A revised and constrained selectionist learner for reduplication

Eric Raimy, Swarthmore College, Katherine Chinn, Bryn Mawr College

Annual LSA Meeting, 2006, Albuquerque, NM

1.0 Precedence Based Reduplication

- Raimy (2000) proposes that phonological representations are directed graphs
- Reduplication results from a phonological representation having a 'loop' in its precedence structure
- Looping precedence structures are uninterpretable by the phonetics-phonology interface and are linearized which produces repetition of phonological material
- Different reduplicative templates are derived from the nature of the 'loop' in a phonological representation- only material 'within' the loop is repeate
- Non-contiguous reduplication patterns (1bc) require more than a single precedence link to be added

(1) Different Reduplication Patterns

a. One-link

Total reduplication- Indonesian

buku 'book' buku-buku 'books'

→ b → u → k → u → % *linearize* # → b → u → k → u → b → u → k → u → %

b. Two-link

Onset Simplification- Tagalog

trabaho ta-trabaho 'just finished working'

→ t → r → a → b → a → h → o → % *linearize*

→ t → a → t → r → a → b → a → h → o → %

c. Three-link

Sanskrit Perfective Reduplication (Idsardi and Raimy *in prep*)

Root Perfective

skau ku-skau-a > > cu-skāv-a 'tear'

→ s → k → a → u → % *linearize* # → k → u → s → k → a → u → %

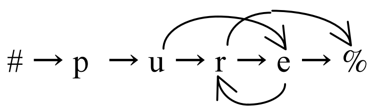
> > cuskāva

d. Three-link

Metathesis- Rotuman

Complete Incomplete

pure puer >> pwer ‘to rule’



linearize # → p → u → e → r → %

>> pwer

- Possible descriptions of added links are determined by *anchor point theory* (APT, Raimy 2005, Fitzgerald to appear) which is part of UG
- Descriptors for ‘beginning’ and ‘end’ of precedence links are based on linear (first, last, next to), segmental (consonant, vowel) and prosodic (onset, stressed) aspects of phonological representations
- Acquisition question is how the learner uses the limited hypothesis space provided by APT to determine what possible descriptions of the links required for a reduplication pattern are
- Iba and Nevins (2004) argue for a *selectionist learner* (Yang 2002) for reduplication:
  –many reduplication patterns are *ambiguous* in that more than a single description of an added link works (e.g. the consonant in a word initial CV sequence is ‘first consonant’, ‘first segment’, ‘before the first vowel’, etc.
  –selectionist aspect of reduplication is supported by dialectical variation in reduplication patterns (Fitzpatrick and Nevins 2004)
- Want to keep selectionist aspect to any model of acquisition for reduplication

2.0 Computational Complexity in Reduplication

2.1 Iba and Nevins (2004) Algorithm

- Iba and Nevins (2004) utilize a ‘brute force’ acquisition algorithm
- Learner is given a base and reduplicated pair:
  –base is scanned to determine all possible anchor points (AP) present in the representation
  –pair of APs from this set are chosen
  –link defined by AP pair is added to base
  –base with added link is linearized
  –linearization output is compared to reduplicated form in training set

–if reduplicated form matches, then AP pair is saved
–repeat until all possible AP pairings are evaluated
–output is a set of working AP pairings

- Computational cost is dependent on the number of APs present in the base form in the training set and how many 'links' are to be considered
- Growth of instantiated APs in a base is dependent on both the number of phonemes and the sequencing of Cs and Vs in the base:
  –'optimal syllables' which alternate CV maximize instantiated APs
  –marked prosodic structures (hiatus, consonant clusters, etc.) reduce instantiated APs
  –there is an upper bound for instantiated APs that is reached when the base reaches a certain number of segments
- (2) shows growth of instantiated AP pairs for alternating CV and identical string (e.g. VVVVV, CCCCCC, etc.) bases
- Only linear (first, last, next to, etc.) and segmental (C vs.. V) information was used to create (2)– addition of prosodic structure (stress and onsets) would increase growth rate and upper bound
- Total computational cost is $O(n/n-1*(n^l-1))$ where n = instantiated AP pairs and l = number of links (this is conservatively $O(n^l)$)

2.2 Chinn and Raimy Algorithm

- Chinn and Raimy algorithm uses string matching to identify repeated region in a given form
- Algorithm uses 3 nested loops to find repetitions:
  –outerloop keeps track of place in string
  –middle loop moves across string to determine if current segment matches the current segment of outerloop
  –innermost loop keeps track of repetitions
- Best case scenario for this algorithm is 'no reduplication' and it will execute in O(n2) because the innermost loop will never execute
- Worst case scenario is that the string is the repetition of a single segment (e.g. /aaaaa/) which causes the algorithm to execute in O(n3)
- Once repetition is determined, the description of the precedence link(s) that must be added can be determined from the repeated string:

–last element of repeated string provides the set 'begin' APs

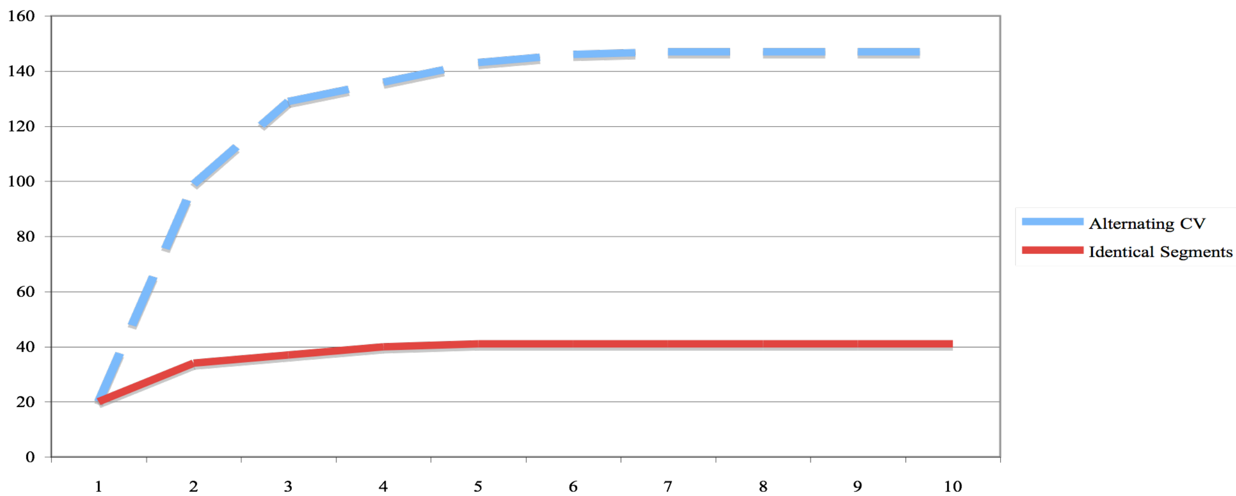
–first element of repeated string provides the set of 'end' APs

–non-contiguous reduplication patterns indicate additional links through comparison to base form

3.0 Comparison

Comments on Chart 2

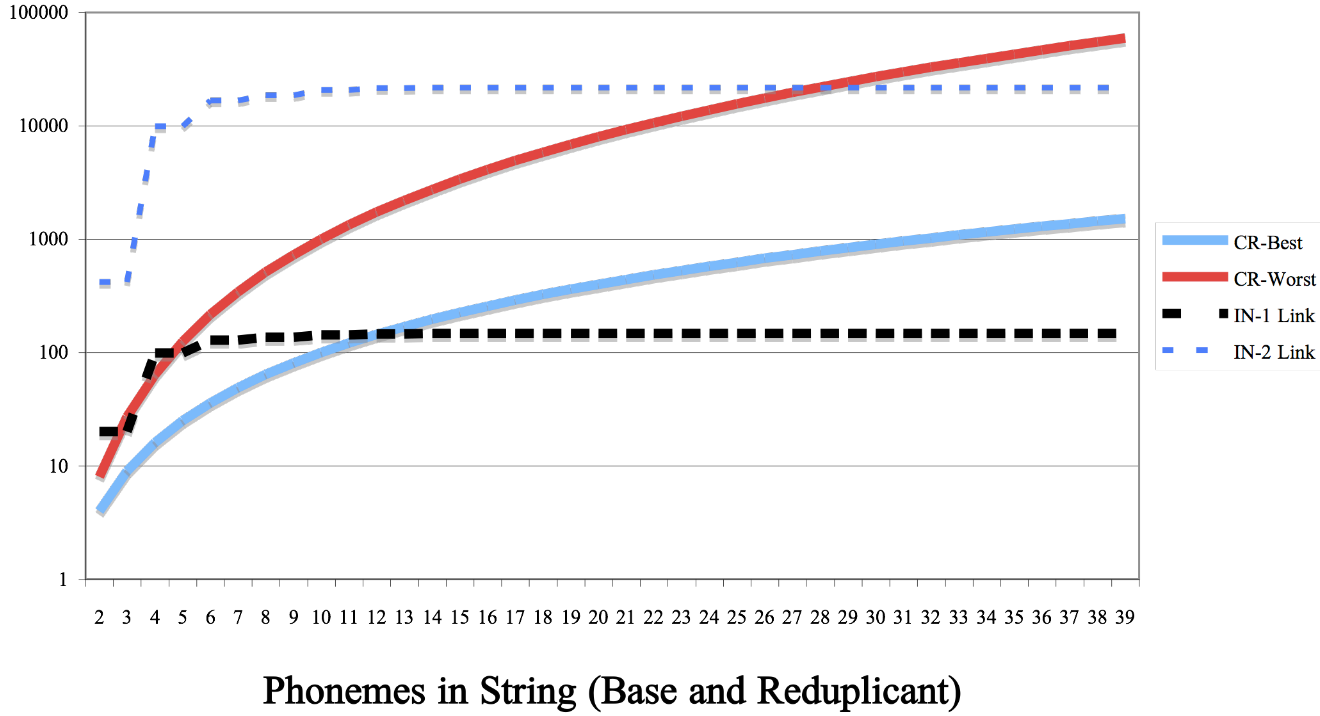
(2) Growth Rate of Instantiated AP Pairs



Number of Segments in Base

- Limit of AP pairings is reached at 7 segments for alternating CV bases- 147
- Limit of AP pairings is reached at 5 segments for identical segment bases- 41

(3) Comparison of Iba Nevin and Chinn Raimy Algorithms



Comments on Chart 3

- Iba Nevins algorithm is better than Chinn Raimy algorithm's best case at 13 segments if only 1 link is considered – Iba Nevins algorithm is empirically inadequate in this version though
- Worst case performance of Chinn Raimy algorithm is better than Iba Nevins 2 link algorithm up to 27 segments in the string
- Best case performance of Chinn Raimy algorithm is better than Iba Nevins 2 link algorithm up to 148 segments in string
- 2 link version of Iba Nevins algorithm may be empirically inadequate (1cd)
- 3 link version of Iba Nevins algorithm performance:
  –1 segment in base : 8420 AP pairs
  –2 segment in base : 980199 AP pairs
- In order to account for all of the data in (1), the Iba Nevins algorithm must consider 3 links

4.0 Conclusion

- (Simplified) Chinn Raimy algorithm will out perform Iba Nevins algorithm as long as the total number of segments in the string is less than instantiated AP pairs given documented reduplication patterns and metathesis

References

Iba, Aaron and Andrew Nevins (2004) A selectionist Learner for Parametrically Ambiguous Reduplicants. Paper presented at the 78th Linguistics Society of America, Boston MA.

Fitzpatrick, Justin (to appear) A theory of possible affix types. In Salanova, A. (ed) *Papers from EVELIN 1*, MITWPL.

Fitzpatrick, Justin and Andrew Nevins (2004) Linearizing nested and overlapping precedence in multiple reduplication. *Proceedings from the 27th Penn Linguistics Colloquium*. pp. 77-85.

Idsardi, William and Eric Raimy (in prep) Insights into reduplication from Sanskrit. Ms. University of Maryland and Swarthmore College.

Raimy, Eric (2000) *The phonology and morphology of reduplication*. Mouton de Gruyter.

Raimy, Eric (2005) Prosodic residue in an atemplatic world. University of Delaware Linguistics Colloquium Series.

Yang, Charles (2002) *Knowledge and learning in natural language*. Oxford University Press.

Acknowledgements

Katherine Chinn's work on this project was supported by a Summer Research Grant provided by Bryn Mawr College. We would like to thank Thomas Hunter (help with math), Rich Wicentowski (help with coding the algorithm), Bill Idsardi (help with comparisons). All mistakes of fact or fiction are our own responsibility.

Contact Info

eraimy1@swarthmore.edu