Learning Phonotactic Grammars from Surface Forms:

Phonotactic Patterns are Neighborhood-distinct

Jeff Heinz

University of California, Los Angeles

April 28, 2006

Introduction

- I will present an unsupervised batch learning algorithm for phonotactic grammars without a priori Optimality-theoretic (OT) constraints (Prince and Smolensky 1993, 2004).
- The premise: linguistic patterns (such as phonotactic patterns) have properties which reflect properties of the learner.
- In particular, the learner leads to a novel, nontrivial hypothesis: all phonotactic patterns are neighborhood-distinct (to be defined momentarily).

Learning in phonology

Learning in Optimality Theory (Tesar 1995, Boersma 1997, Tesar 1998, Tesar and Smolensky 1998, Hayes 1999, Boersma and Hayes 2001, Lin 2002, Pater and Tessier 2003, Pater 2004, Prince and Tesar 2004, Hayes 2004, Riggle 2004, Alderete et al. 2005, Merchant and Tesar to appear, Wilson 2006, Riggle 2006)

Learning in Principles and Parameters (Wexler and Culicover 1980, Dresher and Kaye 1990)

Learning Phonological Rules (Gildea and Jurafsky 1996, Albright and Hayes 2002, 2003)

Learning Phonotactics (Ellison 1994, Frisch 1996, Coleman and Pierrehumbert 1997, Frisch et al. 2004, Albright 2006, Goldsmith 2006, Hayes and Wilson 2006)

Overview

1. Representations of Phonotactic Grammars
2. ATR Harmony Language
3. The Learner
4. Other Results
5. The Neighborhood-distinctness Hypothesis
6. Conclusions

Finite state machines as phonotactic grammars

- They accept or reject words. So it meets the minimum requirement for a phonotactic grammar– a device that at least answers Yes or No when asked if some word is possible (Chomsky and Halle 1968, Halle 1978).

- They can be related to finite state OT models, which allow us to compute a phonotactic finite state acceptor (Riggle 2004), which becomes the target grammar for the learner.

- The grammars are well-defined and can be manipulated (Hopcroft et al. 2001). (See also Johnson (1972), Kaplan and Kay (1981, 1994), Ellison (1994), Eisner (1997), Albro (1998, 2005), Karttunen (1998), Riggle (2004) for finite-state approaches to phonology.)

The ATR harmony language

- ATR Harmony Language (e.g. Kalenjin (Tucker 1964, Lodge 1995). See also Baković (2000) and references therein).
- To simplify matters, assume:
  1. It is CV(C) (word-initial V optional).
  2. It has ten vowels.
     - {i,u,e,o,a} are [+ATR]
     - {I,U,E,O,A} are [-ATR]
  3. It has 8 consonants {p,b,t,d,k,g,m,n}
  4. Vowels are [+syllabic] and consonants are [-syllabic] and have no value for [ATR].

The ATR harmony language target grammar

- There are two constraints:
  1. The syllable structure phonotactic
     - CV(C) syllables (word-initial V OK).
  2. ATR harmony phonotactic
     - All vowels in word must agree in [ATR].

The ATR harmony language

- Vowels in each word agree in [ATR].

1.	a	7.	bedko	13.	I	20.	Ak
2.	ka	8.	piptapu	14.	kO	21.	kOn
3.	puki	9.	mitku	15.	pAkI	22.	pAtkI
4.	kitepo	10.	etiptup	16.	kUtEpA	23.	kUptEpA
5.	pati	11.	ikop	17.	pOtO	24.	pOtkO
6.	atapi	12.	eko	18.	AtEtA	25.	AtEptAp
				19.	IkUp	26.	IkU

Question

Q: How can a finite state acceptor be learned from a finite list of words like *badupi*,*bakta*,...?

A:
– Generalize by writing smaller and smaller descriptions of the observed forms
– guided by the notion of natural class and a structural notion of locality (the neighborhood)

The input with natural classes

- Partition the segmental inventory by natural class and construct a prefix tree.
- Examples:
  - Partition 1: | i,u,e,o,a,I,U,E,O,A | p,b,t,d,k,g,m,n |

    [+syl] and [-syl] divide the inventory into two non-overlapping groups.
  - Partition 2: | i,u,e,o,a | I,U,E,O,A | p,b,t,d,k,g,m,n |

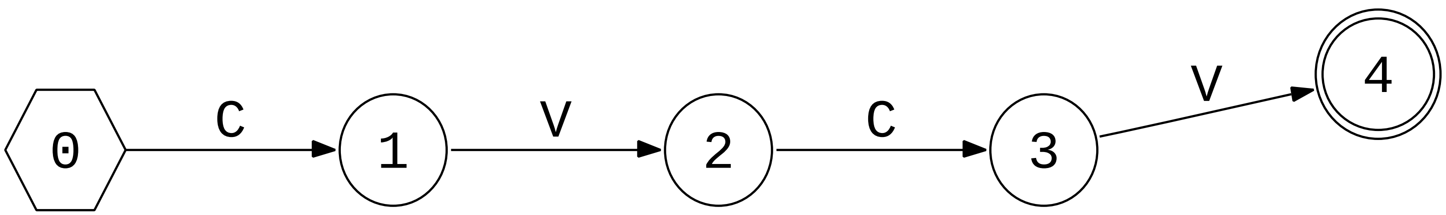
    [+syl,-ATR], [+syl,+ATR] and [-syl] divide the inventory into three non-overlapping groups.
- Thus, [bikta] is read as [CVCCV] by Partition 1.

Prefix tree construction

- A prefix is tree is built one word at a time.
- Follow an existing path in the machine as far as possible.
- When no path exists, a new one is formed.

Building the prefix tree

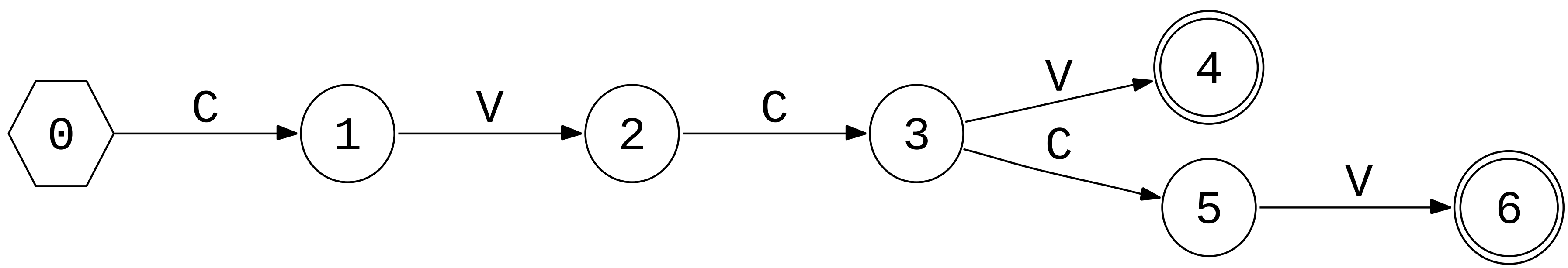
using the [+syl] | [-syl] partition



- Words processed: piku

Building the prefix tree

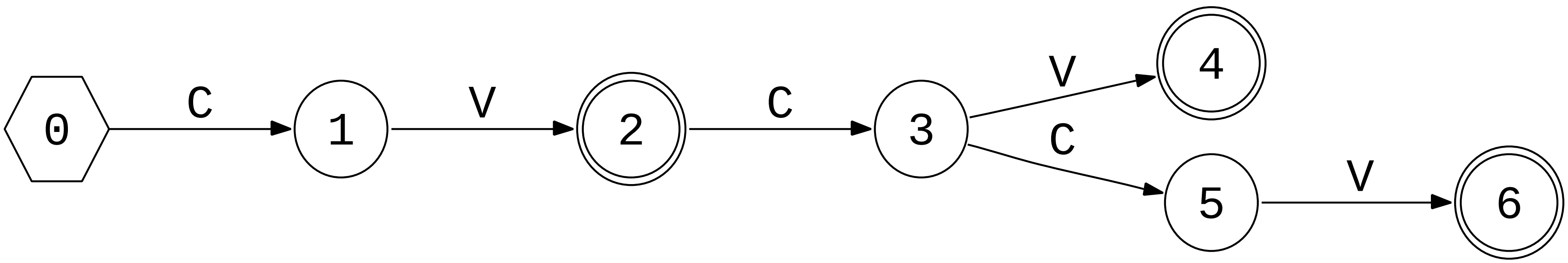
using the [+syl] | [-syl] partition



- Words processed: piku, bItkA

Building the prefix tree

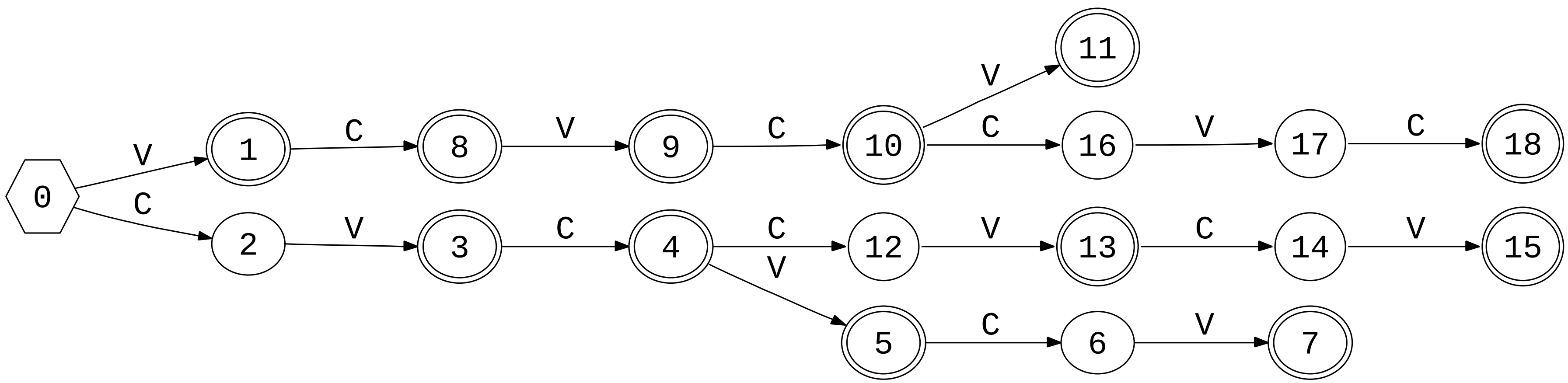
using the [+syl] | [-syl] partition



- Words processed: piku, bItkA, mA

The prefix tree for the ATR harmony language

using the [+syl] | [-syl] partition



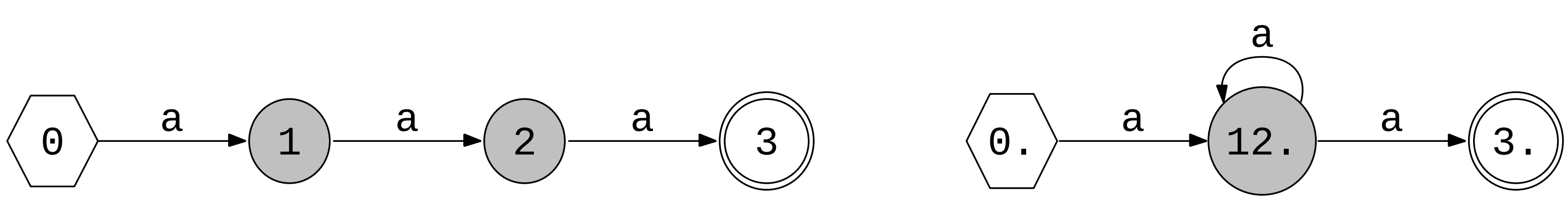
- A structured representation of the input.

Further generalization?

- The learner has made some generalizations by structuring the input with the [syl] partition– e.g. the current grammar can accept any CVCV word.
- However, the current grammar undergeneralizes:
  it cannot accept words of four syllables like CVCVCVCVCV.
- And it overgeneralizes:
  it can accept a word like *bitE*.

State merging

- Correct the undergeneralization by *state-merging.*
- This is a process where two states are identified as equivalent and then *merged* (i.e. combined).
- A key concept behind state merging is that transitions are preserved (Hopcroft et al. 2001, Angluin 1982).
- This is one way in which generalizations may occur (cf. Angluin (1982)).



The learner's state merging criteria

- How does the learner decide whether two states are equivalent in the prefix tree?
- Merge states if their immediate environment is the same.
- I call this environment the *neighborhood*. It is:
  1. the set of incoming symbols to the state
  2. the set of outgoing symbols to the state
  3. whether it is final or not.

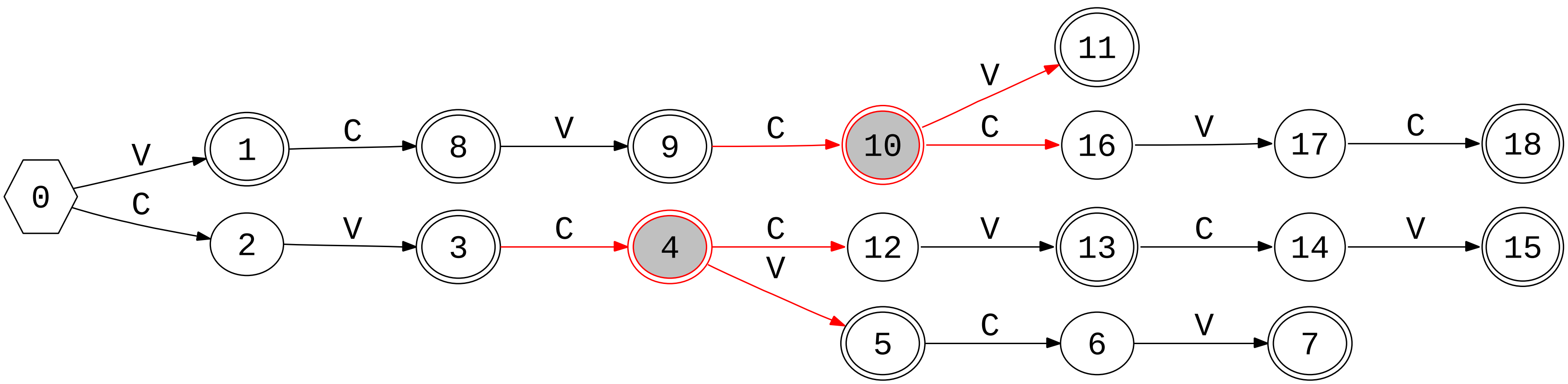
Example of neighborhoods

- State p and q have the same neighborhood.

- The learner merges states in the prefix tree with the same neighborhood.

The prefix tree for the ATR harmony language

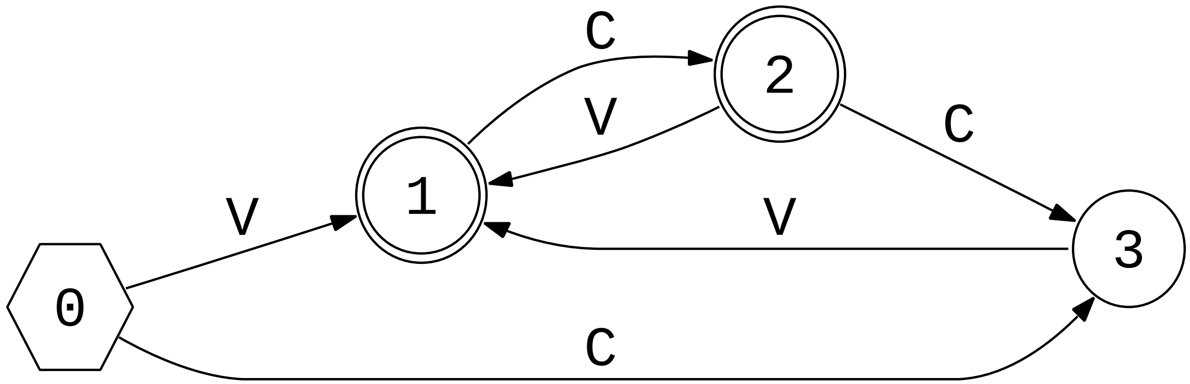
using the [+syl] | [-syl] partition



- States 4 and 10 have the same neighborhood.
- So these states are merged.

The result of merging states with the same neighborhood

(after minimization)



- The machine above accepts

  V,CV,CVC,VCV,CVCV,CVCVC,CVCCVC, ...

- The learner has acquired the syllable structure phonotactic.

- Note there is still overgeneralization because the ATR vowel harmony constraint has not been learned (e.g. *bitE*).

Interim summary of learner

1. Build a prefix tree using some partition by natural class of the segments.
2. Merge states in this machine that have the same neighborhood.

The learner

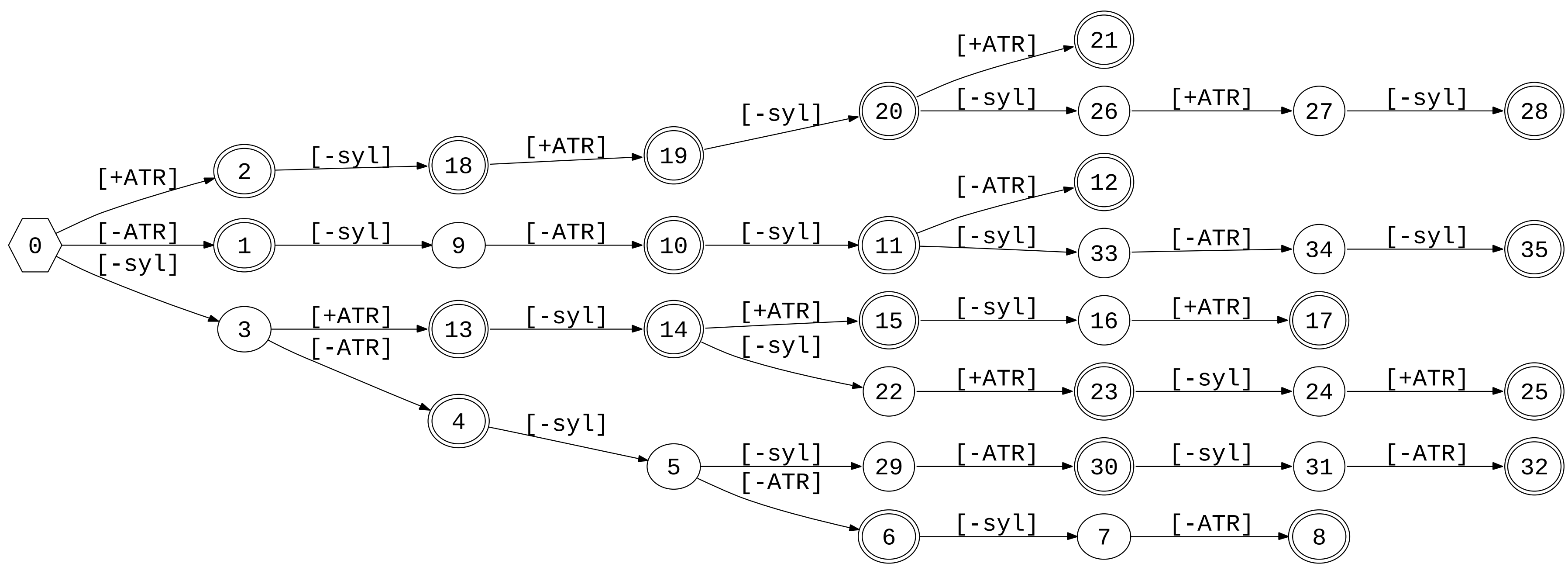
(Now the learner corrects the overgeneralization, e.g. *bitE*)

3. Repeat steps 1-2 with natural classes that partition more finely the segmental inventory.

4. Compare this machine to previously acquired ones, and factor out redundancy by checking for distributional dependencies.

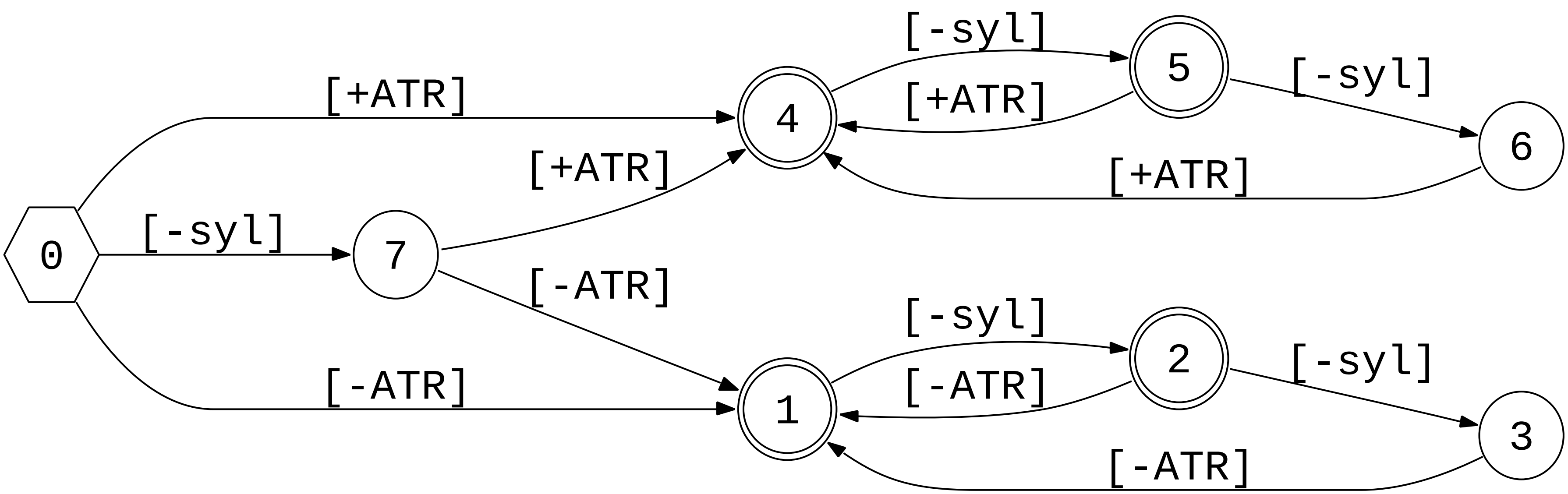
The prefix tree for the ATR harmony language

using the [+syl,+ATR] | [+syl,-ATR] | [-syl] partition



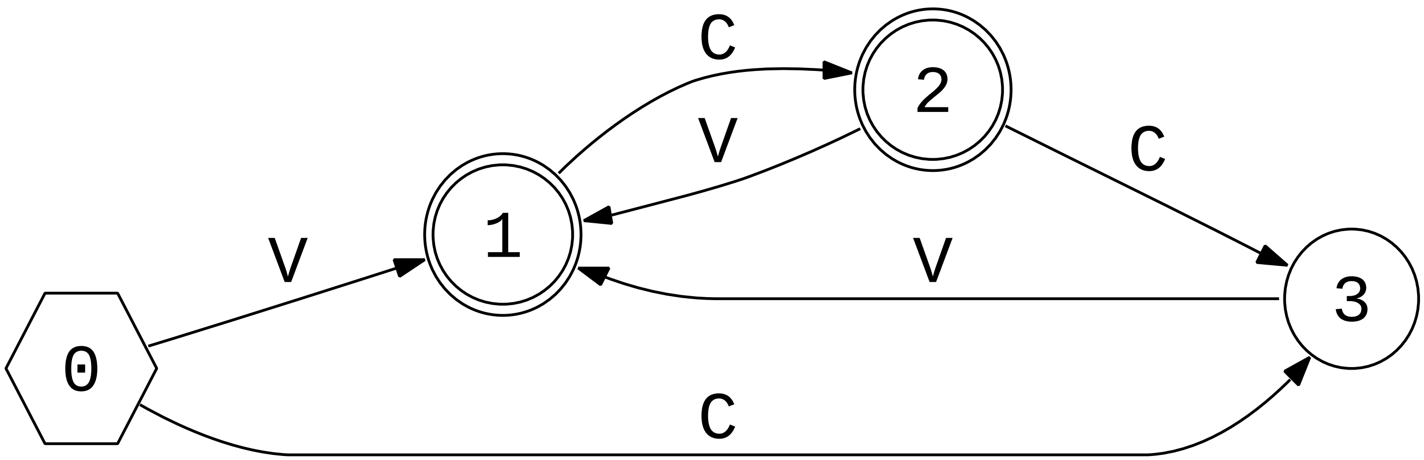
The result of merging states with the same neighborhood

(after minimization)



- The learner has the right language, but redundant syllable structure.

Checking for distributional dependencies



1. Check to see if the distribution of the [ATR] features depends on the distribution of consonants [-syl].

2. Ask if the vocalic paths in the syllable structure machine is traversed by both [+ATR] and [-ATR] vowels.

Checking for distributional dependencies

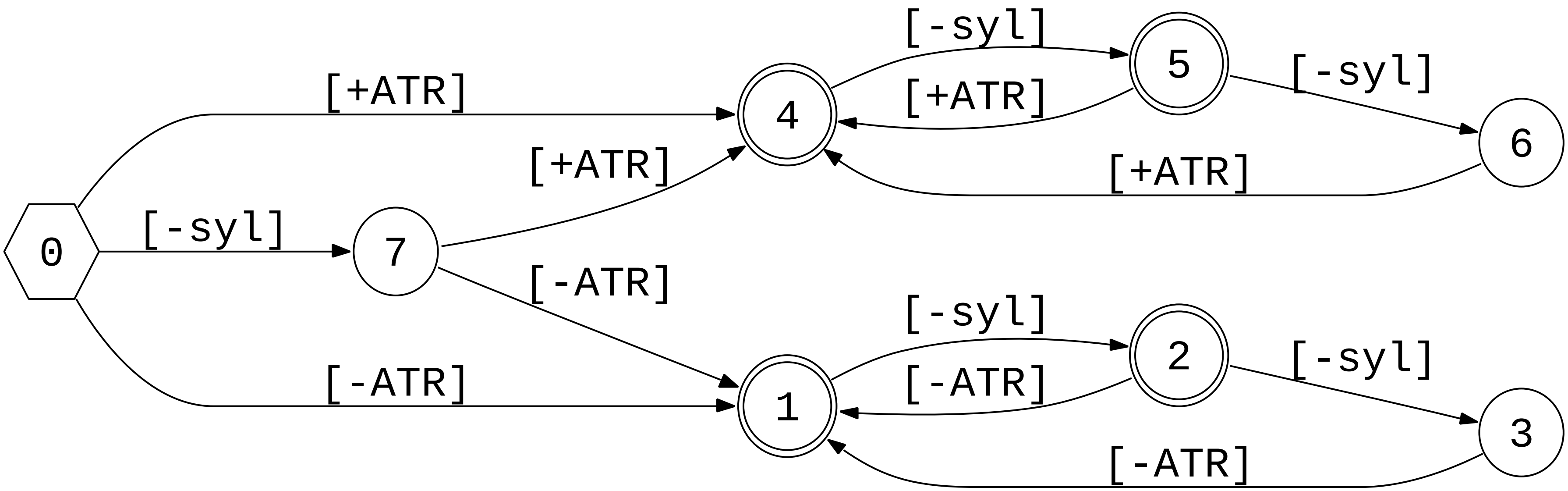
1. How does the learner check if [ATR] is independent of [-syl]?
   1. Remove [+ATR] vowel transitions from the machine, replace the [-ATR] labels with [+syl] labels, and check whether the resulting acceptor accepts the same language as the syllable structure acceptor.
   2. Do the same with the [-ATR] vowels.
   3. If it is in both instances then Yes. Otherwise, No.

Checking for distributional dependencies

2. If Yes– the distribution of ATR is independent of [-syl]– merge states which are connected by transitions bearing the [-syl] (C) label.

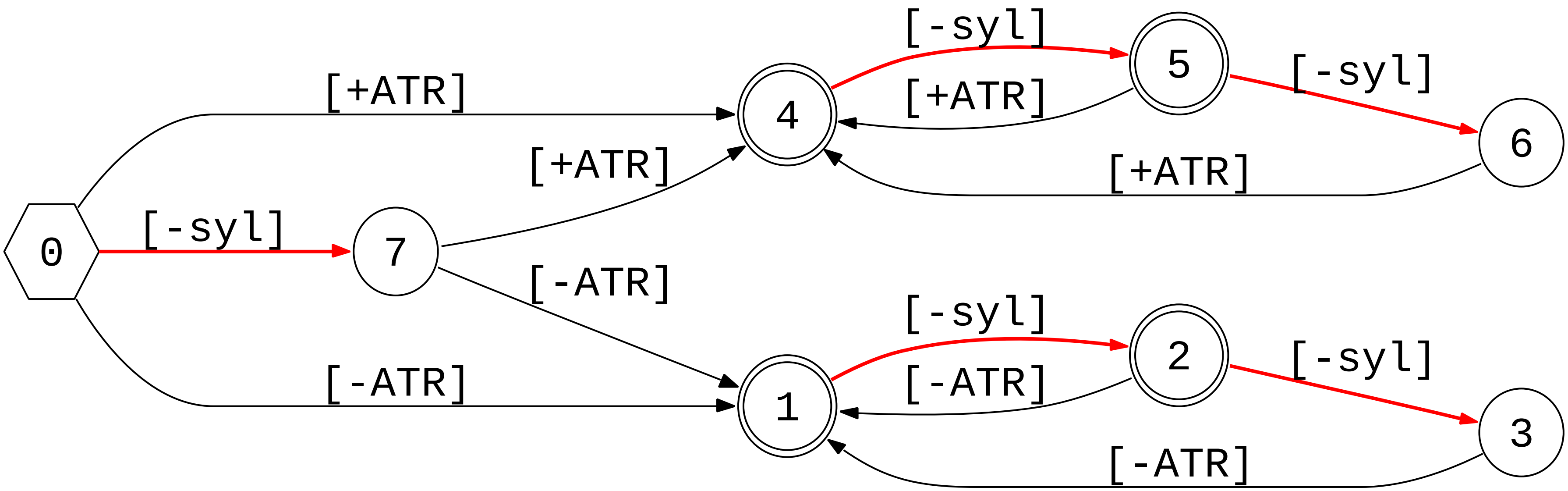
3. If No– the distribution of [ATR] depends on the distribution of [-syl]– then make two machines: one by merging states connected by transitions bearing the [+ATR] label, and one by those bearing the [-ATR] label.

Checking for distributional dependencies



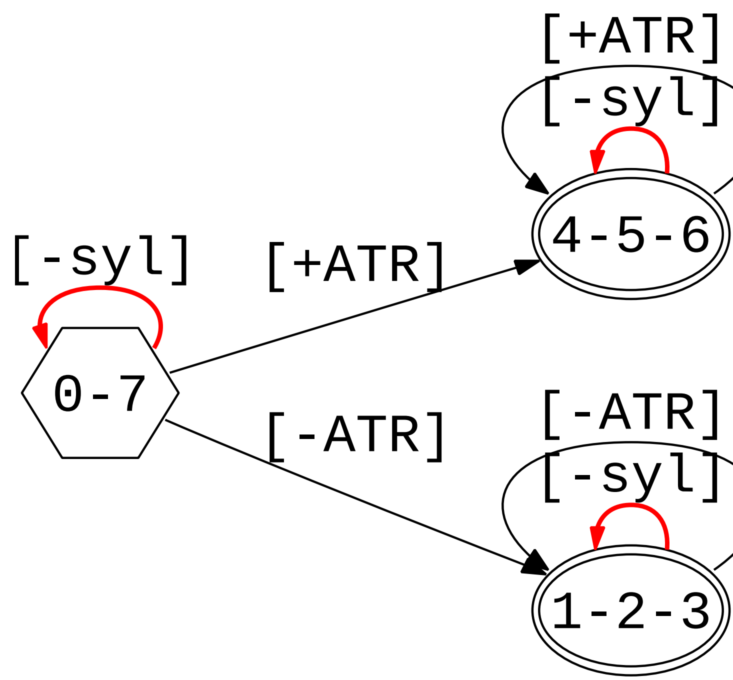
- Since the distribution of [ATR] is independent of the distribution of [-syl], merge states connected by [-syl] transitions.

Checking for distributional dependencies



- Since the distribution of [ATR] is independent of the distribution of [-syl], merge states connected by [-syl] transitions.

Merging states one more time



- The learner has acquired the vowel harmony constraint.

What the algorithm returns

- The algorithm incrementally returns individual finite state machines, each which encodes some regularity about the language.
  - Each individual machine is a phonotactic pattern.
  - Each individual machine is a surface-true constraint.
- A phonotactic grammar is the set of these machines, all of which must be satisfied simultaneously for a word to be acceptable (i.e. the intersection of all the machines is the actual grammar).

Summary of the learner

1. Build a prefix tree of the sample under some partition.
2. Merge states with the same neighborhood.
3. Compare this machine to one acquired earlier under some coarser partition by natural class.
   (a) If the refined blocks in the partition are independent of the static blocks, merge states that are adjoined by static blocks.
   (b) If not, make two machines by merging states adjoined by the refined blocks.
4. Repeat the process.

Other results

- The above algorithm successfully learns the other languages considered in this study (see appendix).

  1. ATR Harmony Language (e.g. Kalenjin (Tucker 1964, Lodge 1995). See also Baković (2000) and references therein).
  2. ATR Contrastive Language (e.g. Akan (Stewart 1967, Ladefoged and Maddieson 1996))
     – The [ATR] feature is freely distributed.
  3. ATR Allophony Language (e.g. Javanese (Archangeli 1995)).
     – -ATR vowels in closed syllables
     – +ATR vowels elsewhere

- A variant of this algorithm learns all the quantity-insensitive stress patterns in Gordon's (2002) typology (Heinz, to appear).

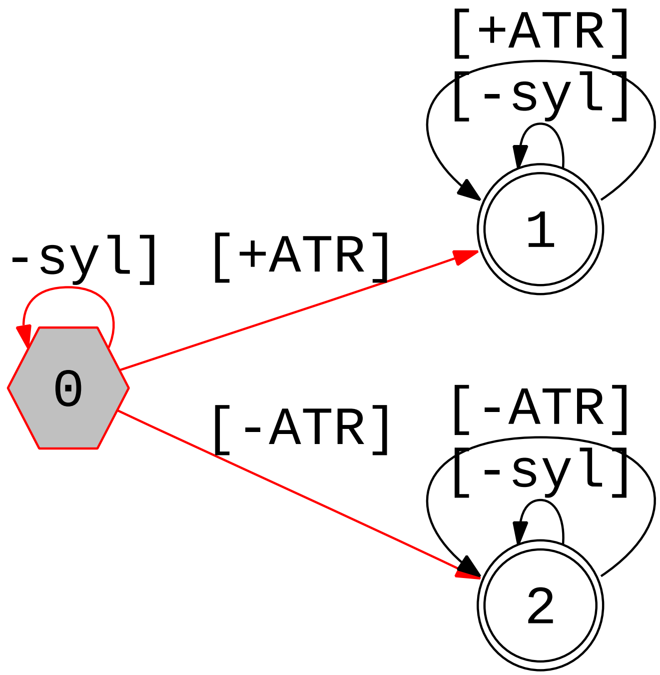
The Neighborhood-distinct Hypothesis:

All phonotactic patterns are neighborhood-distinct.

Neighborhood-distinctness

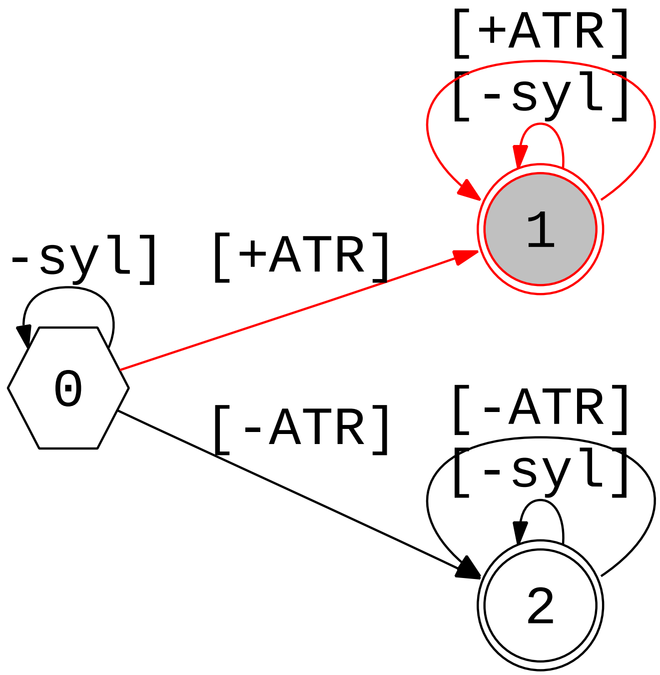
- A language (regular set) is neighborhood-distinct iff there is an acceptor for the language such that each state has its own unique neighborhood.
- Every phonotactic pattern considered to date is *neighborhood-distinct*.

The ATR harmony phonotactic



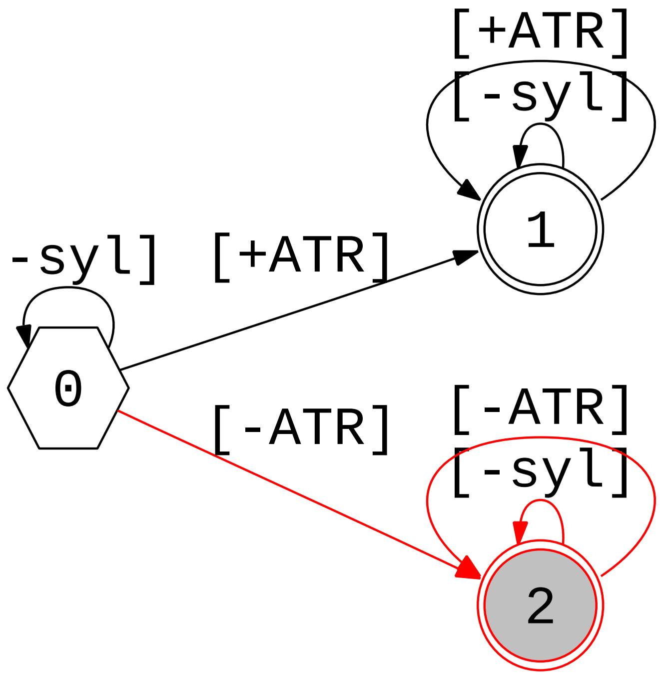
- Neighborhood of State 0
  - Final?=YES
  - Incoming Symbols = {[-syl]}
  - Outgoing Symbols = {[+syl,+ATR],[+syl,-ATR]}

The ATR harmony phonotactic



- Neighborhood of State 1
  - Final?=YES
  - Incoming Symbols = {[-syl],[+syl,+ATR]}
  - Outgoing Symbols = {[+syl,+ATR],[-syl]}

The ATR harmony phonotactic

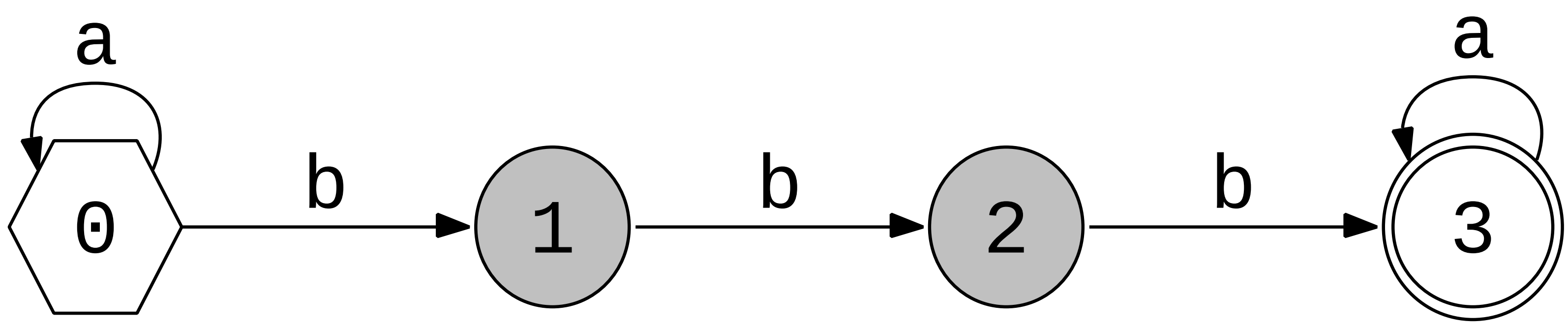


- Neighborhood of State 2
  - Final?=YES
  - Incoming Symbols = {[-syl],[+syl,-ATR]}
  - Outgoing Symbols = {[+syl,-ATR],[-syl]}

Learning Neighborhood-distinctness

- Because the learner merges states with the same neighborhood, it learns neighborhood-distinct patterns.

Example of a non-neighborhood-distinct language: $\mathtt{a}^*\mathtt{bbba}^*$



- It is not possible to construct an acceptor for a language which requires words have exactly three identical adjacent elements...

- because there will always be two states with the same neighborhoods.

Phonology cannot count higher than two

- "Consider first the role of counting in grammar. How long may a count run? General considerations of locality, ...suggest that the answer is probably 'up to two': a rule may fix on one specified element and examine a structurally adjacent element and no other." (McCarthy and Prince 1986:1)
- "Normally, a phonological rule does not count past two ..." (Kenstowicz 1994:372)
- "...the well-established generalization that linguistic rules do not count beyond two ..." (Kenstowicz 1994:597)

Neighborhood-distinctness

- It is an abstract notion of locality.
- It is novel.
- It serves as a strategy for learning by limiting the kinds of generalizations that can be made (e.g. cannot distinguish ‘three’ from ‘more than two’)
- It has global ramifications:
  - It places real limits on machine size: only finitely many languages are neighborhood-distinct.

Conclusions

1. A simple unsupervised batch learning algorithm was presented that succeeds in three case studies.
2. It generalizes successfully using only two notions, natural class and an abstract local notion of environment, the neighborhood.
3. Phonotactic patterns are neighborhood-distinct.

Outstanding issues

1. Efficiency:
   - There may be too many partitions by natural class. How can the learner search this space to find the 'right' ones?
2. The algorithm only learns neighborhood-distinct languages, but not the class of neighborhood-distinct languages.

Future work

1. Are all phonotactic patterns neighborhood-distinct? I.e. how will these results scale up to other phonotactic patterns and real language data?
   (a) Everyone gets simple cases, but are complex phonotactic patterns learnable by this algorithm?
2. What kinds of patterns can the algorithm learn that are not considered possible? Can they be eliminated by other factors?
3. Adapting the algorithm to handle noise (Angluin and Laird 1988).

Thank You.

- Special thanks to Bruce Hayes, Ed Stabler, Colin Wilson and Kie Zuraw for insightful comments and suggestions related to this material. I also thank Greg Kobele, Andy Martin, Katya Pertsova, Shabnam Shademan, Molly Shilman, and Sarah VanWagnenen for helpful discussion.

Appendix: Languages in the study

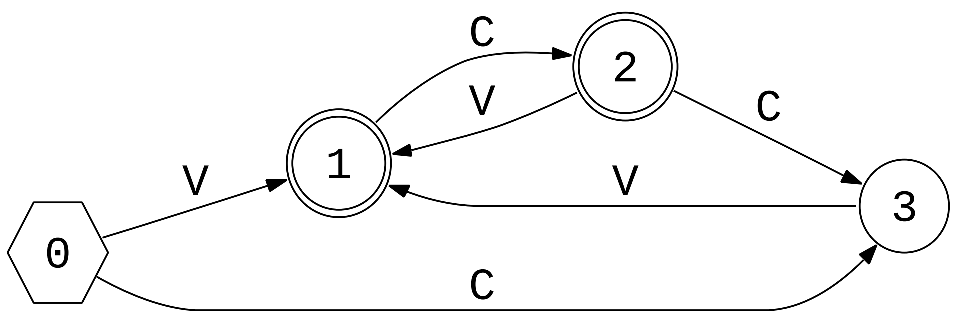
- The languages in the study are all pseudo languages, based on real counterparts.
  1. ATR Harmony Language (e.g. Kalenjin (Tucker 1964, Lodge 1995). See also Baković (2000) and references therein).
  2. ATR Contrastive (everywhere) Language (e.g. Akan (Stewart 1967, Ladefoged and Maddieson 1996))
  3. ATR Allophony Language (e.g. Javanese (Archangeli 1995)).
     - -ATR vowels in closed syllables
     - +ATR vowels elsewhere

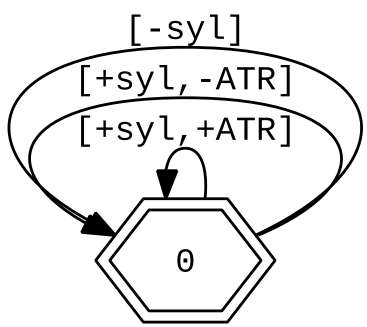
Assumptions

- To simplify matters, assume for all languages:
  1. They are CV(C) (word-initial V optional).
  2. They have ten vowels.
     - {i,u,e,o,a} are [+ATR]
     - {I,U,E,O,A} are [-ATR]
  3. They have 8 consonants {p,b,t,d,k,g,m,n}
  4. Vowels are [+syllabic] and consonants are [-syllabic] and have no value for [ATR].

Pseudo-Akan
Target Grammar

Syllable Structure Phonotactic ⇓





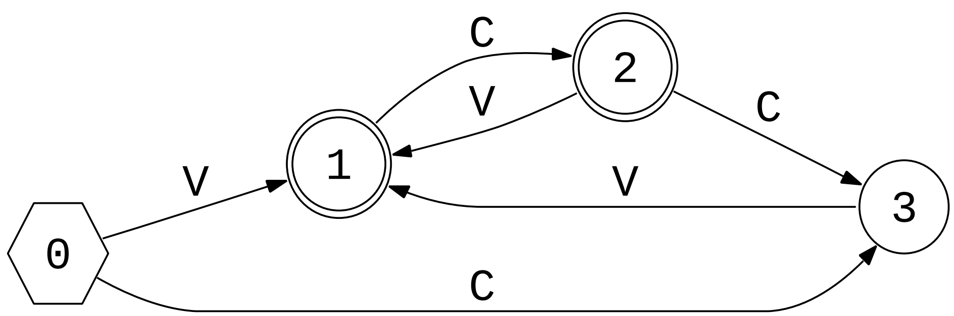
⇐ Free [ATR] Distribution Phonotactic

Pseudo-Akan
Learning Results

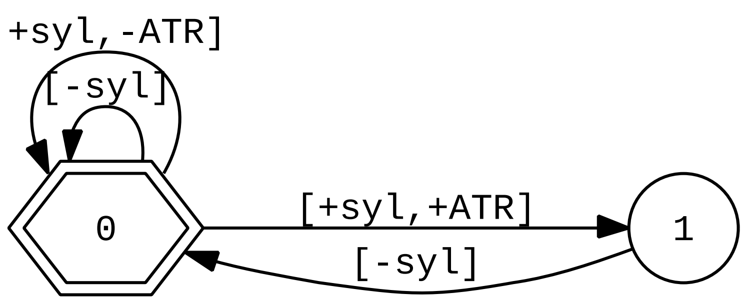
1.	i	8.	montan	15.	Ak	22.	mitIpa
2.	ka	9.	I	16.	IkU	23.	AtetA
3.	eko	10.	kO	17.	atEptAp	24.	pAki
4.	puki	11.	kOn	18.	dAkti	25.	ikOp
5.	atapi	12.	IkUp	19.	bedkO	26.	etIptUp
6.	kitepo	13.	pAtkI	20.	piptApu	27.	potO
7.	bitki	14.	pOtkO	21.	mUtku	28.	kUtepA
						29.	kUptEpA

- With the words in the above table, the learner successfully identifies the target grammar.

Pseudo-Javanese Target Grammar



⇐ Syllable Structure Phonotactic



⇑ [+ATR] vowel must be followed by a consonant.

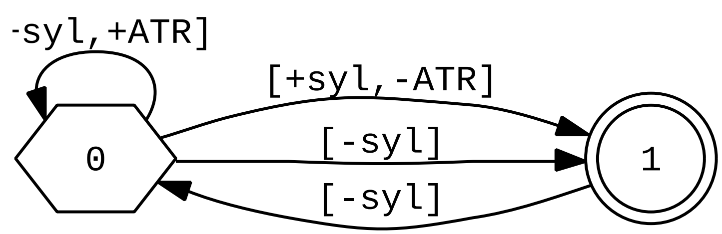
Pseudo-Javanese Learning Results

1.	a	8.	eko	15.	Ak
2.	i	9.	ko	16.	kOn
3.	ka	10.	paki	17.	pAtki
4.	puki	11.	kutepa	18.	kUptapa
5.	kitepo	12.	poto	19.	pOtko
6.	pati	13.	ateta	20.	atEptAp
7.	atapi	14.	iku	21.	ikUp

- With the words in the above table, the learner successfully identifies the target grammar.

Pseudo-Javanese
Learning Results

- The learner also learns another phonotactic for this grammar (shown below).
- This phonotactic says that a CC sequence and [-ATR]C sequence must be followed by a vowel.



References

Albright, Adam. 2006. Gradient Phonotactic effects: lexical? grammatical? both? neither? Talk handout from the 80th Annual LSA Meeting, Albuquerque, NM.

Albright, Adam and Bruce Hayes. 2002. Modeling English past tense intuitions with minimal generalization. *SIGPHON 6: Proceedings of the Sixth Meeting of the ACL Special Interest Group in Computational Phonology* :58–69.

Albright, Adam and Bruce Hayes. 2003. Rules vs. Analogy in English Past Tenses: A Computational/Experimental Study. *Cognition* 90:119–161.

Albro, Dan. 1998. Evaluation, implementation, and extension of Primitive Optimality Theory. Master's thesis, University of California, Los Angeles.

Albro, Dan. 2005. A Large-Scale, LPM-OT Analysis of Malagasy. Ph.D. thesis, University of California, Los Angeles.

Alderete, John, Adrian Brasoveanua, Nazarre Merchant, Alan Prince, and Bruce Tesar. 2005. Contrast analysis aids in the learning of phonological underlying forms. In *The Proceedings of WCCFL 24*. pages 34–42.

Angluin, Dana. 1982. Inference of Reversible Languages. *Journal for the Association of Computing Machinery* 29(3):741–765.

Angluin, Dana and Philip Laird. 1988. Learning from Noisy Examples. *Machine Learning* 2:343–370.

Archangeli, Diana. 1995. The Grounding Hypothesis and Javanese Vowels. In *Papers from the Fifth Annual Meeting of the Southeast Asian Linguistics Society (SEALSV)*, edited by S. Chelliah and W. de Reuse. pages 1–19.

Baković, Eric. 2000. Harmony, Dominance and Control. Ph.D. thesis, Rutgers University.

Boersma, Paul. 1997. How we learn variation, optionality, and probability. *Proceedings of the Institute of Phonetic Sciences* 21.

Boersma, Paul and Bruce Hayes. 2001. Empirical tests of the Gradual Learning Algorithm. *Linguistic Inquiry* 32:45–86.

Chomsky, Noam and Morris Halle. 1968. *The Sound Pattern of English*. Harper & Row.

Coleman, John and Janet Pierrehumbert. 1997. Stochastic Phonological Grammars and Acceptability. In *Compuational Phonolgy*. Somerset, NJ: Association for Computational Linguistics, pages 49–56. Third Meeting of the ACL Special Interest Group in Computational Phonology.

Dresher, Elan and Jonathan Kaye. 1990. A Computational Learning Model for Metrical Phonology. *Cognition* 34:137–195.

Eisner, Jason. 1997. What Constraints Should OT Allow? Talk handout, Linguistic Society of America, Chicago. Available on the Rutgers Optimality Archive, ROA#204-0797, http://roa.rutgers.edu/.

Ellison, T.M. 1994. The Iterative Learning of Phonological Constraints. *Computational Linguistics* 20(3).

Frisch, S., J. Pierrehumbert, and M. Broe. 2004. Similarity Avoidance and the OCP. *Natural Language and Linguistic Theory* 22:179–228.

Frisch, Stephan. 1996. Similarity and Frequency in Phonology. Ph.D. thesis, Northwestern University.

Gildea, Daniel and Daniel Jurafsky. 1996. Learning Bias and Phonological-rule Induction. *Association for Computational Linguistics* .

Goldsmith, John. 2006. Information Thoery and Phonology. Slides presented at the 80th Annual LSA in Alburquerque, New Mexico.

Gordon, Matthew. 2002. A Factorial Typology of Quantity-Insensitive Stress. *Natural Language and Linguistic Theory* 20(3):491–552. Additional appendices availables at http://www.linguistics.ucsb.edu/faculty/gordon/pubs.html.

Halle, Morris. 1978. Knowledge Unlearned and Untaught: What Speakers Know about the Sounds of Their Language. In *Linguistic Theory and Psychological Reality*. The MIT Prss.

Hayes, Bruce. 1999. Phonetically-Driven Phonology: The Role of Optimality Theory and Inductive Grounding. In *Functionalism and Formalism in Linguistics, Volume I: General Papers*. John Benjamins, Amsterdam, pages 243–285.

Hayes, Bruce. 2004. Phonological acquisition in Optimality Theory: the early stages. In *Fixing Priorities: Constraints in Phonological Acquisition*, edited by Rene Kager, Joe Pater, and Wim Zonneveld. Cambridge University Press.

Hayes, Bruce and Colin Wilson. 2006. The UCLA Phonotactic Learner. Talk handout from UCLA Phonology Seminar.

Heinz, Jeffrey. To appear. Learning Quantity Insensitive Stress Systems via Local Inference. In *Proceedings of the Association of Computational Linguistics Special Interest Group in Phonology (ACL-SIGPHON 2006)*. New York City.

Hopcroft, John, Rajeev Motwani, and Jeffrey Ullman. 2001. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley.

Johnson, C. Douglas. 1972. *Formal Aspects of Phonological Description*. The Hague: Mouton.

Kaplan, Ronald and Martin Kay. 1981. Phonological Rules and Finite State Transducers. Paper presented at ACL/LSA Conference, New York.

Kaplan, Ronald and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics* 20(3):331–378.

Karttunen, Lauri. 1998. The proper treatment of optimality theory in computational phonology. *Finite-state methods in natural language processing* :1–12.

Kenstowicz, Michael. 1994. *Phonology in Generative Grammar*. Blackwell Publishers.

Ladefoged, Peter and Ian Maddieson. 1996. *Sounds of the World's Languages*. Blackwell Publishers.

Lin, Ying. 2002. Probably Approximately Correct Learning of Constraint Ranking. Master's thesis, University of California, Los Angeles.

Lodge, Ken. 1995. Kalenjin phonology and morphology: A further exemplification of underspecification and non-destructive phonology. *Lingua* 96:29–43.

McCarthy, John and Alan Prince. 1986. Prosodic Morphology. Ms., Department of Linguistics, University of Massachusetts, Amherst, and Program in Linguistics, Brandeis University, Waltham, Mass.

Merchant, Nazarre and Bruce Tesar. to appear. Learning underlying forms by searching restricted lexical subspaces. In *The Proceedings of CLS 41*. ROA-811.

Pater, Joe. 2004. Exceptions and Optimality Theory: Typology and Learnability. Conference on Redefining Elicitation: Novel Data in Phonological Theory. New York University.

Pater, Joe and Anne Marie Tessier. 2003. Phonotactic Knowledge and the Acquisition of Alternations. In *Proceedings of the 15th International Congress on Phonetic Sciences, Barcelona*, edited by M.J. Solé, D. Recasens, and J. Romero. pages 1777–1180.

Prince, Alan and Paul Smolensky. 1993. Optimality Theory: Constraint Interaction in Generative Grammar. Technical Report 2, Rutgers University Center for Cognitive Science.

Prince, Alan and Paul Smolensky. 2004. *Optimality Theory: Constraint Interaction in Generative Grammar*. Blackwell Publishing.

Prince, Alan and Bruce Tesar. 2004. Fixing priorities: constraints in phonological acquisition. In *Fixing Priorities: Constraints in Phonological Acquisition*. Cambridge: Cambridge University Cambridge: Cambride Univeristy Press.

Riggle, Jason. 2004. Generation, Recognition, and Learning in Finite State Optimality Theory. Ph.D. thesis, University of California, Los Angeles.

Riggle, Jason. 2006. Using Entropy to Learn OT Grammars From Surface

Forms Alone. Slides presented at WCCFL 25, University of Seattle, Washington.

Stewart, J.M. 1967. Tongue root position in Akan vowel harmony. *Phonetica* 16:185–204.

Tesar, Bruce. 1995. Computational Optimality Theory. Ph.D. thesis, University of Colorado at Boulder.

Tesar, Bruce. 1998. An Interative Strategy for Language Learning. *Lingua* 104:131–145.

Tesar, Bruce and Paul Smolensky. 1998. Learnability in Optimality Theory. *Linguistic Inquiry* 29:229–268.

Tucker, Archibald N. 1964. Kalenjin Phonetics. In *In Honour of Daniel Jones*, edited by D. Abercrombie, D. B. Fry, P. A. D. MacCarthy, N. C. Scott, and J. L. M. Trim. Longmans, London, pages 445–470.

Wexler, Kenneth and Peter Culicover. 1980. *Formal Principles of Language Acquisition*. MIT Press.

Wilson, Colin. 2006. The Choice Luce Ranker. Talk Handout presented at the UCLA Phonoloogy Seminar.