Objectives & Introduction

Goals: Want to compute template morphology such that...
- Model it as productive process
- Understand its computational properties
- Limit its expressive power

Contribution: Use multi-tape finite-state transducers with local subclasses

Templatic morphology

Most languages use concatenative morphology
- English progressive: hold → holding

Semitic has templatic morphology (McCarthy, 1981)

Active verbs

- kftb 'it was written'
- Root consonants C: ktb
- Template T: CV.CVC

Passive verbs

- Inflectional vowels V: a
- Root consonants C: ktb
- Template T: CV.CVC

Locality in Natural language

Most natural language processes are local
- uses bounded finite window (Chandice, 2014)
- computed with N-grams

Our question:
- Are Semitic templates local?

Computing morphology

- hold → holding
- Easy to compute
- Single-tape FST (IT-FST)

- kftb 'it was written'
- How to compute?
- IT-FST aren't for non-linearity
- State-explosion -> large finite language
- Modifications reflect state-explosion

Computing locality

- kftb 'it was written'
- seems non-local because of discontinuity
- Unknown locality

Locality across template types

Matching

- 1-1 Matching: ktb ui CV.CVC
- Input: ktb Output: CV.CVC
- Power: MT-1-ISL

Final spread

- ktb a CV.CVC
- Input: ktb Output: CV.CVC
-受限: MT-2-ISL

Gemination

- ktb ui CV.CVC
- Input: ktb Output: CV.CVC
- MT-2-ISL

Pre-association

- ktb a CV.CVC
- Input: ktb Output: CV.CVC
- MT-1-ISL

Partial copying

- btd a CV.CVC
- Input: btd Output: CV.CVC
- 2-way

Total copying

- $la$ CV.CVC
- Input: $la$ Output: CV.CVC
- 2-way

Concatenation

- $ui$ in V-1V-CVC.CVC
- Input: $ui$ Output: V-1V-CVC.CVC
- MT-3-ISL

Semitic templates are

- Typologically rare
- Theoretically cool
- Computationally local ... with the right representation = MT-FSTs

References


Computing locality

Concatenation (=suffusion) is computationally weak = k Input Strictly Local subclass
- keep track of ONLY last k segments in input

Suffusion is 1-ISL.
- 1-ISL states keep track of last k-1 (=0) seen input (empty string)

Template-filling (1-1 match) is 1-ISL over multiple tapes
- Change is based on current input symbol on two tapes

What does an Arabic MT-FST look like?

Template-filling is local over multiple tapes

Locality in template-filling depends on # of vowels, consonants, and slots

Computing locality

Concatenation (=suffusion) is computationally weak = k Input Strictly Local subclass
- keep track of ONLY last k segments in input

Suffusion is 1-ISL.
- 1-ISL states keep track of last k-1 (=0) seen input (empty string)

Template-filling (1-1 match) is 1-ISL over multiple tapes
- Change is based on current input symbol on two tapes

Discussion

Finiteness

- Arabic has at most 2 syllable templates
- Could model Arabic with just large finite language-language mapping
- But finiteness costs you generalizations (Savitch, 1993)

Phonological emergence

- MT-FST took template T as morphological input
- Contemporary theories argue there is NO template T, it’s phonologically emergent
- But the phonology still needs to create a template (= MT-FST)

Expressivity

- Expressivity of MT-FST depends on how you encode your input
- With the right encoding, majority rules is MT-ISL
- Problem is curtailed by letting morphology dictate your encoding

Conclusions