Grammatical inference and subregular phonology

Adam Jardine
Rutgers University

December 11, 2019 · Tel Aviv University

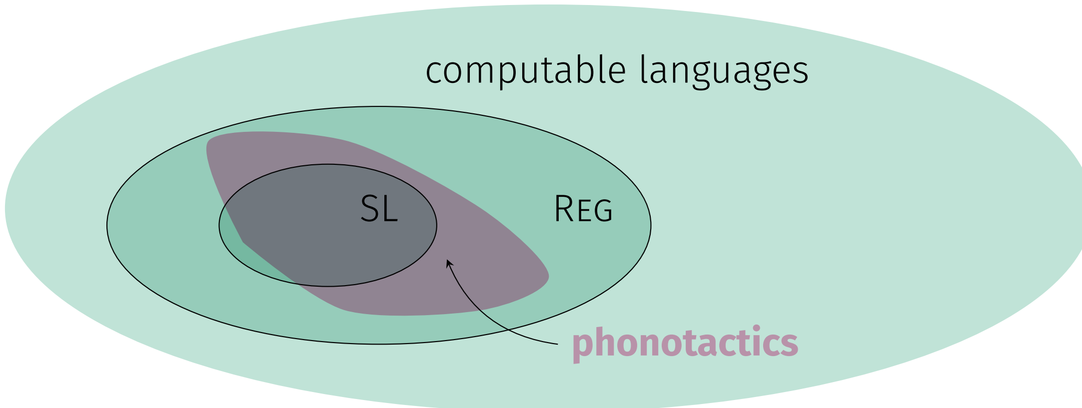
Review

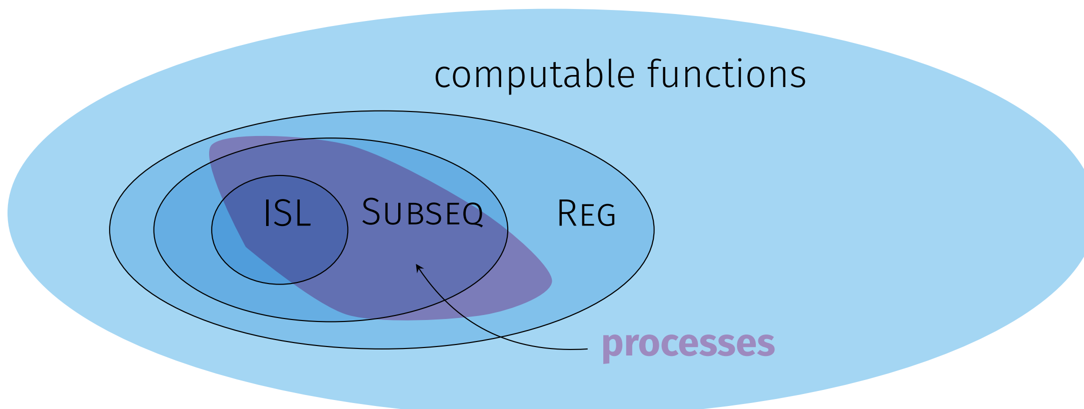
“[V]arious formal and substantive universals are intrinsic properties of the language-acquisition system, these providing a schema that is applied to data and that determines in a highly restricted way the general form and, in part, even the substantive features of the grammar that may emerge upon presentation of appropriate data.”

Chomsky, *Aspects*

“[I]f an algorithm performs well on a certain class of problems then it necessarily pays for that with degraded performance on the set of all remaining problems.”

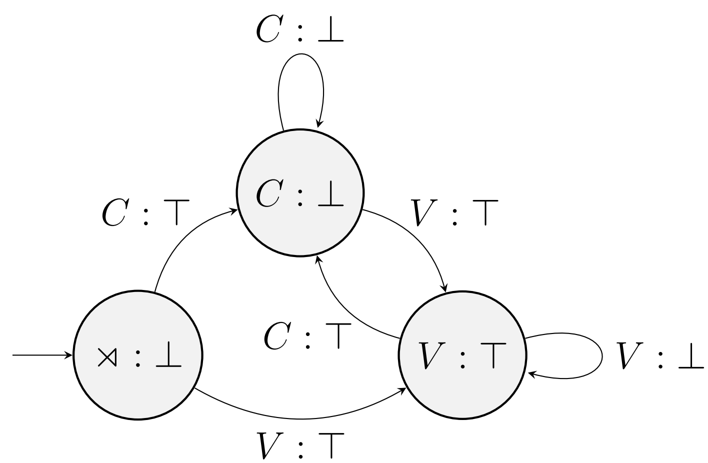
Wolpert and Macready (1997), *NFL Thms.*


computable languages
SL
Reg
phonotactics


computable functions
ISL
Subseq
Reg
processes

Review

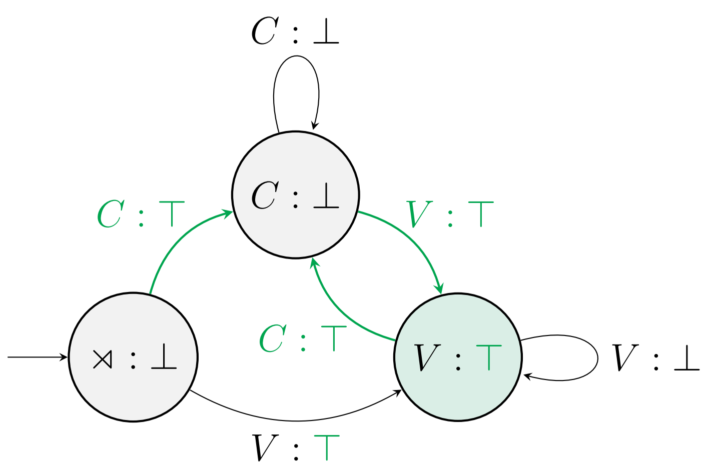
- Computational characterizations of phonological patterns identify **structure** that can be used by a learner



Review

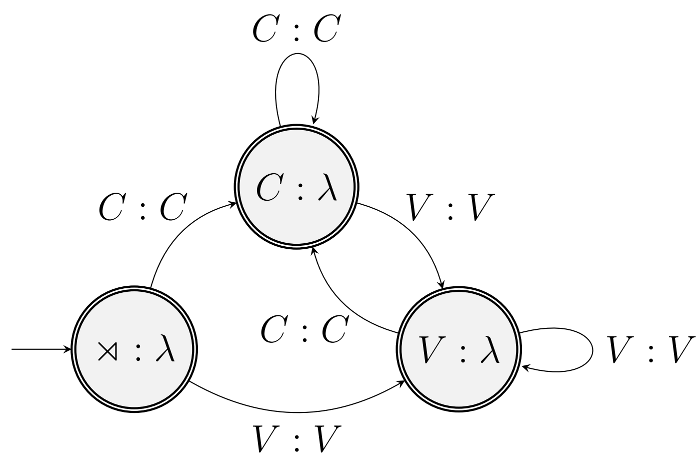
- Computational characterizations of phonological patterns identify **structure** that can be used by a learner

	data
0	CV
1	V
2	$CVCV$



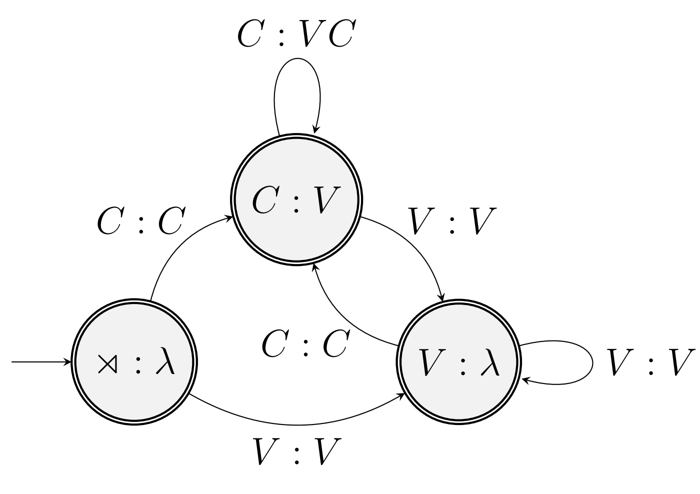
Review

- Computational characterizations of phonological patterns identify **structure** that can be used by a learner



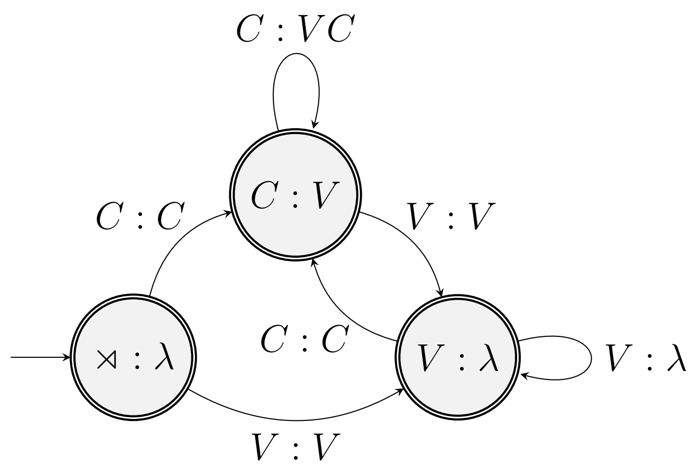
Review

- Computational characterizations of phonological patterns identify **structure** that can be used by a learner



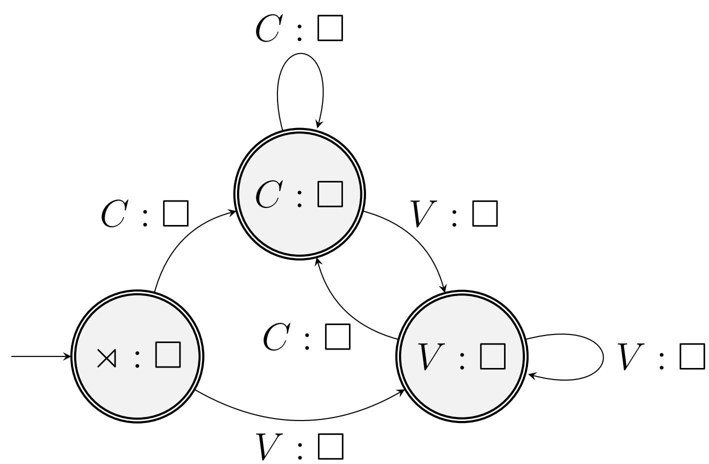
Review

- Computational characterizations of phonological patterns identify **structure** that can be used by a learner



Review

- Computational characterizations of phonological patterns identify **structure** that can be used by a learner



Today

- Using automata structure for learning
  - ISL functions
  - **SL distributions**
- Open questions

Learning ISL functions

Learning input strictly local functions

- When learning languages, presentation is a sequence of examples of L

t	datum
0	V
1	$CVCV$
2	$CVVCVCV$
$\vdots$	

- When learning functions, …

Learning input strictly local functions

- When learning languages, presentation is a sequence of examples of L

t	datum
0	V
1	$CVCV$
2	$CVVCVCV$
$\vdots$	

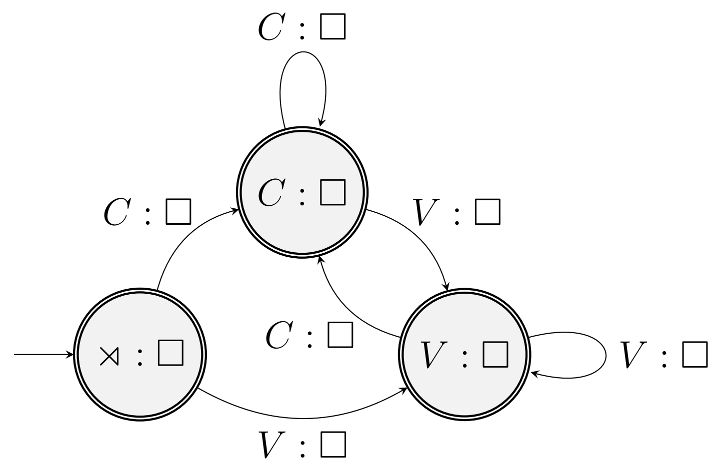
- When learning functions, presentation is of example pairs from f

t	datum
0	(C, CV)
1	$(CVC, CVCV)$
2	$(CVCV, CVCV)$
$\vdots$	

Learning input strictly local functions

t	datum
0	(C, CV)
1	$(CVC, CVCV)$
2	$(CVCV, CVCV)$
3	$(VCVC, VCVCV)$

$\xrightarrow{?}$



Learning input strictly local functions

- The **longest common prefix (lcp)** is the longest initial sequence shared by a set of strings

$$\texttt{lcp}(\{CVCV, CVCCV, CVCVC\}) =$$

Learning input strictly local functions

- The **longest common prefix (lcp)** is the longest initial sequence shared by a set of strings

$$\texttt{lcp}(\{CVCV, CVCCV, CVCVC\}) = CVC$$

$$\texttt{lcp}(\{CVCV, CCVCV, CVCVC\}) =$$

Learning input strictly local functions

- The **longest common prefix (lcp)** is the longest initial sequence shared by a set of strings

$$\texttt{lcp}(\{CVCV, CVCCV, CVCVC\}) = CVC$$

$$\texttt{lcp}(\{CVCV, CCVCV, CVCVC\}) = C$$

Learning input strictly local functions

- The **longest common prefix (lcp)** is the longest initial sequence shared by a set of strings

$$\texttt{lcp}(\{CVCV, CVCCV, CVCVC\}) = CVC$$

$$\texttt{lcp}(\{CVCV, CCVCV, CVCVC\}) = C$$

- Call our data sequence $d \subset f$

(CV, CV)
(CVC, CVC)
$(CVCVC, CVCVC)$
$(VCVVC, VCVC)$
$(VCVV, VCV)$

$$d^p(w) = \texttt{lcp}(d(w\Sigma^*))$$

Learning input strictly local functions

- The **longest common prefix (lcp)** is the longest initial sequence shared by a set of strings

$$\texttt{lcp}(\{CVCV, CVCCV, CVCVC\}) = CVC$$

$$\texttt{lcp}(\{CVCV, CCVCV, CVCVC\}) = C$$

- Call our data sequence $d \subset f$

(CV, CV)
(CVC, CVC)
$(CVCVC, CVCVC)$
$(VCVVC, VCVC)$
$(VCVV, VCV)$

$d^p(w) = \texttt{lcp}(d(w\Sigma^*))$
$d^p(CVC) = ...$

Learning input strictly local functions

- The **longest common prefix (lcp)** is the longest initial sequence shared by a set of strings

$$\texttt{lcp}(\{CVCV, CVCCV, CVCVC\}) = CVC$$

$$\texttt{lcp}(\{CVCV, CCVCV, CVCVC\}) = C$$

- Call our data sequence $d \subset f$

$$\begin{array}{ll} (CV, CV) & d^p(w) = \texttt{lcp}(d(w\Sigma^*)) \\ (CVC, CVC) & d^p(CVC) = CVC \\ (CVCVC, CVCVC) & d^p(VCVV) = ... \\ (VCVVC, VCVC) & \\ (VCVV, VCV) & \end{array}$$

Learning input strictly local functions

- The **longest common prefix (lcp)** is the longest initial sequence shared by a set of strings

$$\texttt{lcp}(\{CVCV, CVCCV, CVCVC\}) = CVC$$

$$\texttt{lcp}(\{CVCV, CCVCV, CVCVC\}) = C$$

- Call our data sequence $d \subset f$

(CV, CV)
(CVC, CVC)
$(CVCVC, CVCVC)$
$(VCVVC, VCVC)$
$(VCVV, VCV)$

$d^p(w) = \texttt{lcp}(d(w\Sigma^*))$
$d^p(CVC) = CVC$
$d^p(VCVV) = VCV$

Learning input strictly local functions

- Call our data sequence $d \subset f$

$$
\begin{array}{ll}
(CV, CV) & d^p(w) = \texttt{lcp}(d(w\Sigma^*)) \\
(CVC, CVC) & d^p(CVC) = CVC \\
(CVCVC, CVCVC) & d^p(VCVV) = VCV \\
(VCVVC, VCVC) & \\
(VCVV, VCV) & d_w(u) = d^p(w)^{-1}d(wu)
\end{array}
$$

Learning input strictly local functions

- Call our data sequence $d \subset f$

$$(CV, CV)$$
$$(CVC, CVC)$$
$$(CVCVC, CVCVC)$$
$$(VCVVC, VCVC)$$
$$(VCVV, VCV)$$

$$d^p(w) = \texttt{lcp}(d(w\Sigma^*))$$
$$d^p(CVC) = CVC$$
$$d^p(VCVV) = VCV$$

$$d_w(u) = d^p(w)^{-1}d(wu)$$
$$d_{CV}(C) = d^p(CV)^{-1}d(CVC)$$
$$= (CV)^{-1}CVC \quad = C$$

Learning input strictly local functions

- Call our data sequence $d \subset f$

$$(CV, CV)$$
$$(CVC, CVC)$$
$$(CVCVC, CVCVC)$$
$$(VCVVC, VCVC)$$
$$(VCVV, VCV)$$

$$d^p(w) = \texttt{lcp}(d(w\Sigma^*))$$
$$d^p(CVC) = CVC$$
$$d^p(VCVV) = VCV$$

$$d_w(u) = d^p(w)^{-1}d(wu)$$
$$d_{CV}(C) = d^p(CV)^{-1}d(CVC)$$
$$= (CV)^{-1}CVC \quad = C$$
$$d_{VCV}(V) = d^p(VCV)^{-1}d(VCVV)$$
$$= (VCV)^{-1}VCV = \lambda$$

Learning input strictly local functions

- Call our data sequence $d \subset f$

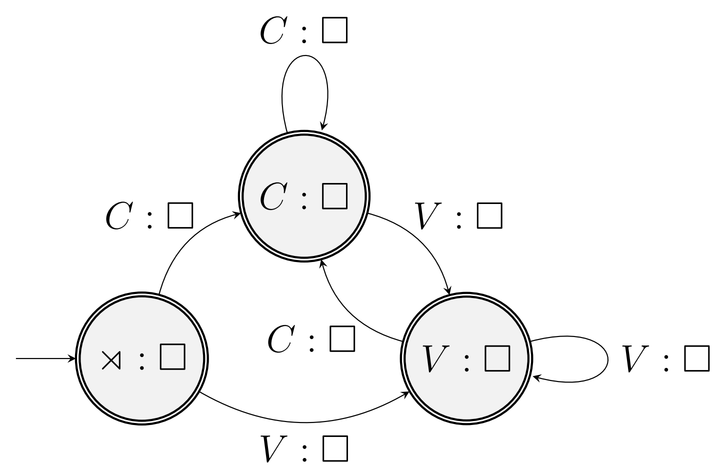
$$
\begin{aligned}
&(CV, CV)\\
&(CVC, CVC)\\
&(CVCVC, CVCVC)\\
&(VCVVC, VCVC)\\
&(VCVV, VCV)
\end{aligned}
$$

$$
\begin{aligned}
d^p(w) &= \texttt{lcp}(d(w\Sigma^*))\\
d^p(CVC) &= CVC\\
d^p(VCVV) &= VCV
\end{aligned}
$$

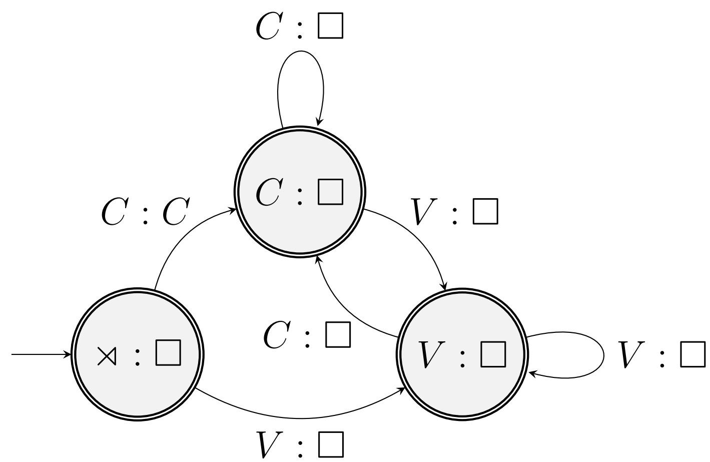
$$
\begin{aligned}
d_w(u) &= d^p(w)^{-1}d(wu)\\
d_{CV}(C) &= d^p(CV)^{-1}d(CVC)\\
&= (CV)^{-1}CVC \quad = C\\
d_{VCV}(V) &= d^p(VCV)^{-1}d(VCVV)\\
&= (VCV)^{-1}VCV = \lambda
\end{aligned}
$$

$$d^p_w(u) = \texttt{lcp}(d_w(u\Sigma^*))$$

(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)

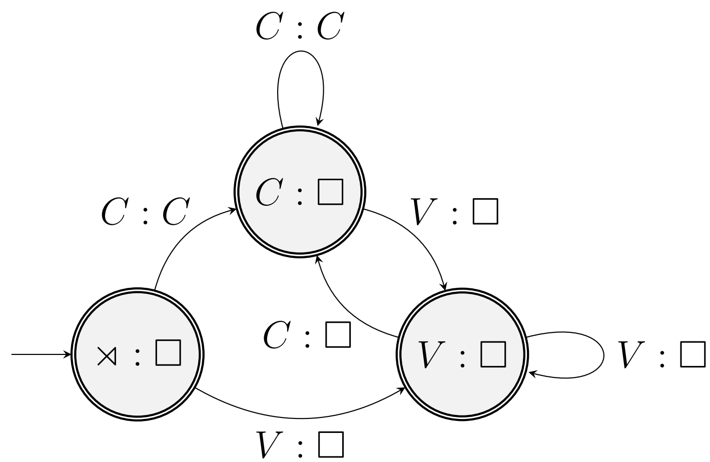


(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)



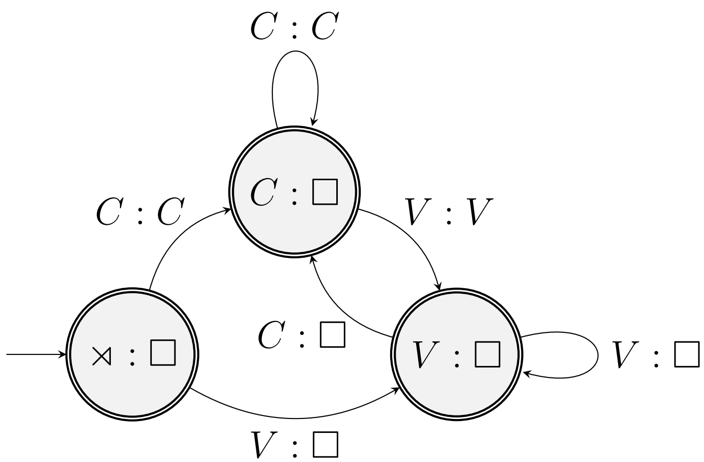
$$d_{\lambda}^{p}(C) = C$$

(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)



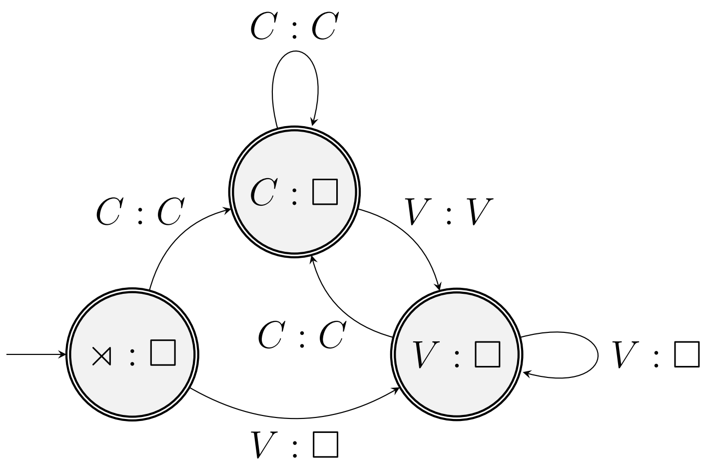
$$d_C^p(C) = C$$

(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)



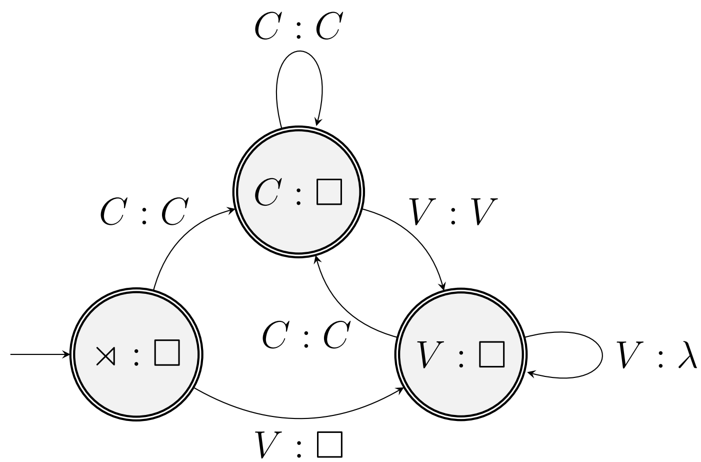
$$d_C^p(V) = V$$

(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)



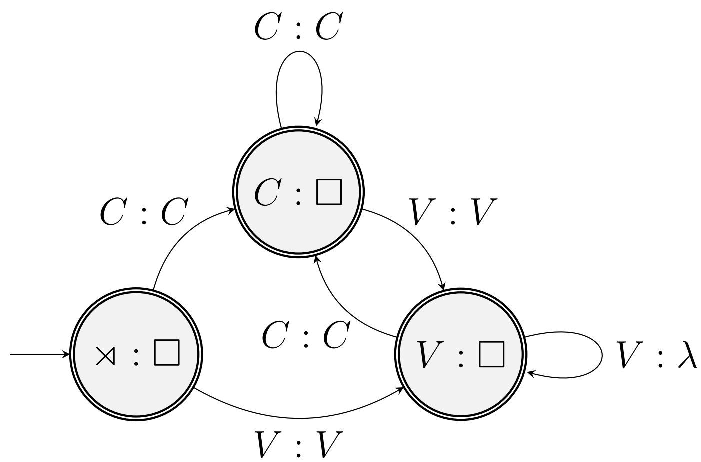
$$d^{p}_{CV}(C) = C$$

(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)



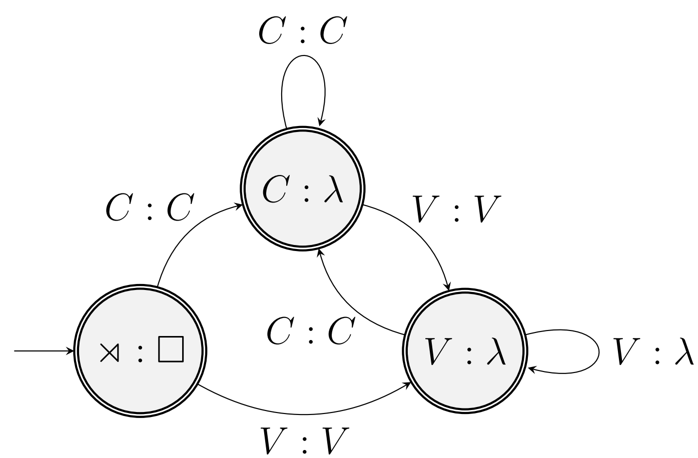
$$d^{p}_{CV}(V) = \lambda$$

(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)



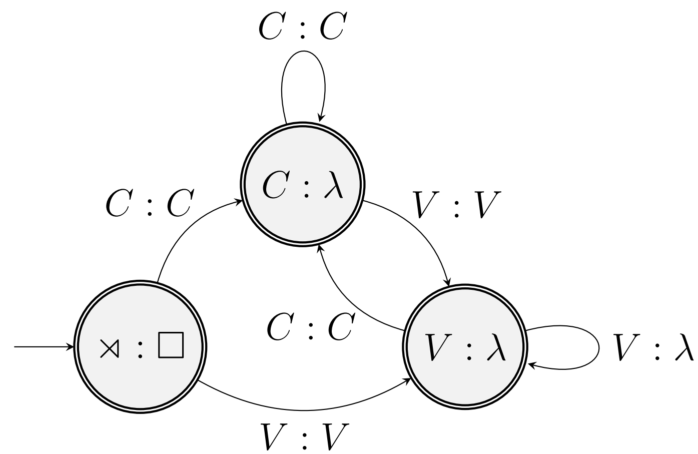
$$d^p_\lambda(V) = V$$

(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)



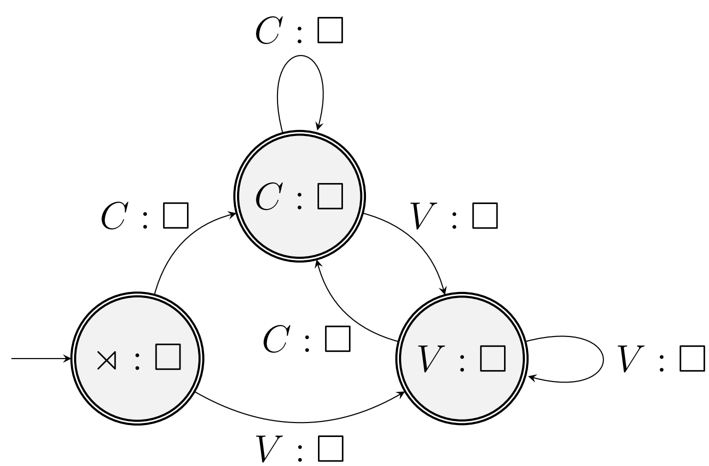
$$d^p(CVC)^{-1}d(CVC) = \lambda, \quad d^p(V)^{-1}d(V) = \lambda$$

(CVC, CVC)
(CVV, CV)
$(CVCCV, CVCCV)$
$(CCVCC, CCVCC)$
$(CCCVCV, CCCVCV)$
$(CVVCV, CVCV)$
(V, V)



Learning input strictly local functions

- As any two ISL$_k$ functions share the same structure, this method ILPD-learns the ISL$_k$ functions



- This method extends to *any* class of functions that shares such a structure (Jardine et al., 2014)

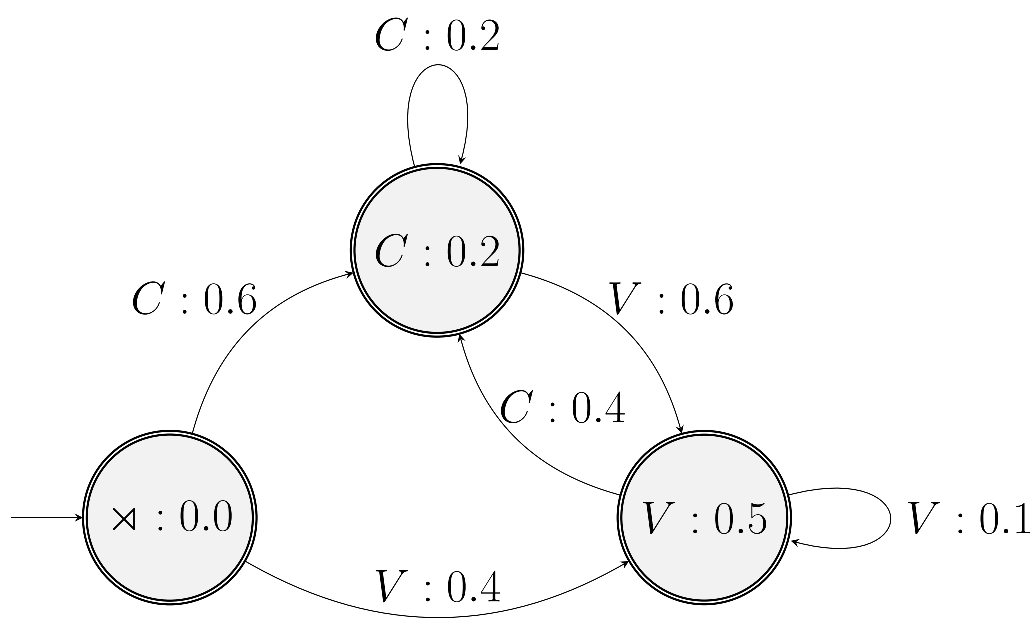
Learning input strictly local functions

- A learning algorithm for grammars that explicitly encode computational properties of phonological patterns
- Learning for OSL (Chandlee et al., 2015) and tier-based OSL (Burness and McMullin, 2019) use a similar (yet distinct) method
- Learning URs uses this same structural concept (Hua et al. in progress)
- Learning for optional ISL processes uses the same basic idea (Heinz in progress) based on Beros and de la Higuera (2016)

Learning SL distributions

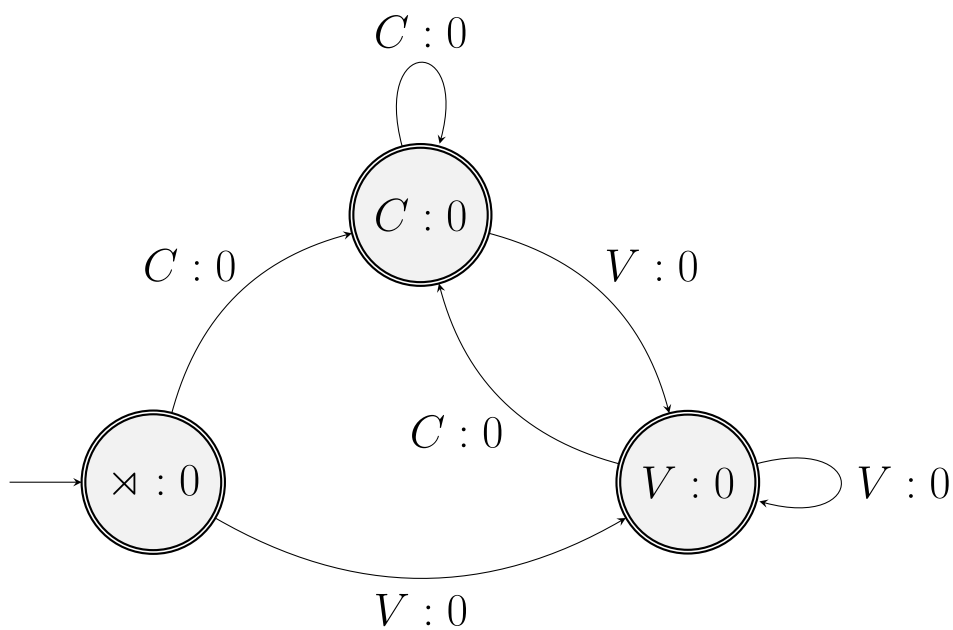
Learning strictly local distributions

- Probability distributions can be described with **the same structure**.



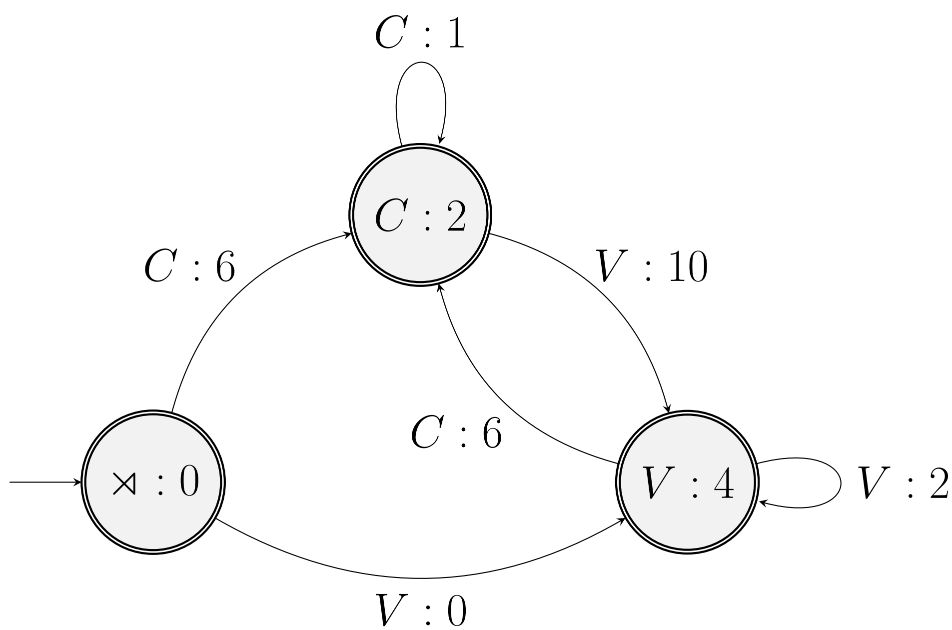
Learning strictly local distributions

CVC
CVV
$CVCCV$
$CVCVC$
$CVCV$
$CVVCV$



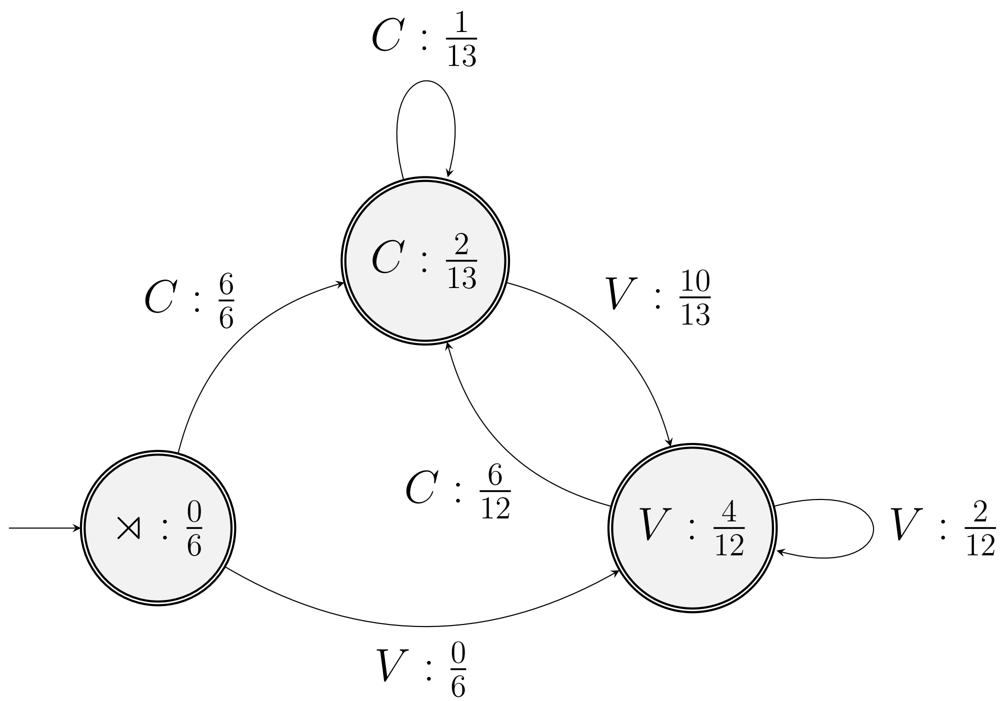
Learning strictly local distributions

CVC
CVV
$CVCCV$
$CVCVC$
$CVCV$
$CVVCV$



Learning strictly local distributions

CVC
CVV
$CVCCV$
$CVCVC$
$CVCV$
$CVVCV$



Learning structured distributions

- This same technique can be extended to...
  - Learning strictly piecewise distributions: Heinz and Rogers (2010)
  - Learning SL distributions over features: Heinz and Koirala (2010)
  - ...

Review

- Studying computational principles that underly phonological patterns identify structural properties for learning:
  - phonotactics
  - processes
  - stochastic generalizations
- A theory of phonology based on these principles derives typological predictions from learning

Open questions

- Non-string representations are best characterized using **logic** (Jardine, 2016; Strother-Garcia, 2017)
- Learning with logic is a wide-open question (Strother-Garcia et al., 2016)
- Learning using features (Chandlee et al., 2019)
- Learning URs (Hua et al., in progress)
- Learning optionality (Heinz et al., in progress) and stochastic processes (wide open)
- Distinguishing accidental versus systematic gaps (Rawski in progress)

Open questions

- A useful tool:

  https://github.com/alenaks/SigmaPie