Subregular Learning of a Phonology and a Set of Underlying Forms

Wenyue Hua, Adam Jardine, and Huteng Dai

Rutgers University

Introduction

UR and Phonological grammar learning

- **Learning problem:** the simultaneous inference of underlying representations (URs) and a phonological grammar from alternating surface representations (SRs)

  (Merchant, 2008; Tesar, 2014; Cotterell et al., 2015)

- **Our proposal:** a solution based on the structure provided by the input strictly local (ISL) functions

  (Chandlee and Heinz, 2018; Jardine et al., 2014)

English plural

The plural morpheme in English has at least two pronunciations: [s] and [z].

Morphemes	SR
CAT-PL	[kæts]
CUFF-PL	[kʌfs]
DEATH-PL	[dɛθs]
GIRL-PL	[gɜːrlz]
CHAIR-PL	[t͡ʃɛərz]
BOY-PL	[bɔɪz]
...	...

English plural

The plural morpheme in English has at least two pronunciations: [s] and [z].

Analysis:

- A map from morphemes to URs

  CAT→/kæt/

  PL→/z/

  etc.
- A map from URs to SRs

  /z/ → [s] /[-SONORANT] ___

Morphemes	SR
CAT-PL	[kæts]
CUFF-PL	[kʌfs]
DEATH-PL	[dɛθs]
GIRL-PL	[gɜ:rlz]
CHAIR-PL	[t͡ʃɛərz]
BOY-PL	[bɔɪz]
...	...

Formalizing the problem

- M: set of morphemes
- Σ: alphabet of symbols in SR and UR (segmental inventory)
- Targets:
    - lexicon function $\ell : M^* \to \Sigma^*$
    - phonology function $\varphi : \Sigma^* \to \Sigma^*$

Formalizing the problem

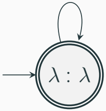
- M: set of morphemes
- Σ: alphabet of symbols in SR and UR (segmental inventory)
- Targets:
  - lexicon function $\ell : M^* \to \Sigma^*$
  - phonology function $\varphi : \Sigma^* \to \Sigma^*$
- Learning data is generated by $\varphi \circ \ell$; i.e., a finite set D such that

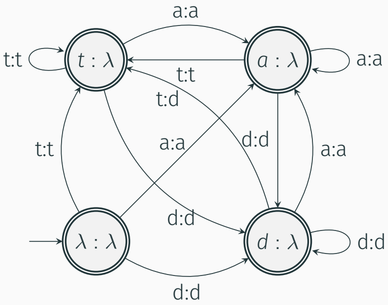
$$\forall \langle m, s \rangle \in D, \quad \varphi(\ell(m)) = s$$

- Note: ℓ is an ISL_1 function from morphemes to URs
- ISL functions only make changes in the output with respect to the local information in the input.

Target transducers and learning data

r_1 : /tat/ s_1 : /ta/
r_2 : /tad/ s_2 : /da/
r_3 : /a/ s_3 : /da/

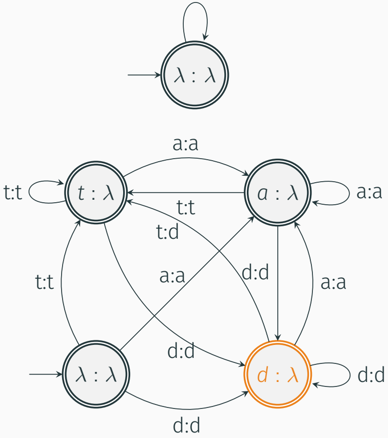




m	s
$r_1 s_1$	[tatta]
$r_1 s_2$	[tatda]
$r_1 s_3$	[tata]
$r_2 s_1$	[tadda]
$r_2 s_2$	[tadda]
$r_2 s_3$	[tada]
$r_3 s_1$	[ata]
$r_3 s_2$	[ada]
$r_3 s_3$	[aa]

Target transducers and learning data

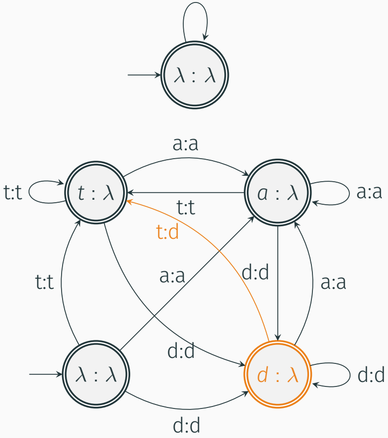
r_1 : /tat/ s_1 : /ta/
r_2 : /tad/ s_2 : /da/
r_3 : /a/ s_3 : /da/



m	s
r_1s_1	[tatta]
r_1s_2	[tatda]
r_1s_3	[tata]
r_2s_1	[tadda]
r_2s_2	[tadda]
r_2s_3	[tada]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]

Target transducers and learning data

r_1 : /tat/ s_1 : /ta/
r_2 : /tad/ s_2 : /da/
r_3 : /a/ s_3 : /da/



m	s
r_1s_1	[tatta]
r_1s_2	[tatda]
r_1s_3	[tata]
r_2s_1	[tadda]
r_2s_2	[tadda]
r_2s_3	[tada]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]

Assumptions

The learner knows *a priori*:

- A lexicon of morphemes/lexical meanings (M), in which each morpheme has only one UR (ℓ is ISL_1).
- The ISL_k structure of phonology function. Here, we focus on ISL_2

Proposed Learner

Roadmap

- Initial hypothesis
- Learning procedure

Initial hypothesis

- Initial hypothesis for ℓ: prefix tree transducer T_ℓ, obtained from SR segmentation based on longest common prefix (LCP)

(Oncina et al., 1993; Jardine et al., 2014)

Single Process Example: Initial lexicon transducer

m	s
$r_1 s_1$	[tatta]
$r_1 s_2$	[tatda]
$r_1 s_3$	[tata]
$r_2 s_1$	[tadda]
$r_2 s_2$	[tadda]
$r_2 s_3$	[tada]
$r_3 s_1$	[ata]
$r_3 s_2$	[ada]
$r_3 s_3$	[aa]

LCP of SR is tat

Single Process Example: Initial lexicon transducer

m	s
$r_1 s_1$	[tatta]
$r_1 s_2$	[tatda]
$r_1 s_3$	[tata]
$r_2 s_1$	[tadda]
$r_2 s_2$	[tadda]
$r_2 s_3$	[tada]
$r_3 s_1$	[ata]
$r_3 s_2$	[ada]
$r_3 s_3$	[aa]

LCP of SR is tad

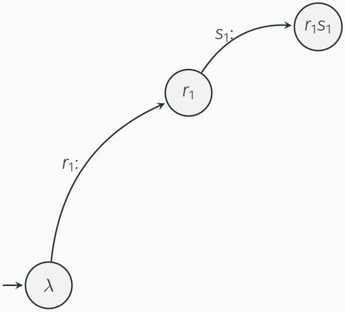
Single Process Example: Initial lexicon transducer

m	s
$r_1 s_1$	[tatta]
$r_1 s_2$	[tatda]
$r_1 s_3$	[tata]
$r_2 s_1$	[tadda]
$r_2 s_2$	[tadda]
$r_2 s_3$	[tada]
$r_3 s_1$	[ata]
$r_3 s_2$	[ada]
$r_3 s_3$	[aa]

LCP of SR is a

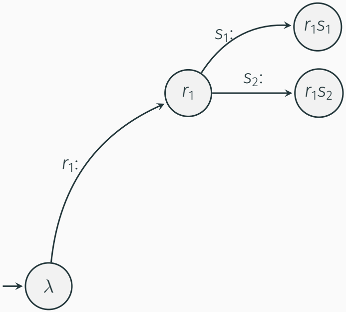
Single Process Example: Initial lexicon transducer

m	s
r_1s_1	[tatta]
r_1s_2	[tatda]
r_1s_3	[tata]
r_2s_1	[tadda]
r_2s_2	[tadda]
r_2s_3	[tada]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]



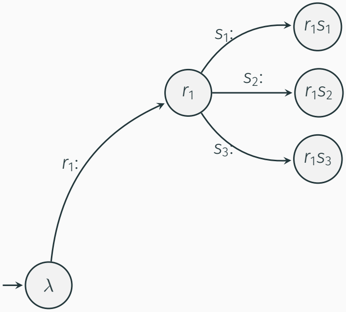
Single Process Example: Initial lexicon transducer

m	s
r_1s_1	[tatta]
r_1s_2	[tatda]
r_1s_3	[tata]
r_2s_1	[tadda]
r_2s_2	[tadda]
r_2s_3	[tada]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]



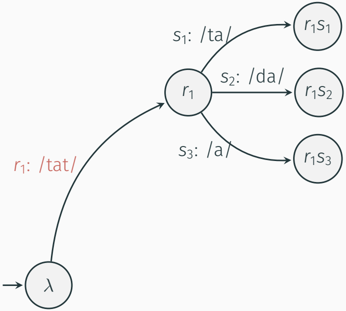
Single Process Example: Initial lexicon transducer

m	s
r_1s_1	[tatta]
r_1s_2	[tatda]
r_1s_3	[tata]
r_2s_1	[tadda]
r_2s_2	[tadda]
r_2s_3	[tada]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]



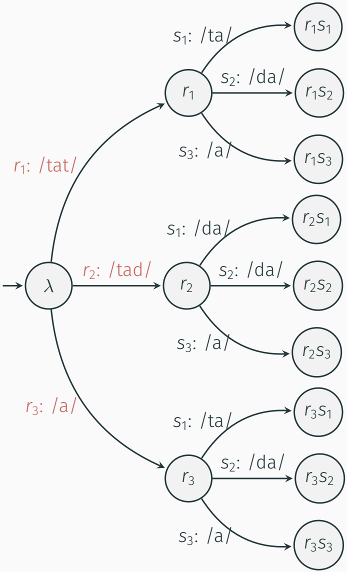
Single Process Example: Initial lexicon transducer

m	s
r_1s_1	[tatta]
r_1s_2	[tatda]
r_1s_3	[tata]
r_2s_1	[tadda]
r_2s_2	[tadda]
r_2s_3	[tada]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]



Single Process Example: Initial lexicon transducer

m	s
r_1s_1	[tatta]
r_1s_2	[tatda]
r_1s_3	[tata]
r_2s_1	[tadda]
r_2s_2	[tadda]
r_2s_3	[tada]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]



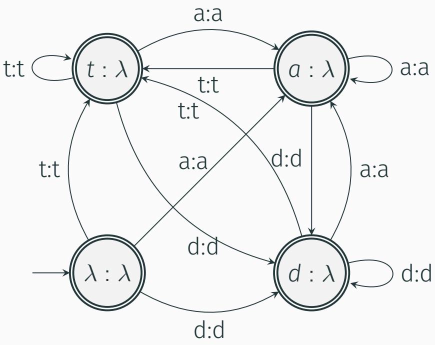
Initial hypothesis

- Initial hypothesis for ℓ: prefix tree transducer T_ℓ, obtained from SR segmentation based on longest common prefix

(Oncina et al., 1993; Jardine et al., 2014)

- Initial hypothesis for φ: ISL_2 transducer T_φ for identity function

Single Process Example: phonology transducer



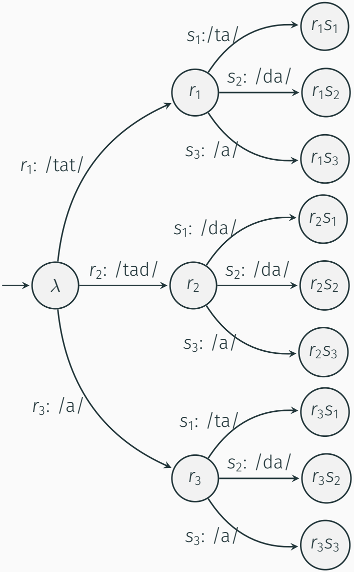
Identity function:
/tadt/→[tadt]
/tatt/→[tatt]
...

Learning Procedure

- Modify lexicon transducer T_ℓ until one UR per morpheme
- For each change in lexicon transducer T_ℓ, make *opposite* change in phonology transducer T_φ

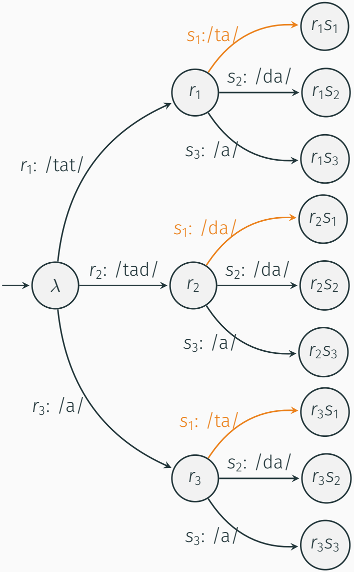
Inconsistency detection

The learner detects the inconsistency on lexicon transducer T_ℓ



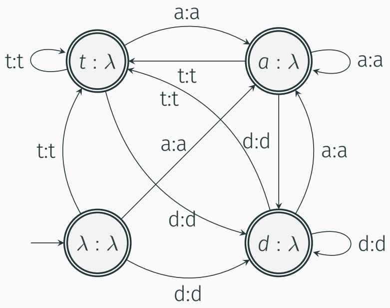
Inconsistency detection

The learner detects the inconsistency on lexicon transducer T_ℓ



Environment Collection

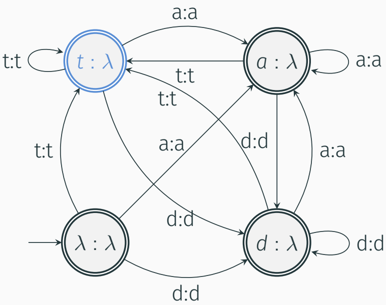
Find environment for different SRs based on the phonology transducer.



environment	s_1

Environment Collection

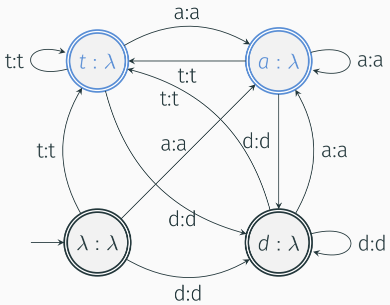
Find environment for different SRs based on the phonology transducer.



	environment	s_1
$r_1 s_1$	tat	ta

Environment Collection

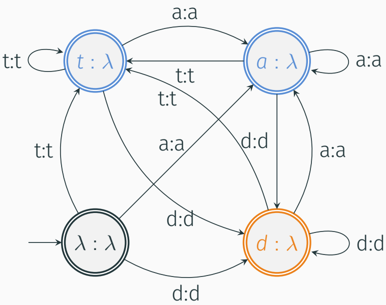
Find environment for different SRs based on the phonology transducer.



	environment	s_1
r_1s_1	tat	ta
r_3s_1	a	ta

Environment Collection

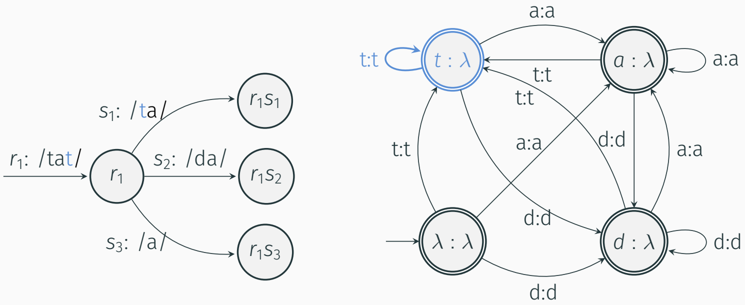
Find environment for different SRs based on the phonology transducer.



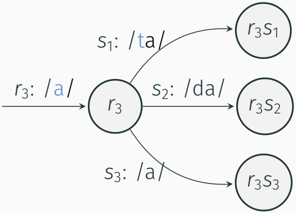
	environment	s_1
r_1s_1	tat	ta
r_3s_1	a	ta
r_2s_1	tad	da

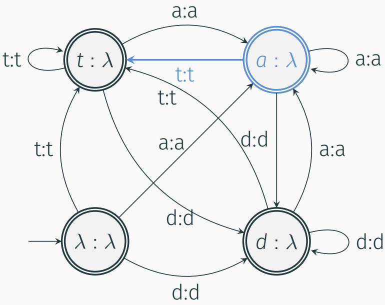
Modifying Transducers

Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.



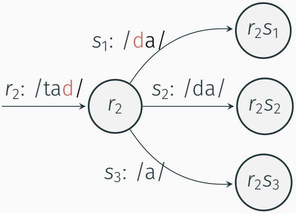
Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.

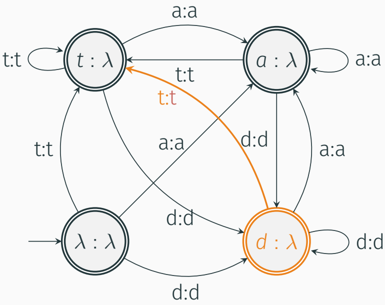




Modifying Transducers

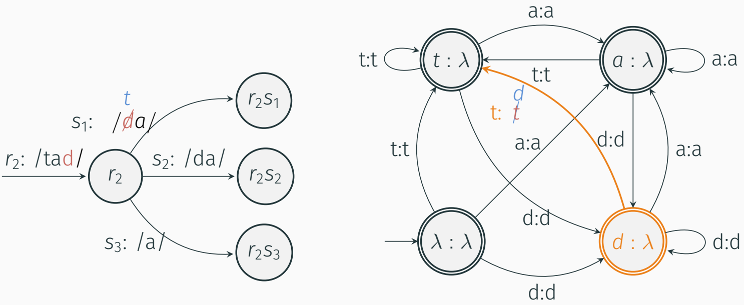
Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.





Modifying Transducers

Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.



Primary Result

- The learner can learn assimilation, dissimilation, deletion, epenthesis and metathesis.
- So far it learns only one phonology process from a data set.
- In particular, it is able to learn opacity, *i.e.* self-counter-feeding and self-counter-bleeding.

Future Research

- Learn multiple phonology processes simultaneously from one data set.
- Learn all ISL_2 functions.
- Learn all ISL_k functions for any given k.
- Learn Output-Strictly-Local (OSL) phonology transformations.

  (Chandlee et al., 2015).
- ...

Discussion

Three questions

- Why we design an ISL learner, *i.e.* why not OSL learner or Output Tier Based Strictly Local (OTSL) learner?

  (Chandlee et al., 2015; Burness and McMullin, 2019)

- How abstract is the learnt UR? Specifically, is it able to learn abstract URs?

  (Kiparsky, 1968; Kenstowicz and Kisseberth, 2014)

- What differentiate subregular learners from other learners?

Why ISL phonology?

- Empirically significant: 94% of phonology patterns in P-Base database are ISL.

  (Mielke, 2004; Chandlee and Heinz, 2018).
- Learning based on the structure of ISL class can be extended to OSL and OTSL functions, which both share a particular structure

Can we learn abstract URs?

- It depends.

Can we learn abstract URs?

- It depends.
- Two cases: whether the abstract UR exerts phonological influence on the target phonology function.

Subregular learner *vs.* Other learners

- Current learner: specific to learning phonology.

  (Gallistel and King, 2011; Heinz, 2010).

- Cue-based parameter setting model and Learners based on Optimality Theory: non-specific to learning phonology

  (Dresher and Kaye, 1990; Dresher, 1999; Jarosz, 2006; Apoussidou, 2007; Merchant, 2008; Merchant and Tesar, 2008).

Subregular learner *vs.* Other learners

- Current learner: **specific to learning phonology**.

  (Gallistel and King, 2011; Heinz, 2010).

- Cue-based parameter setting model and Learners based on Optimality Theory: **non-specific to learning phonology**

  (Dresher and Kaye, 1990; Dresher, 1999; Jarosz, 2006; Apoussidou, 2007; Merchant, 2008; Merchant and Tesar, 2008).

- Modular learning: an ISL learner is an independent module in learning the whole phonology. By composing different modules, the knowledge of the whole is acquired.

  Heinz (2010, 2011).

Conclusion

Summary

- Subregular classes of functions provide structure for the simultaneous induction of URs and the phonological grammar
- The general procedure here can be extended to learning iterative (output-based) processes, long-distance processes, and process interactions

Acknowledgements

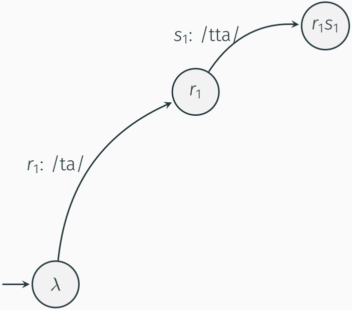
We thank Bruce Tesar, Adam McCollumn, Mariapaola D'imperio, Colin Wilson, Eric Baković, Chris Oakden, Dine Mamadou, Jill Harper, Sreekar Raghotham, the Rutgers MathLing group, and the audience at the Rutgers/SBU/Haverford/Delaware subregular phonology workshop, for their insightful comments.

Backup slides

Regressive Assimilation

/t/ becomes voiced [d] before /d/.

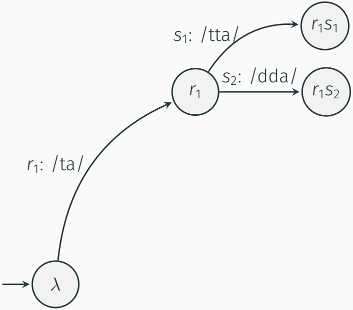
m	s
$r_1 s_1$	[tatta]
$r_1 s_2$	[tadda]
$r_1 s_3$	[tata]
$r_2 s_1$	[tadta]
$r_2 s_2$	[tadda]
$r_2 s_3$	[tata]
$r_3 s_1$	[ata]
$r_3 s_2$	[ada]
$r_3 s_3$	[aa]



Regressive Assimilation

/t/ becomes voiced [d] before /d/.

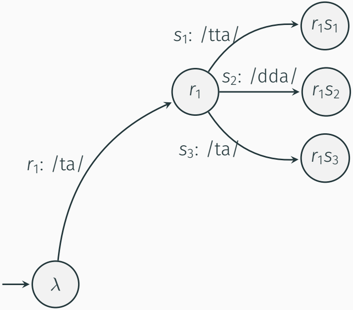
m	s
r_1s_1	[tatta]
r_1s_2	[tadda]
r_1s_3	[tata]
r_2s_1	[tadta]
r_2s_2	[tadda]
r_2s_3	[tata]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]



Regressive Assimilation

/t/ becomes voiced [d] before /d/.

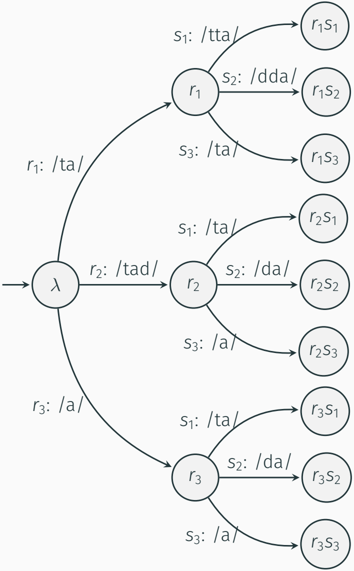
m	s
r_1s_1	[tatta]
r_1s_2	[tadda]
r_1s_3	[tata]
r_2s_1	[tadta]
r_2s_2	[tadda]
r_2s_3	[tata]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]



Regressive Assimilation

/t/ becomes voiced [d] before /d/.

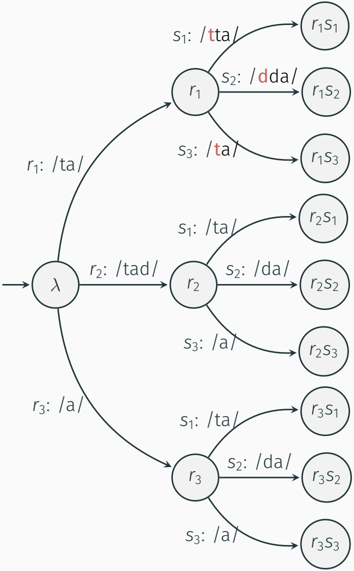
m	s
r_1s_1	[tatta]
r_1s_2	[tadda]
r_1s_3	[tata]
r_2s_1	[tadta]
r_2s_2	[tadda]
r_2s_3	[tata]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]



Regressive Assimilation

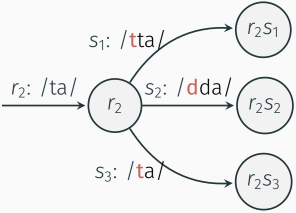
/t/ becomes voiced [d] before /d/.

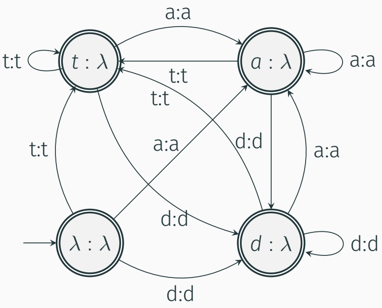
m	s
$r_1 s_1$	[tatta]
$r_1 s_2$	[tadda]
$r_1 s_3$	[tata]
$r_2 s_1$	[tadta]
$r_2 s_2$	[tadda]
$r_2 s_3$	[tata]
$r_3 s_1$	[ata]
$r_3 s_2$	[ada]
$r_3 s_3$	[aa]



Modifying Transducers

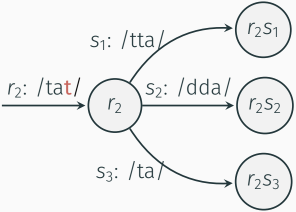
Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.

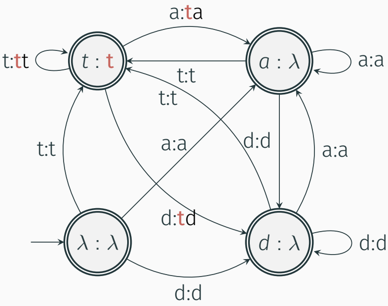




Modifying Transducers

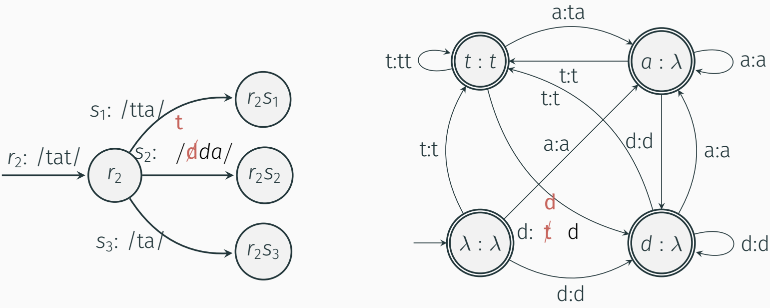
Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.





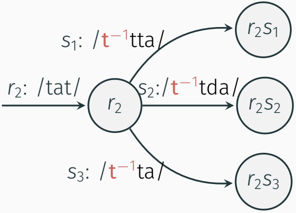
Modifying Transducers

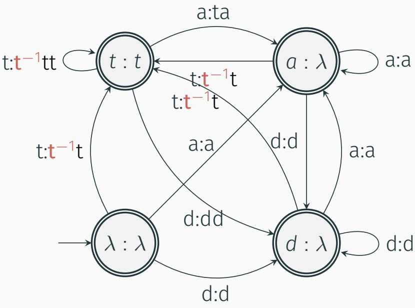
Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.



Modifying Transducers

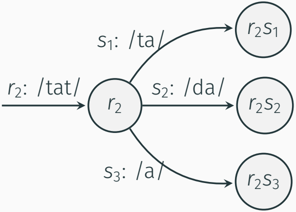
Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.

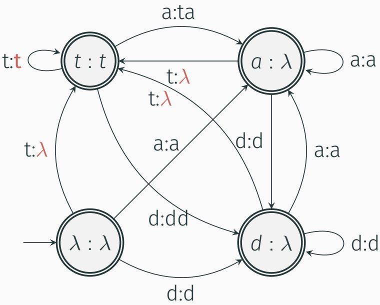




Modifying Transducers

Change SR into UR in the prefix-tree transducer.
Make an opposite change in the phonology transducer.





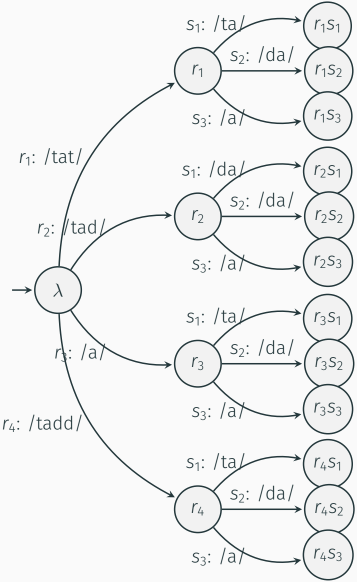
Opacity

- counter-feeding
- counter-bleeding

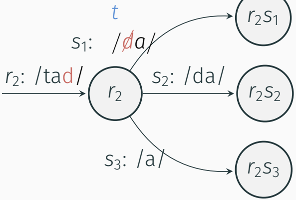
Counter-feeding

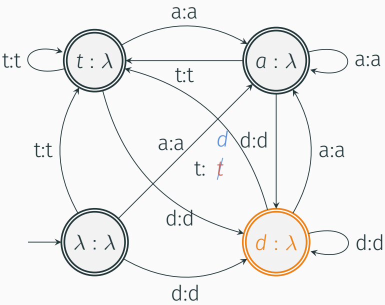
/t/ becomes a voiced [d] after /d/.

m	s
r_1s_1	[tatta]
r_1s_2	[tatda]
r_1s_3	[tata]
r_2s_1	[tadda]
r_2s_2	[tadda]
r_2s_3	[tada]
r_3s_1	[ata]
r_3s_2	[ada]
r_3s_3	[aa]
r_4s_1	[taddta]
r_4s_2	[taddda]
r_4s_3	[tadda]

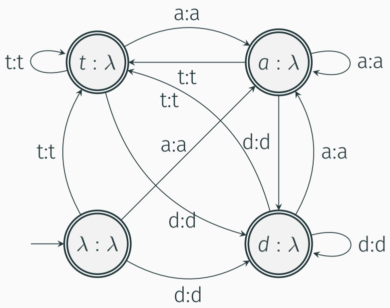


Modified Transducer



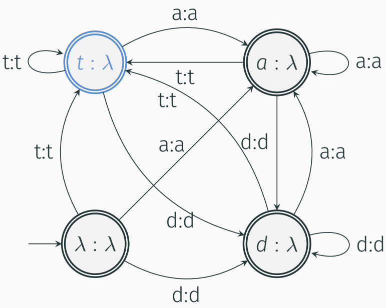


What about the environment?



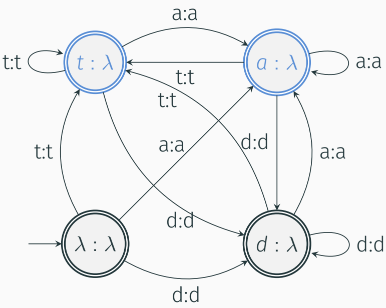
environment	s_1

What about the environment?



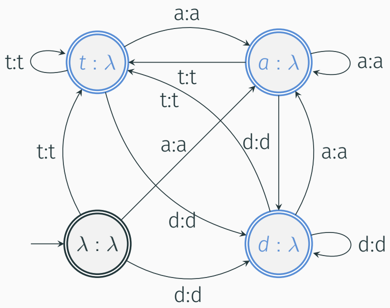
	environment	s_1
$r_1 s_1$	tat	ta

What about the environment?



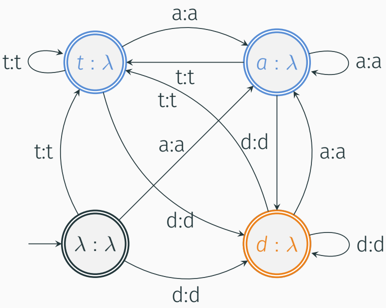
	environment	s_1
r_1s_1	tat	ta
r_3s_1	a	ta

What about the environment?



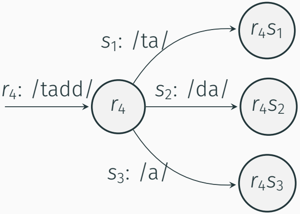
	environment	s_1
$r_1 s_1$	tat	ta
$r_3 s_1$	a	ta
$r_4 s_1$	tadd	ta

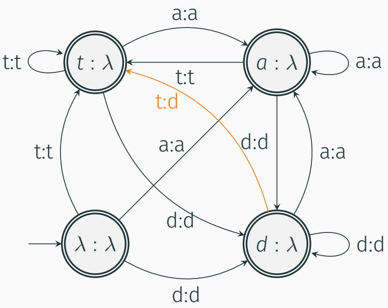
What about the environment?



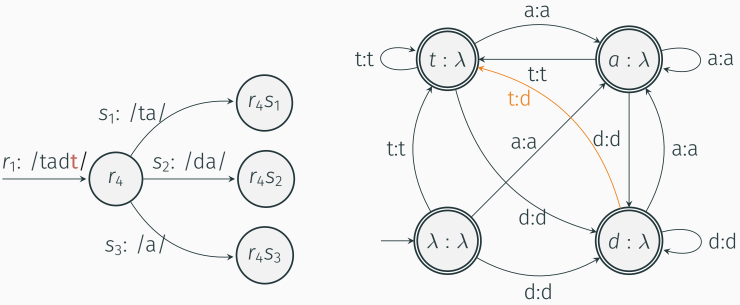
	environment	s_1
$r_1 s_1$	tat	ta
$r_3 s_1$	a	ta
$r_4 s_1$	tadd	ta
$r_2 s_1$	tad	da

What should we do about the clash of the environment?

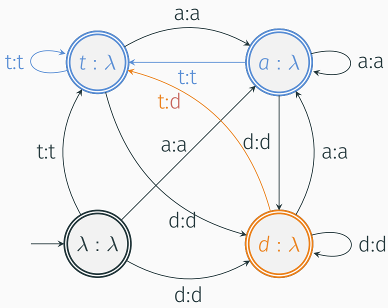




What should we do about the clash of the environment?



What should we do about the clash of the environment?



	environment	s_1
r_1s_1	tat	ta
r_3s_1	a	ta
r_4s_1	tadt	ta
r_2s_1	tad	da

References

Apoussidou, D. (2007). The learnability of metrical phonology: Lot. *Universiteit van Amsterdam [Host]*.

Burness, P. and McMullin, K. (2019). Efficient learning of output tier-based strictly 2-local functions. In *Proceedings of the 16th Meeting on the Mathematics of Language*, pages 78–90, Toronto, Canada. Association for Computational Linguistics.

Chandlee, J., Eyraud, R., and Heinz, J. (2015). Output strictly local functions. In *Proceedings of the 14th Meeting on the Mathematics of Language (MoL 2015)*, pages 112–125, Chicago, USA. Association for Computational Linguistics.

Chandlee, J. and Heinz, J. (2018). Strictly locality and phonological maps. *LI*, 49:23–60.

Cotterell, R., Peng, N., and Eisner, J. (2015). Modeling word forms using latent underlying morphs and phonology. *Transactions of the Association for Computational Linguistics*, 3:433–447.

Dresher, B. E. (1999). Charting the learning path: Cues to parameter setting. *Linguistic Inquiry*, 30(1):27–67.

Dresher, B. E. and Kaye, J. D. (1990). A computational learning model for metrical phonology. *Cognition*, 34(2):137–195.

Gallistel, C. R. and King, A. P. (2011). *Memory and the computational brain: Why cognitive science will transform neuroscience*, volume 6. John Wiley & Sons.

Heinz, J. (2010). Learning long-distance phonotactics. *Linguistic Inquiry*, 41(4):623–661.

Heinz, J. (2011). Computational phonology part II: Grammars, learning, and the future. *Language and Linguistics Compass*, 5(4):153–168.

Jardine, A., Chandlee, J., Eyraud, R., and Heinz, J. (2014). Very efficient learning of structured classes of subsequential functions from positive data. In *International Conference on Grammatical Inference*, pages 94–108.

Jarosz, G. (2006). *Rich lexicons and restrictive grammars: Maximum likelihood learning in Optimality Theory*. PhD thesis, John Hopkins University.

Kenstowicz, M. and Kisseberth, C. (2014). *Topics in phonological theory*. Elsevier.

Kiparsky, P. (1968). *How abstract is phonology?* Indiana University Linguistics Club.

Merchant, N. and Tesar, B. (2008). Learning underlying forms by searching restricted lexical subspaces. In *In The Proceedings of CLS 41. ROA-811*. Citeseer.

Merchant, N. N. (2008). *Discovering underlying forms: Contrast pairs and ranking*. PhD thesis, Rutgers University-Graduate School-New Brunswick.

Mielke, J. (2004). P-base: Database of sound patterns.

Oncina, J., García, P., and Vidal, E. (1993). Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):448–458.

Tesar, B. (2014). *Output-driven phonology: Theory and learning*. Cambridge University Press.