A Computational Analysis of Tone Sandhi Ordering Paradoxes
Chris Oakden (Rutgers University) and Jane Chandlee (Haverford College)

Computational phonology has largely proceeded by analyzing processes in isolation (Kaplan and Kay, 1994; Heinz and Lai, 2013, among others), with more recent work investigating process interaction (Baković and Blumenfeld, 2017; Chandlee et al., 2018). Our more thorough understanding of the range of ways in which processes can co-occur and interact (see Baković, 2007) raises the question of what happens to the computational complexity of a grammar in which rule interaction is both transparent and opaque. This paper addresses that question through a computational analysis of multiple tone sandhi processes in Changting, a Hakka dialect of Chinese. Tone sandhi is a fitting phenomena to study in this regard, given the overlapping nature of triggers and targets both within and across processes. The analysis indicates that despite the challenges Changting tone sandhi (CTS) has posed for both rule- and constraint-based theories of phonology, its computational complexity is still quite limited.

The relevant tone sandhi rules are as follows (Chen, 2004) (H = high, M = mid, L = low, R = rising):

1. M → L / __ R
2. R → H / __ M
3. L → M / __ M
4. M → L / __ L
5. /MRM/ → [LHM]
6. /RMR/ → [HLR]
7. /MLM/ → [MMM]
8. /LML/ → [LLL]

CTS presents a challenge for a rule-based theory of phonology in the form of ordering paradoxes. The mapping in (5) of /MRM/ to [LHM] suggests that rule (1) applies first to derive the intermediate form LRM and then (2) applies to derive [LHM]. But the mapping in (6) suggests the opposite order: /RMR/ to HMR by rule (2) and then HMR to [HLR] by rule (1). Both of these orders correspond to a counterbleeding interaction. Likewise, the mapping in (7) suggests rule (3) is ordered before rule (4): /MLM/ → [MMM], bleeding (4). But the mapping in (8) again indicates the opposite order: /LML/ → [LLL], bleeding (3). As for constraint-based theories, achieving this same set of examples results in ranking paradoxes. For example, in the analysis of Chen (2004), in which candidates are derivations, a constraint like TEMPE (scan the string from left to right) outranking ECON (minimize derivational steps) would correctly choose MRM → LRM → LHM over MRM → MHM, but then the reverse ranking is necessary for MLM → MMM to win out over MLM → LLM → LMM.

The issues CTS presents for both theoretical frameworks may imply that CTS is inherently a complex pattern. Our aim is to assess whether that impression of complexity aligns with a more formal and rigorous notion of computational complexity. In particular, we ask whether CTS can be represented with subregular functions (i.e., proper subsets of the regular relations) as has been argued for a range of segmental phonological processes (see Chandlee and Heinz, 2018, and references therein). The answer is yes, as CTS can be modeled using two of the most restrictive subregular classes of functions, the input strictly local (ISL) and output strictly local (OSL) functions. The analyses presented here are based on the first order logical characterization of ISL discussed in Chandlee and Jardine (2019). Analyses using this logical characterization start by representing strings with graphs, as in Figure 1. A string like MRM is modeled as a set of labeled positions (e.g., position 0 is labeled M, position 1 is labeled R, etc.), and a successor function s that provide an immediate ordering of these positions (e.g., s(0) = 1, s(1) = 2).

Figure 1: Graph representations for the mapping of /MRM/ to [LHM]. Numbers indicate positions that the x variable in the formulas range over, and arrows represent the successor function.

To model a mapping such as /MRM/ → [LHM], logical formulas are defined to indicate how the output correspondent of each input position should be labeled. For example, the interaction of rules (1) and (2) can be modeled with the following set of formulas, in which a formula marked with a prime refers to
output positions and a formula without a prime refers to input positions:

\[
L'(x) \overset{\text{def}}{=} L(x) \lor (M(x) \land R(s(x))) \quad M'(x) \overset{\text{def}}{=} M(x) \land \neg R(s(x))
\]

\[
H'(x) \overset{\text{def}}{=} H(x) \lor (R(x) \land M(s(x))) \quad R'(x) \overset{\text{def}}{=} R(x) \land \neg M(s(x))
\]

In the definitions above, \(L'(x)\) says that an output position should be labeled L if either its input correspondent is labeled L or its input correspondent is labeled M and the successor of its input correspondent is labeled R. This achieves the effect of rule (1), by picking out those positions that are subject to it. \(H'(x)\) states that an output position is labeled H if its input correspondent is labeled H, or is labeled R and its successor is labeled M, achieving the effect of rule (2) in a similar way. Crucially, these definitions refer to input structure only, so the ‘application’ of one rule—i.e. labeling an output position based on input information—does not affect the structural environment of the other rule, nor does it matter which direction the string is evaluated in. In this way these formulas also model the mapping of /RMR/ \(\to [HLR]\) and the opaque interaction between rules (1) and (2).

Importantly, these formulas are all quantifier-free, and logical transductions (like the one above) achieved with only quantifier-free first order logic have been shown to correspond to the class of ISL functions. This is because the determination of what to label each output position can be made based only on information about its input correspondent and the input positions within a fixed window around it (i.e., that information is not global in such a way that would require a quantifier to examine the entire string).

The bleeding interaction of rules (3) and (4) can be represented with the formulas below, which differ from the formulas above in that they also refer to output positions.

\[
L'(x) \overset{\text{def}}{=} (L(x) \land \neg M'(s(x))) \lor (M(x) \land L'(s(x)))
\]

\[
M'(x) \overset{\text{def}}{=} (M(x) \land \neg L'(s(x))) \lor (L(x) \land M'(s(x)))
\]

These definitions model the interaction between the ML and LM rules to produce the maps in (7) and (8). Since they refer to output positions and use the successor function only, the strings are necessarily evaluated from right-to-left. In other words, for this bleeding interaction, direction does matter. Like ISL, OSL functions determine output based on a fixed window; the difference is that this window includes the current position and a bounded number of output positions that follow it. A logical characterization of the combined map of all four rules necessarily incorporates both ISL and OSL properties; we show that this is also possible with a subregular class of functions (namely the input-output strictly local functions).

Thus, modeling the co-occurrence of transparent and opaque rule interactions is possible using subregular classes of functions defined in terms of input and output structure. More intuitively, the reason such an analysis of CTS is possible is because the conditions under which rules (1) and (2) change a particular tone correspond to unique substrings in the input alone, meaning the intermediate forms of a derivational representation are not necessary (and in fact, are problematic as the ordering and ranking paradoxes explained above show). It is also for such rules that direction of application is irrelevant. Direction is, however, relevant in the analysis of rules (3) and (4), precisely because they make reference to output structure. This distinction is unavailable to rule-based and classical OT accounts.

References


