property shared by a wide range of unidirectional iterative patterns in both QI and QS stress. The inability of the OSL stress ITERATION function to provide any lookahead is made up for by EO cleanup functions that instantiate the phonological properties of clash, lapse, and non-finality in way that is restrictive and captures their limitation to the word edge. Despite the descriptive differences and apparent surface complexity of some of the patterns, the explicit compositional analyses confirm that they all share fundamental OSL-plus-EO computational properties, described here as the atomic properties of stress. While surface differences that appear quite substantial suggest a more disparate typology for stress when taken at face value, the systems are actually of equal complexity.

5. Discussion

5.1 Levels of stress

To this point, the distinction between primary and secondary stress has been ignored. This is only to communicate the generalisations drawn above with as much clarity as possible – it is not a rejection of levels of stress or theories that refer to them. If the ITERATION functions from above applied secondary stress instead, then a PRIMARY STRESS function could apply afterwards and promote an existing secondary stress. In Murinbata, for example, the initial stress is promoted:

(60) a. ə̀sə́ə̀sə̀ ↦ ə́sə́ə̀sə̀

b. ə̀sə́ə̀sə̀sə ↦ ə́sə́ə̀sə̀sə

It is apparent that PRIMARY STRESS is identical to the other EO cleanup functions presented above, provided that main stress is always anchored a fixed distance from a word edge. This is the case in all cases of stress that I am aware of. The function thus adheres to the same restriction of OSL plus EO. A similar function can be added to any of the analyses presented above to provide a complete picture that includes different levels of stress. Note that this assumes a 'bottom-up' method of assigning primary stress. Future work could investigate differences in computation of bottom-up and top-down analyses of main stress.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

5.2 Multidirectional patterns

The analyses above demonstrated a theory of iterative stress patterns where stress is applied in one direction. There are, however, iterative patterns that are truly bidirectional in that they apply stress iteratively both left to right and again right to left.[10] To avoid potential terminological confusion, I adopt the term 'multidirectional' for these patterns at the suggestion of an anonymous reviewer. One example is Cahuilla (Seiler 1977; Levin 1988; Idsardi 1992; Hayes 1995), which, in all-light-syllable forms, stresses the first syllable of the root and then alternating syllables in both directions away from that first stress. This could be modelled with a pair of OSL functions reading the string in opposite directions: one L-OSL function to start and iterate stress from the root and one R-OSL function to iterate stress the other direction, that is, into the prefix. This is essentially a pair of the OSL stress iteration functions seen throughout the article with the added requirement to pay attention to root boundaries. However, since the composition contains two OSL functions, it does not match the restriction to OSL plus EO adopted here. How the restriction imposed by EO might be extended to bidirectional OSL plus OSL processes without allowing gross overgeneration is a question for future research.

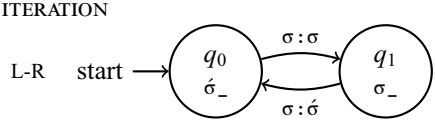
5.3 Parity-counting patterns

The proposal that iterative stress is the composition of an OSL function with EO functions is a testable hypothesis about the typology of iterative stress patterns. It makes explicit claims about what a possible iterative pattern can be. This is exemplified by the pattern of Creek (Haas 1977; Halle & Vergnaud 1987; Hayes 1995). In words with only light syllables, stress falls on whichever of the penult or ultima is even, counting from the left (this ignores syllable weight and its effects, which are orthogonal to the point made here):

(61) a. σσσσ́
b. σσσσ́σ

Parity counting, in general, is properly regular when considered as a formal language (i.e. stringset; Heinz 2007b; Rogers *et al.* 2013; Graf 2017). This suggests that the stress *function* for Creek may be of a higher complexity than what is needed for iterative stress. This is demonstrated by the fact that an OSL plus OSL composition could derive the correct stress map for Creek by using stress in the intermediate form as a pseudo-parity count:

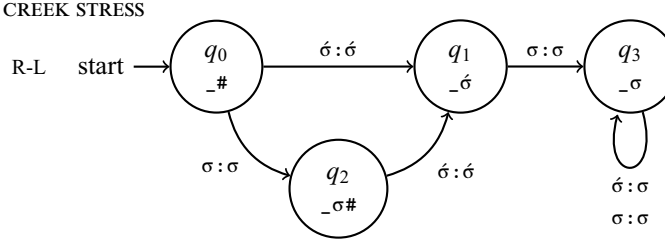
(62) a. ITERATION



[10]Thanks to Bill Idsardi for pointing these out.

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

b. CREEK STRESS



The L-OSL function in (62a) stresses every other syllable starting with the peninitial, just like other iterative functions seen above. The second function in (62b) reads right to left, keeping the first stress it encounters but removing all others. This approach is analogous to that of Halle & Vergnaud (1987), which also employs a stress-removal function dubbed 'conflation'. The compositional analysis here generates the attested surface pattern:

(63)

	(62a)		(62b)	
a. σσσσσ	↦	σσ́σσ́σ	↦	σσσσ́σ
b. σσσσσσ	↦	σσ́σσ́σσ́	↦	σσσσσσ́

The potentially unlimited alteration to the string in the form of deleting all stresses except the rightmost one is a straightforward violation of the restrictions imposed on EO functions, and so such a pattern is excluded from the typology of iterative stress.

However, there is some evidence that Creek behaves like Cayuga above, preserving additional stresses (Martin & Johnson 2002; Martin 2011). This would make Creek an iterative pattern that *does* adhere to the EO restriction. Notably, the difference in complexity between the two shows that, separate from any other factors contributing to its existence, secondary stress is a computational aid that is directly responsible for the lower level of complexity of iterative patterns when compared to unbounded patterns. It also shows that, as a restriction, EO makes substantive distinctions when it comes to stress typology.

5.4 Long-distance patterns

Other attested patterns excluded here are truly long-distance patterns, such as DTO and DTS stress, of which the latter is not even subsequential (Hao & Andersson 2019; Koser & Jardine 2020b). These are excluded because they involve functions that are neither OSL or EO. Take, for example, the DTO pattern 'LHOR', found in Kwakw'ala (Hayes 1995). In §3.3, it was demonstrated that this type of pattern can be calculated by a subsequential function. Remember, however, that full subsequential power overgenerates by predicting pathological counting patterns such as the 'stress every odd heavy syllable' pattern described in (23). A legitimate question, then, is whether reference to subsequential functions is necessary, or if some more restrictive characterisation of the 'atoms' of long-distance patterns can be provided, as it was for the iterative patterns described above.

Preliminarily, the answer to this question is yes – Burness & McMullin (2020) describe the *strictly piecewise* (SP) class of functions, an extension of the SP languages

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

(Heinz 2009; Rogers *et al*. 2010). This class of functions is sufficient to capture many long-distance phonological patterns, but is more restrictive than full subsequential power. Intuitively, in an SP function, each input symbol has a *consistent effect* on the behaviour of other inputs that appear further along in the string. An example comes from sibilant harmony in Aari (Hayward 1990), where an underlying /s/ surfaces as [ʃ] if another [ʃ] appears at any earlier point in the word. This means that [ʃ] determines the behaviour of all subsequent tokens of underlying /s/ in a consistent, long-distance manner: only [ʃ] will ever surface.

Though a DTO stress pattern such as LHOR requires subsequential power as a single function, it can be expressed as the composition of an EO function and an SP function. The EO function places the default 'rightmost' stress, and the SP function stresses the leftmost heavy. From this point, no other input symbols may surface with stress. Thus, a stressed heavy has a consistent, long-distance effect on the rest of the word that can be described with an SP function. Some example derivations are given in (64):

(64)

		RIGHTMOST		LEFTMOST H	
a.	LLLLLL	↦	LLLLLĹ	↦	LLLLLĹ
b.	LHLLHL	↦	LHLLHĹ	↦	LĤLLHL
c.	LLLLHL	↦	LLLLHĹ	↦	LLLLĤL

Though a full treatment of the atomic properties of long-distance stress patterns is beyond the scope of this article, the analysis provided here suggests that long-distance patterns may also be treated as the composition of simpler individual atoms, even when a single-function analysis requires properly subsequential power. Just as with iterative patterns, this eliminates unwanted typological predictions while still allowing attested patterns such as LHOR to be described. SP functions are less expressive than subsequential functions: though SP functions describe long-distance processes, they cannot derive a pathological pattern such as 'stress every odd heavy syllable'. This is precisely because, in such a pattern, the effect of a given input symbol on further inputs is *inconsistent.* For example, both [Ĥ…H] and [Ĥ…Ĥ] are licit configurations, because the decision to stress a given heavy syllable depends on its parity, not the preceding symbols in the word. Thus, just as OSL plus EO compositions provide a tighter characterisation of iterative patterns than the subsequential class, an EO plus SP composition avoids the overgeneration possible with subsequential functions for at least DTO long-distance patterns as well. This is analogous to the result of Rogers *et al*. (2013), who found that stress patterns as formal languages could be factored into combinations of local and piecewise constraints.

6. Conclusion

This article has presented a theory of unidirectional iterative stress as the composition of its constituent parts or 'atoms'. These atoms are local functions that correspond to basic stress generalisations seen in the literature, such as non-finality, clash, lapse and the basic iteration of stress. Specifically, iterative stress patterns were shown to be the composition of an OSL iteration function and a small number of EO 'cleanup'

https://doi.org/10.1017/S0952675723000076 Published online by Cambridge University Press

functions that make changes when OSL iteration is not enough to capture the particular pattern alone.

This approach is preferable to a single-function analysis because it highlights the complexity of the individual atoms and demonstrates that various patterns with different surface characteristics, such as being QI or QS, share fundamental computational properties. It also preserves a well-defined notion of locality that is lost in the single function analysis, which necessarily combine lookahead with iteration, making them neither OSL nor EO. The composed map, however, emphasises that these patterns are indeed local.

Acknowledgements. I would like to thank Adam Jardine, Adam McCollum, and Bruce Tesar – along with everyone else at Rutgers Linguistics – for their guidance in this work. The comments of three anonymous reviewers, as well as the editing team at *Phonology,* also leave the article immeasurably improved. I am forever thankful to all involved.

Competing interests. The author declares no competing interests.

References

Bach, Emmon (1975). Long vowels and stress in Kwakiutl. *Texas Linguistics Forum* **2**. 9–19.
Baek, Hyunah (2018). Computational representation of unbounded stress. *CLS* **53**. 13–24.
Bailey, Todd (1995). *Nonmetrical constraints on stress*. PhD dissertation, University of Minnesota.
Booij, Geert E. (1983). Principles and parameters in prosodic phonology. *Linguistics* **21**. 249–280.
Buckley, Eugene (2009). Locality in metrical typology. *Phonology* **26**. 389–435.
Burness, Phillip & Kevin McMullin (2020). Modelling non-local maps as strictly piecewise functions. *Proceedings of the Society for Computation in Linguistics* **3**. Article no. 60.
Chandlee, Jane (2014). *Strictly local phonological processes*. PhD dissertation, University of Delaware.
Chandlee, Jane, Rémi Eyraud & Jeffrey Heinz (2015). Output strictly local functions. In *Proceedings of the 14th Meeting on the Mathematics of Language*. Chicago, IL: Association for Computational Linguistics. 112–125.
Chandlee, Jane & Jeffrey Heinz (2018). Strict locality and phonological maps. *LI* **49**. 23–60.
Chomsky, Noam & Morris Halle (1968). *The sound pattern of English*. New York: Harper and Row.
Cowan, H. K. J. (1965). *Grammar of the Sentani language*. Number 47 in Verhandelingen van het Koninklijk Instituut voor Taal-, Land- en Volkenkunde. Dordrecht: Springer.
Dolatian, Hossep, Nate Koser, Jon Rawski & Kristina Strother-Garcia (2021). Computational restrictions on iterative prosodic processes. In Ryan Bennett, Richard Bibbs, Mykel L. Brinkerhoff, Max J. Kaplan, Stephanie Rich, Amanda Rysling, Nicholas Van Handel & Maya Wax Cavallaro (eds.) *Supplemental proceedings of the 2020 Annual Meeting on Phonology*. Washington, DC: Linguistic Society of America, 12 pp.
Dresher, B. Elan & Jonathan D. Kaye (1990). A computational learning model for metrical phonology. *Cognition* **34**. 137–195.
Eisner, Jason (1997). What constraints should OT allow? Paper presented at the Annual Meeting of the Linguistic Society of America, Chicago, January 1997. ROA #204.
Elgot, Calvin C. & Jorge E. Mezei (1965). On relations defined by generalized finite automata. *IBM Journal of Research and Development* **9**. 47–67.
Foster, Michael K. (1982). Alternating weak and strong syllables in Cayuga words. *IJAL* **48**. 59–72.
Furby, Christine E. (1974). Garawa phonology. In Christine E. Furby, Luise A. Hercus & Christine Kilham (eds.) *Papers in Australian linguistics*, vol. **7**. Number 37 in Pacific Linguistics Series A. Canberra: Linguistic Circle of Canberra. 1–12.
Gordon, Matthew (2002). A factorial typology of quantity-insensitive stress. *NLLT* **20**. 491–552.
Graf, Thomas (2017). The power of locality domains in phonology. *Phonology* **34**. 385–405.
Haas, Mary (1977). Tonal accent in Creek. *Southern California Occasional Papers in Linguistics* **4**. 195–208.

Halle, Morris & Jean-Roger Vergnaud (1978). *Metrical structures in phonology*. Ms., Massachusetts Institute of Technology.
Halle, Morris & Jean-Roger Vergnaud (1987). *An essay on stress*. Cambridge, MA: MIT Press.
Hammond, Michael (1984). *Constraining metrical theory*. PhD dissertation, University of California, Los Angeles.
Hansen, Kenneth Carl & Lesley Ellen Hansen (1969). Pintupi phonology. *Oceanic Linguistics* **8**. 153–170.
Hao, Yiding & Samuel Andersson (2019). Unbounded stress in subregular phonology. In Garrett Nicolai & Ryan Cotterell (eds.) *The 16th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*. Stroudsburg, PA: Association for Computational Linguistics. 135–143.
Hayes, Bruce (1981). *A metrical theory of stress rules*. PhD dissertation, Massachusetts Institute of Technology.
Hayes, Bruce (1995). *Metrical stress theory: principles and case studies*. Chicago, IL: University of Chicago Press.
Hayward, Richard J. (1990). Notes on the Aari language. In Richard J. Hayward (ed.) *Omotic language studies*. London: School of Oriental and African Studies. 425–493.
Heinz, Jeff (2018). The computational nature of phonological generalizations. In Larry M. Hyman & Frans Plank (eds.) *Phonological typology*. Berlin: De Gruyter Mouton. 126–195.
Heinz, Jeffrey (2007a). *The inductive learning of phonotactic patterns*. PhD dissertation, University of California, Los Angeles.
Heinz, Jeffrey (2007b). Learning unbounded stress systems via local inference. *NELS* **37**. 261–274.
Heinz, Jeffrey (2009). On the role of locality in learning stress patterns. *Phonology* **26**. 303–351.
Heinz, Jeffrey (2014). Culminativity times harmony equals unbounded stress. In Harry van der Hulst (ed.) *Word stress: theoretical and typological issues*. Cambridge: Cambridge University Press. 255–275.
Heinz, Jeffrey & Regine Lai (2013). Vowel harmony and subsequentiality. In András Kornai & Marco Kuhlmann (eds.) *Proceedings of the 13th Meeting on Mathematics of Language*. Stroudsburg, PA: Association for Computational Linguistics. 52–63.
Hercus, Luise (1986). *Victorian languages: a late survey*. Number 77 in Pacific Linguistics Series B. Canberra: Australian National University.
Hyde, Brett (2002). A restrictive theory of metrical stress. *Phonology* **19**. 313–359.
Hyde, Brett (2008). *Alignment constraints: distance-sensitivity, order-sensitivity, and the midpoint pathology*. Ms., Washington University. ROA #998.
Hyde, Brett (2011). Extrametricality and non-finality. In Marc van Oostendorp, Colin J. Ewen, Elizabeth Hume & Keren D. Rice (eds.) *The Blackwell companion to phonology*, vol. **2**. Oxford: Wiley-Blackwell. 1027–1051.
Hyman, Larry M. (1977). On the nature of linguistic stress. *Southern California Occasional Papers in Linguistics* **4**. 37–82.
Idsardi, William (1992). *The computation of prosody*. PhD dissertation, Massachusetts Institute of Technology.
Jardine, Adam (2016). Computationally, tone is different. *Phonology* **33**. 385–405.
Johnson, Douglas (1972). *Formal aspects of phonological description*. The Hague: Mouton.
Kager, René (1992). Are there any truly quantity-insensitive systems? *BLS* **18**. 123–132.
Kager, René (2005). Rhythmic licensing: an extended typology. In *Proceedings of the 3rd International Conference on Phonology*. Seoul: The Phonology–Morphology Circle of Korea. 5–31.
Kager, René (2012). Stress in windows: language typology and factorial typology. *Lingua* **122**. 1454–1493.
Kaplan, Ronald & Martin Kay (1994). Regular models of phonological rule systems. *Computational Linguistics* **20**. 331–378.
Kaye, Jonathan D. (1973). Odawa stress and related phenomena. In Glyne L. Piggott & Jonathan D. Kaye (eds.) *Odawa language project: second report*. Toronto: Centre for Linguistic Studies, University of Toronto. 42–50.
Koser, Nate & Adam Jardine (2020a). The complexity of optimizing over strictly local constraints. *Penn Working Papers in Linguistics* **26**. Article no. 15.
Koser, Nate & Adam Jardine (2020b). The computational nature of stress assignment. In Hyunah Baek, Chikako Takahashi & Alex Hong-Lun Yeung (eds.) *Proceedings of the 2019 Annual Meeting on Phonology*. Washington, DC: Linguistic Society of America, 11 pp.

Levin, Juliette (1988). Bi-directional foot construction as a window on level ordering. In Michael Hammond & Michael Noonan (eds.) *Theoretical morphology*. Leiden: Brill. 339–352.
Liberman, Mark & Alan Prince (1977). On stress and linguistic rhythm. *LI* **8**. 249–336.
Luo, Huan (2017). Long-distance consonant agreement and subsequentiality. *Glossa* **2**. 52.
Martin, Jack (2011). *A grammar of Creek (Muskogee)*. Lincoln: University of Nebraska Press.
Martin, Jack & Keith Johnson (2002). An acoustic study of 'tonal accent' in Creek. *IJAL* **68**. 28–50.
Matteson, Esther (1965). *The Piro (Arawakan) language*. Number 22 in University of California Publications in Linguistics. Berkeley, CA: University of California Press.
McCollum, Adam, Eric Baković, Anna Mai & Eric Meinhardt (2020). Unbounded circumambient patterns in segmental phonology. *Phonology* **37**. 215–255.
Mohri, Mehryar (1997). Finite-state transducers in language and speech processing. *Computational Linguistics* **23**. 269–311.
Oncina, José, Pedro Garcia & Enrique Vidal (1993). Learning subsequential transducers for pattern recognition tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**. 448–458.
Padgett, Jaye (1995). Partial class behavior and nasal place assimilation. In Keiichiro Suzuki & Dirk Elzinga (eds.) *Proceedings of the 1995 Southwestern Workshop on Optimality Theory (SWOT)*. Tucson: Department of Linguistics, University of Arizona. 145–183.
Payne, Amanda (2017). All dissimilation is computationally subsequential. *Lg* **93**. e353–e371. Online only. https://doi.org/10.1353/lan.2017.0076.
Prince, Alan (1983). Relating to the grid. *LI* **14**. 19–100.
Prince, Alan & Paul Smolensky (1993). Optimality theory: constraint interaction in generative grammar. Technical Report 2, Rutgers Center for Cognitive Science.
Rogers, James, Jeffrey Heinz, Gil Bailey, Matt Edlefsen, Molly Visscher, David Wellcome & Sean Wibel (2010). On languages piecewise testable in the strict sense. In Christian Ebert, Gerhard Jäger & Jens Michaelis (eds.) *The mathematics of language*. Berlin: Springer. 255–265.
Rogers, James, Jeffrey Heinz, Margaret Fero, Jeremy Hurst, Dakotah Lambert & Sean Wibel (2013). Cognitive and sub-regular complexity. In Glyn Morrill & Mark-Jan Nederhof (eds.) *Formal grammar: 17th and 18th international conferences*. Number 8035 in Lecture Notes in Computer Science. Berlin: Springer. 90–108.
Rogers, James & Dakotah Lambert (2019). Extracting subregular constraints from regular stringsets. *Journal of Language Modeling* **7**. 143–176.
Sakarovitch, Jacques (2006). *Elements of automata theory*. Cambridge: Cambridge University Press.
Salomaa, Arto (1969). *Theory of automata*. Oxford: Pergamon Press.
Schützenberger, Marcel Paul (1977). Sur une variante des fonctions sequentielles. *Theoretical Computer Science* **4**. 47–57.
Seiler, Hansjakob (1977). *Cahuilla grammar*. Banning, CA: Malki Museum Press.
Selkirk, Elizabeth (1980). The role of prosodic categories in English word stress. *LI* **11**. 563–605.
Stanton, Juliet (2016). Learnability shapes typology: the case of the midpoint pathology. *Lg* **92**. 753–791.
Street, Chester S. & Gregory P. Mollinjin (1981). The phonology of Murinbata. In Bruce E. Waters (ed.) *Australian phonologies: collected papers*. Darwin, NT: Summer Institute of Linguistics. 183–244.
Whitman, William (1947). Descriptive grammar of Ioway-Oto. *IJAL* **13**. 233–248.
Wilson, Colin (2003). Analyzing unbounded spreading with constraints: marks, targets, and derivations. Ms., University of California, Los Angeles.
Wilson, Colin (2006). Unbounded spreading is myopic. Ms., University of California, Los Angeles.