Learning with Partially Ordered Representations

Jane Chandlee, Remi Eyraud, Jeffrey Heinz, Adam Jardine, Jonathan Rawski

Jane Chandlee
(Haverford)

Remi Eyraud
(Aix-Marseille)

Jeff Heinz
(Stony Brook)

Adam Jardine
(Rutgers)

The Talk in a Nutshell

Previously

- Efficient Learning of subregular languages and functions
- Question: How to extend these learners for multiple, shared properties?

Today

- Describe model-theoretic characterization of strings and trees
- Describe the partial order structure of the space of feature-based hypotheses
- Showcase a learning algorithm which exploits this structure to generalize from data to grammars.

The Talk in a Nutshell

Previously

- Efficient Learning of subregular languages and functions
- Question: How to extend these learners for multiple, shared properties?

Today

- Describe model-theoretic characterization of strings and trees
- Describe the partial order structure of the space of feature-based hypotheses
- Showcase a learning algorithm which exploits this structure to generalize from data to grammars.

Finite Word Models

'word' is synonymous with 'structure.'

- A model of a word is a representation of it.
- A (Relational) Model contains two kinds of elements.
  - A domain: a finite set of elements.
  - Relations over domain elements.
- Every word has a model.
- Different words have different models.

Finite Word Models

$$\mathbb{W}^{\triangleleft,\triangleleft^+} = \langle \mathcal{D}^{\mathbb{W}}, \triangleleft^{\mathbb{W}}, {\triangleleft^+}^{\mathbb{W}}, P_\sigma^{\mathbb{W}} \rangle_{\sigma \in \Sigma}$$

$$\mathbb{W}^{\triangleleft} = \langle \mathcal{D}^{\mathbb{W}}, \triangleleft^{\mathbb{W}}, P_\sigma^{\mathbb{W}} \rangle_{\sigma \in \Sigma} \qquad \mathbb{W}^{\triangleleft^+} = \langle \mathcal{D}^{\mathbb{W}}, {\triangleleft^+}^{\mathbb{W}}, P_\sigma^{\mathbb{W}} \rangle_{\sigma \in \Sigma}$$

$\mathcal{D}^{\mathbb{W}}$ — Finite set of elements (positions)

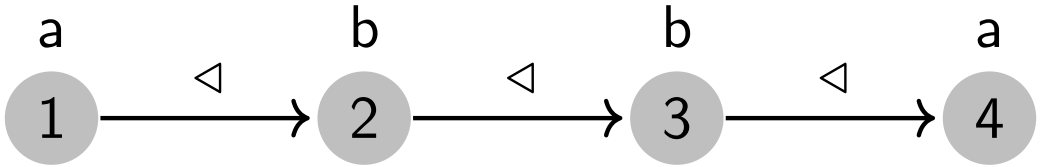
$\triangleleft^{\mathbb{W}}$ — immediate linear precedence on $\mathcal{D}$

${\triangleleft^+}^{\mathbb{W}}$ — (arbitrary) linear precedence on $\mathcal{D}$

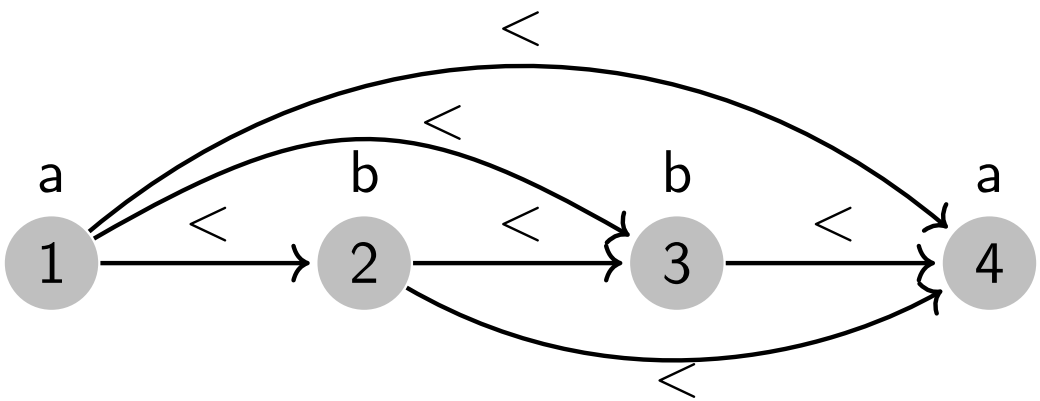
$P_\sigma^{\mathbb{W}}$ — Subset of $\mathcal{D}$ at which σ occurs

Finite Word Models

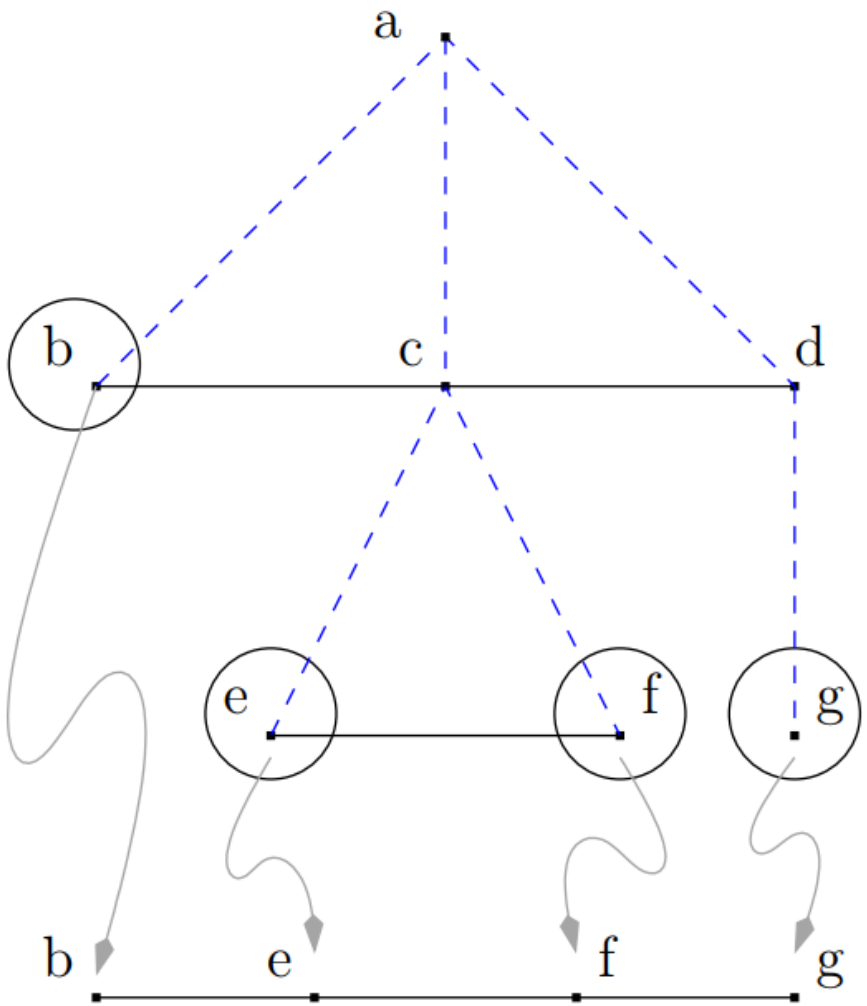
1. Successor (Immediate Precedence)



2. General precedence

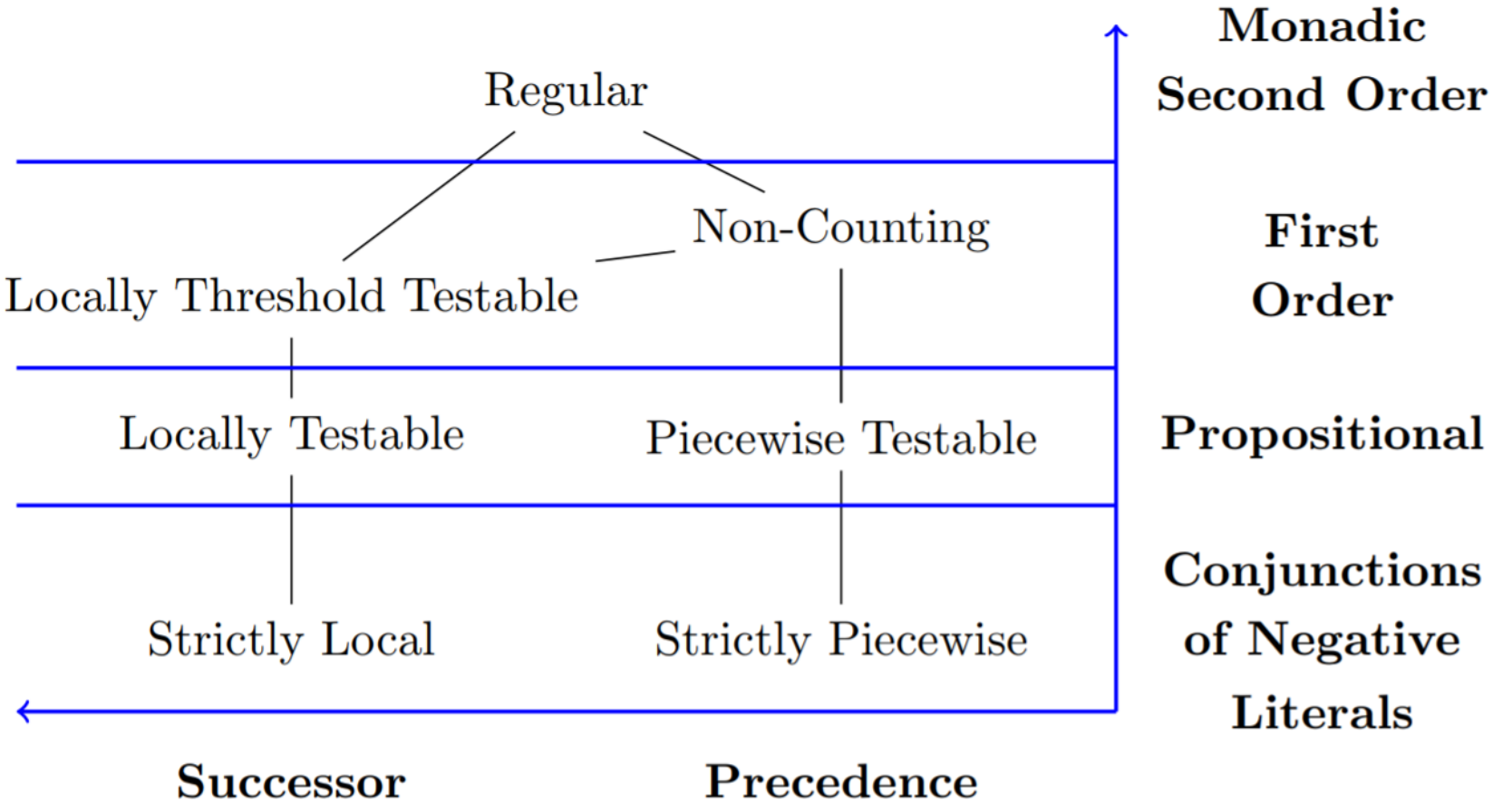


Tree Models (Rogers 2003)

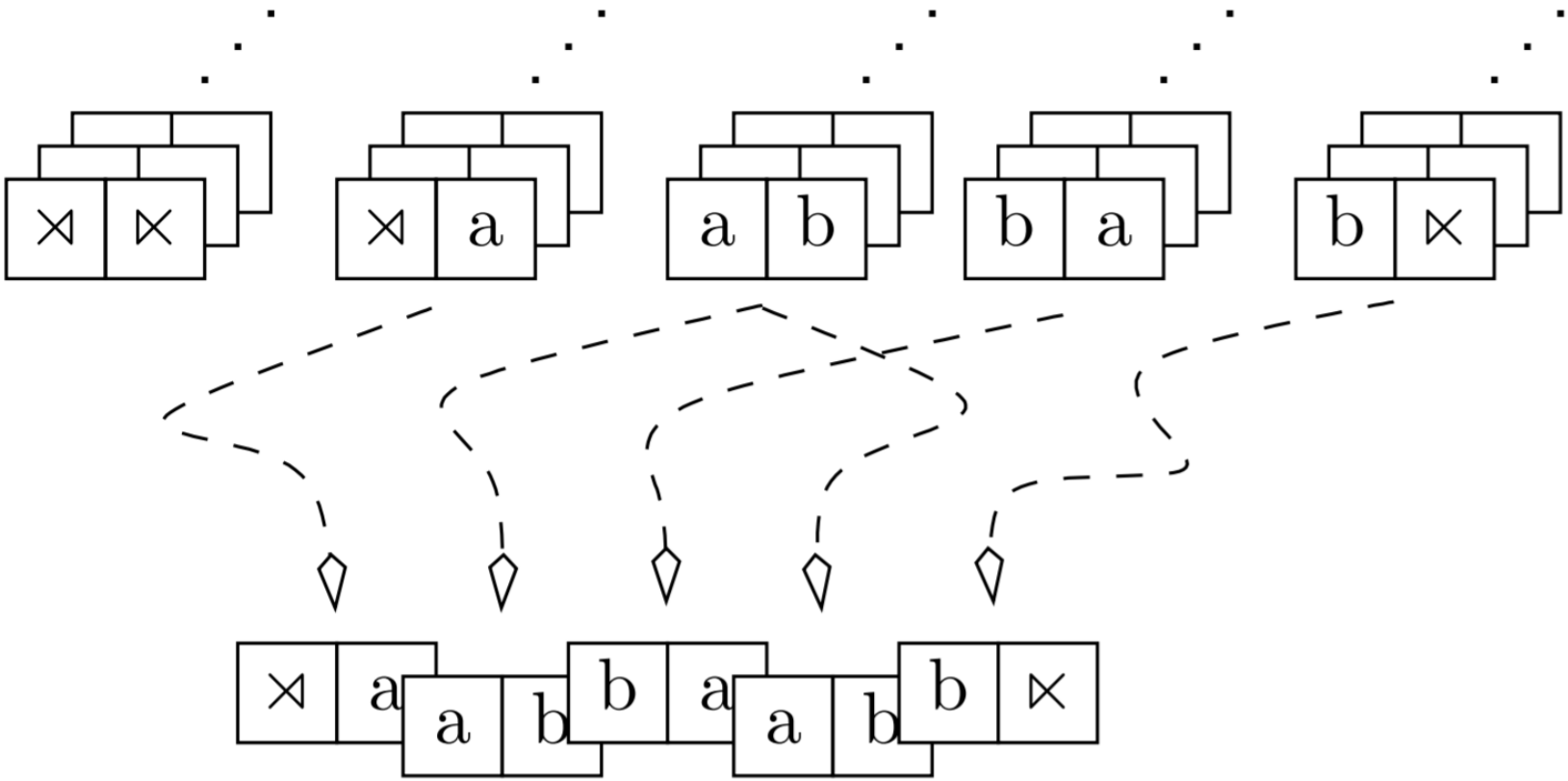


Pic courtesy of Rogers 2014 ESSLLI course.

Subregular Hierarchy (Rogers et al 2013)



Local Factors



Pics courtesy of Heinz and Rogers 2014 ESSLLI course.

Locality and Projection

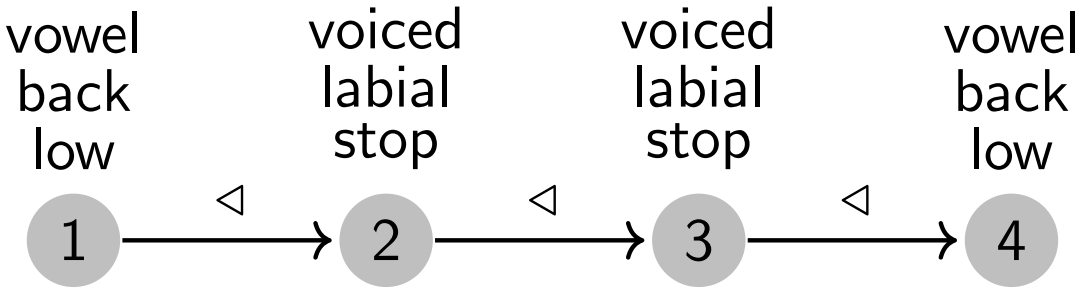
Theorem (Medvedev) *A set of strings is Regular iff it is a homomorphic image of a Strictly 2-Local set.*

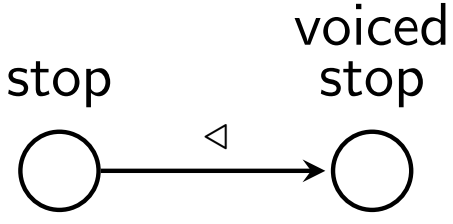
Theorem (Thatcher) *A set of Σ-labeled trees is recognizable by a finite-state tree automaton (i.e. regular) iff it is a projection of a local set of trees.*

Theorem (Thatcher) *A set of strings L is the yield of a local set of trees (equivalently, is the yield of a recognizable set of trees) iff it is Context-Free.*

Unconventional Word Models

Successor (Immediate Precedence)





The Challenge of Features

Distinctive Feature Theory

"part of the heart of phonology" — Rice (2003)
"The most fudamental insight gained during the last century"
— Ladefoged & Halle (1988)

$$\text{*NT} \rightarrow \{ \text{*nt, *np, *nk, *mt, *mp, *mk, } \ldots \}$$

Wilson & Gallagher 2018

"Could there be a non-statistical model that learns by memorizing feature sequences? The problem confronting such a model is that any given segment sequence has may different featural representations. Without a method for deciding which representations are relevant for assessing wellformedness (the role that statistics plays in Maxent) **learning is doomed**."

The Challenge of Features

Distinctive Feature Theory

"part of the heart of phonology" — Rice (2003)
"The most fudamental insight gained during the last century"
— Ladefoged & Halle (1988)

$$\text{*NT} \rightarrow \{ \text{*nt, *np, *nk, *mt, *mp, *mk, } \ldots \}$$

Wilson & Gallagher 2018

"Could there be a non-statistical model that learns by memorizing feature sequences? The problem confronting such a model is that any given segment sequence has may different featural representations. Without a method for deciding which representations are relevant for assessing wellformedness (the role that statistics plays in Maxent) **learning is doomed**."

Example

Imagine the sequence *nt* is not present in a corpus. There are many possible equivalent constraints:

*nt
*[+nasal][+coronal]
*[+consonant][+coronal,-continuant]
*[+sonorant][-sonorant]
....

How can a learner decide which of these constraints is responsible for the absence of *nt*?

Example

Imagine the sequence *nt* is not present in a corpus. There are many possible equivalent constraints:

*nt
*[+nasal][+coronal]
*[+consonant][+coronal,-continuant]
*[+sonorant][-sonorant]
....

How can a learner decide which of these constraints is responsible for the absence of *nt*?

Example

Imagine the sequence *nt* is not present in a corpus. There are many possible equivalent constraints:

*nt
*[+nasal][+coronal]
*[+consonant][+coronal,-continuant]
*[+sonorant][-sonorant]
....

How can a learner decide which of these constraints is responsible for the absence of *nt*?

Constraint Explosion (Hayes and Wilson 2008)

As we add segments and features, the amount of possible hypotheses grows larger. How much larger?

Table 2
Number of possible constraints for various values of $|C|$ and n

		$\|C\|$ = 30	100	200	400
	1	30	100	200	400
	2	900	10,000	40,000	160,000
n	**3**	27,000	1,000,000	8 million	64 million
	4	810,000	100 million	1.6 billion	26 billion
	5	24 million	10 billion	320 billion	10 trillion

Some definitions

Definition (Restriction)

$A = \langle D^A; \succ, R_1^A, \ldots, R_n^A \rangle$ *is a restriction of* $B = \langle D^B; \succ, R_1^B, \ldots, R_n^B \rangle$ iff $D^A \subseteq D^B$ and for each m-ary relation R_i, we have $R_i^A = \{(x_1 \ldots x_m) \in R_i^B \mid x_1, \ldots, x_m \in D^A\}$.

Intuition: Identifies a subset A of the domain of B and strips B of all elements and relations which are not wholly within A.

Some definitions

Definition (Subfactor)

Structure A is a *subfactor* of structure B ($A \sqsubseteq B$) if A is connected, there exists a restriction of B denoted B', and there exists $h : A \rightarrow B'$ such that for all $a_1, \ldots a_m \in A$ and for all R_i in the model signature: if $h(a_1), \ldots h(a_m) \in B'$ and $R_i(a_1, \ldots a_m)$ holds in A then $R_i(h(a_1), \ldots h(a_m))$ holds in B'. If $A \sqsubseteq B$ we also say that B is a *superfactor* of A.

Intuition: Properties that hold of the connected structure A also hold in a related way within B.

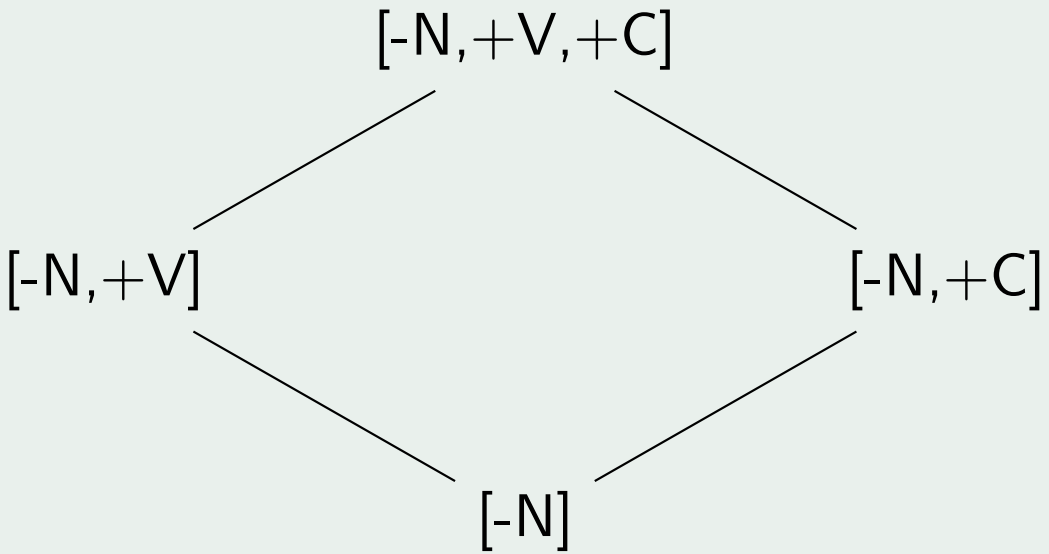
Subfactor Ideals

Definition (Ideal)

A non-empty subset S of a poset $\langle A, \leq \rangle$ is an **ideal** iff

- for every $x \in S$, $y \leq x$ implies $y \in S$, and
- for all $x, y \in S$ there is some $z \in S$ s.t $x \leq z$ and $y \leq z$.

Example

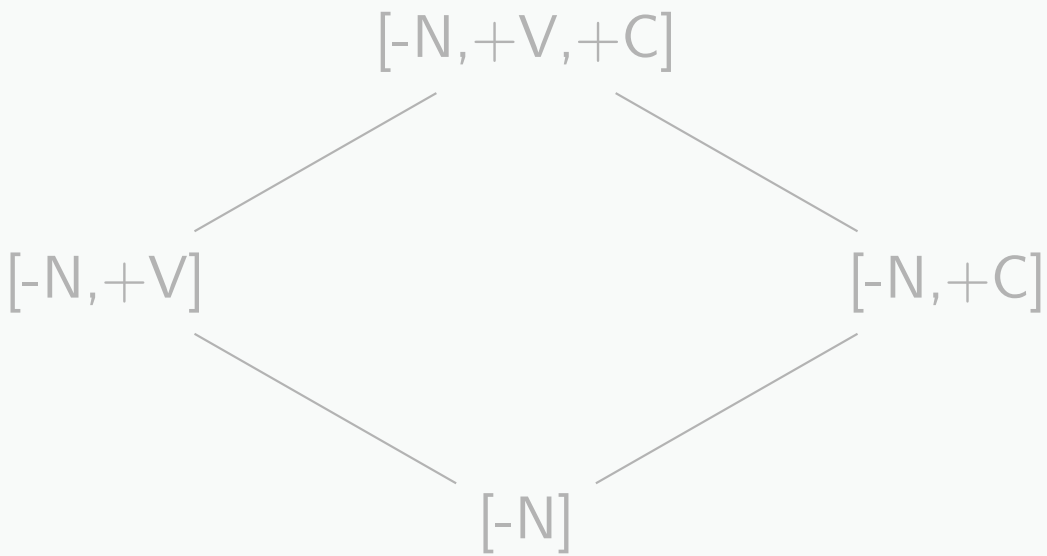


Grammatical Entailments

Subfactor Ideals

If **s** is a subfactor of **t** for G and G generates **t**,
then G generates **s**.

Example

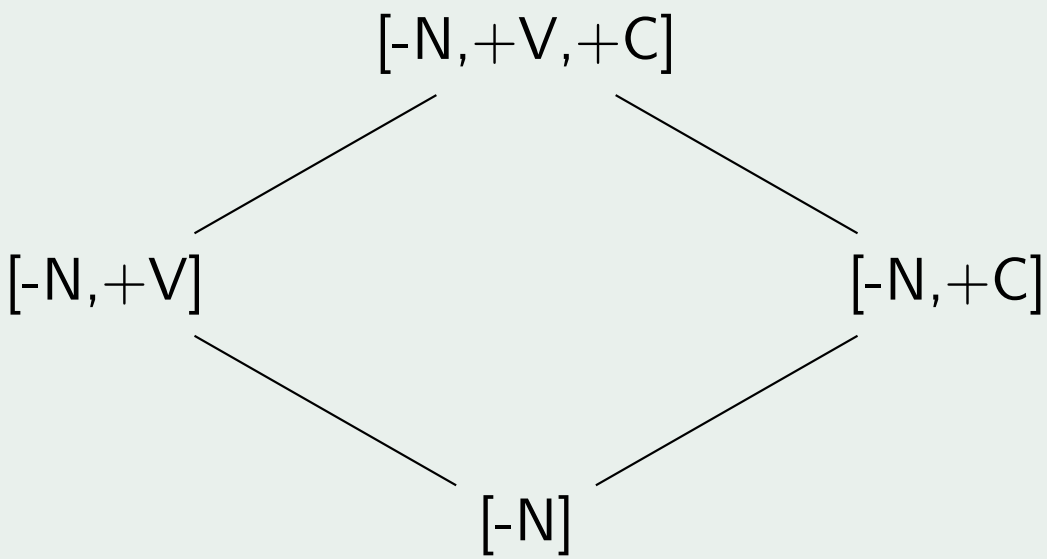


Grammatical Entailments

Subfactor Ideals

If **s** is a subfactor of **t** for G and G generates **t**,
then G generates **s**.

Example

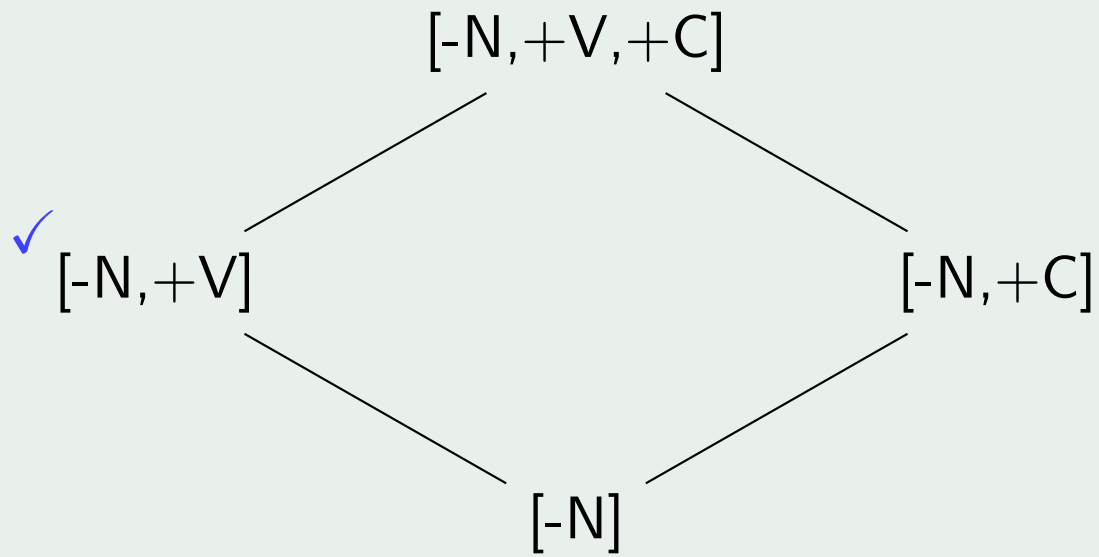


Grammatical Entailments

Subfactor Ideals

If **s** is a subfactor of **t** for G and G generates **t**,
then G generates **s**.

Example

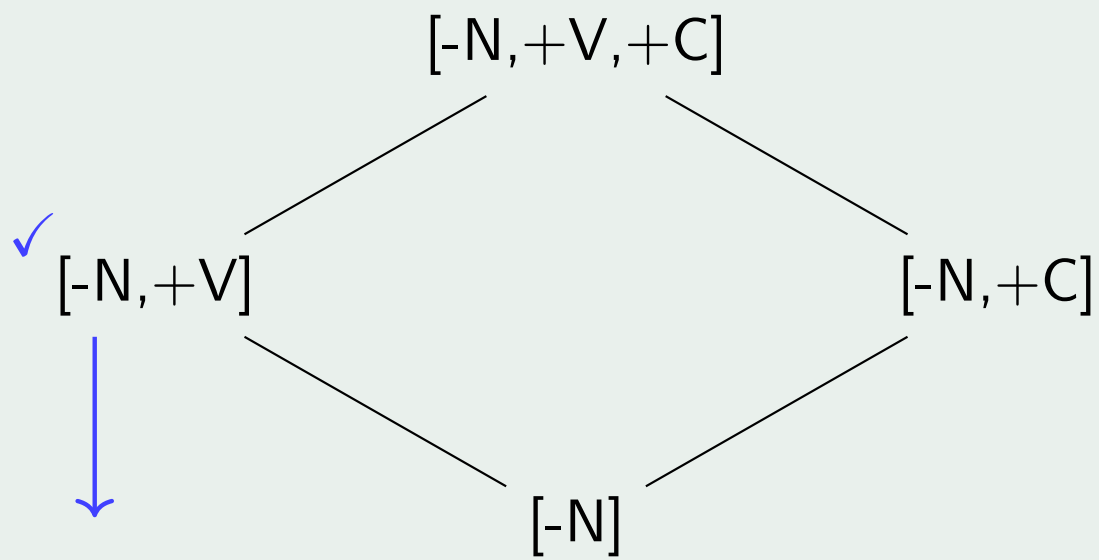


Grammatical Entailments

Subfactor Ideals

If **s** is a subfactor of **t** for G and G generates **t**,
then G generates **s**.

Example

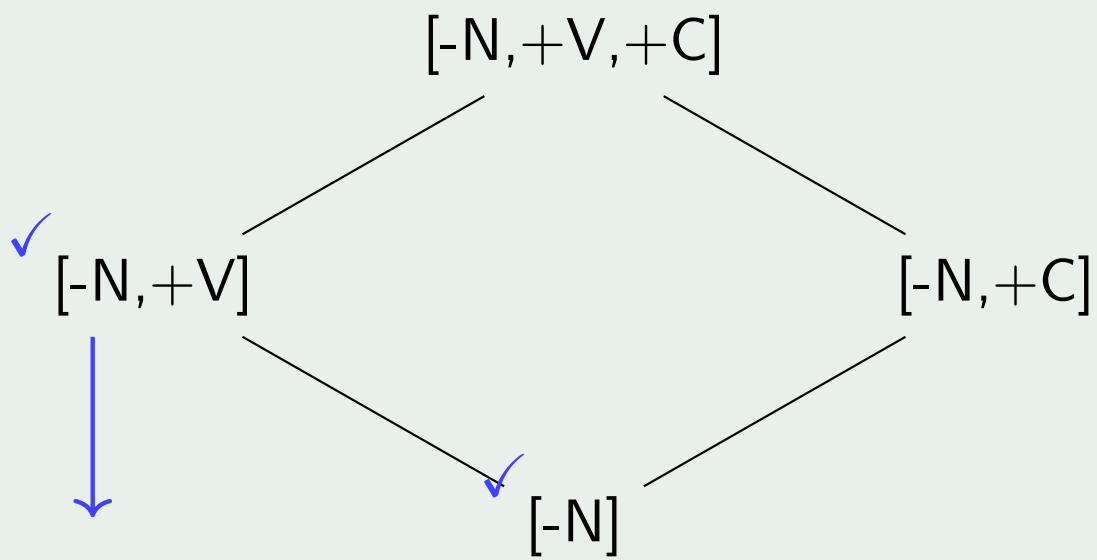


Grammatical Entailments

Subfactor Ideals

If **s** is a subfactor of **t** for G and G generates **t**,
then G generates **s**.

Example

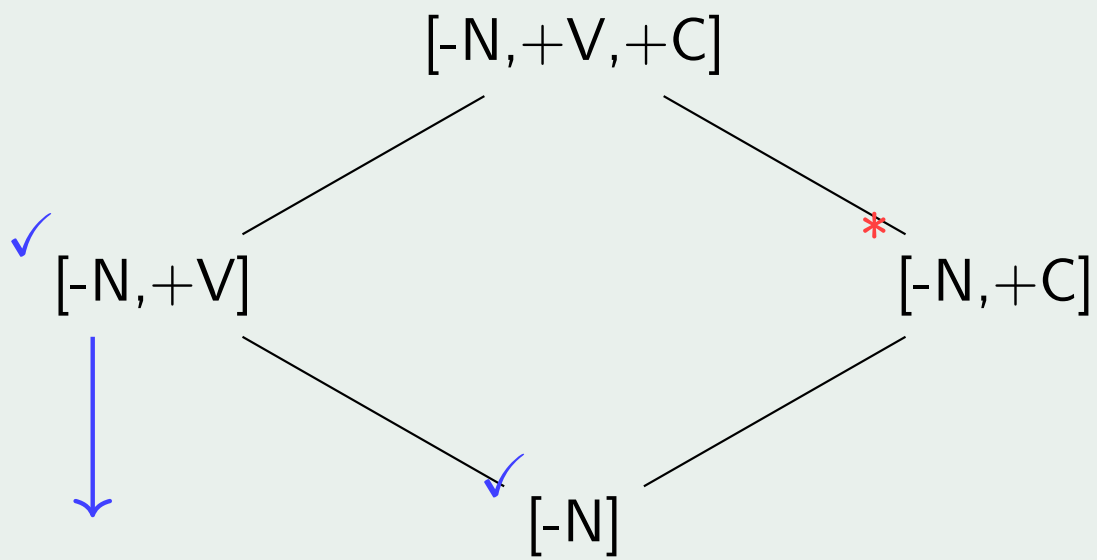


Grammatical Entailments

Subfactor Ideals

If **s** is a subfactor of **t** for G and G generates **t**,
then G generates **s**.

Example

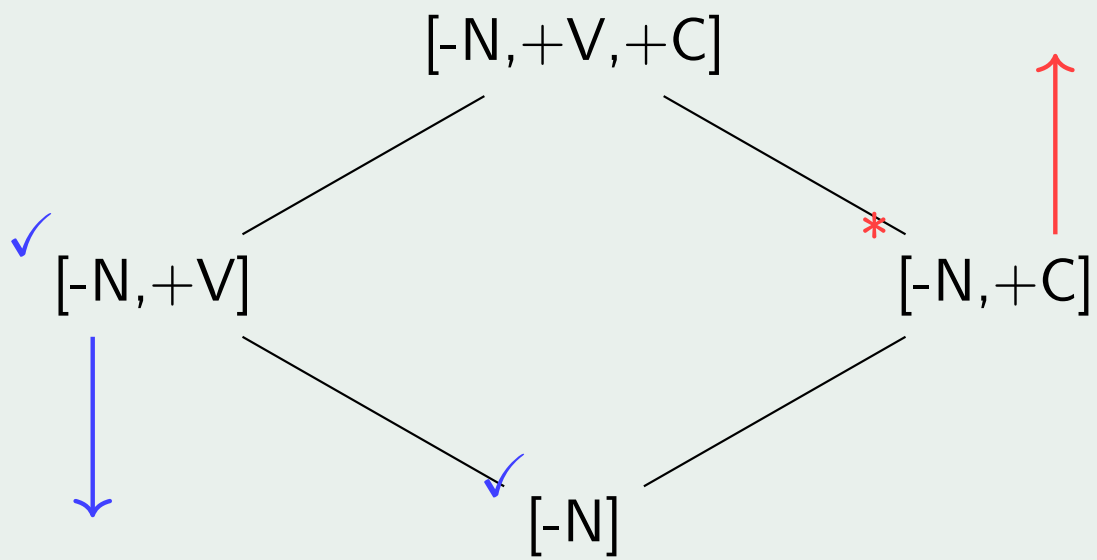


Grammatical Entailments

Subfactor Ideals

If **s** is a subfactor of **t** for G and G generates **t**,
then G generates **s**.

Example

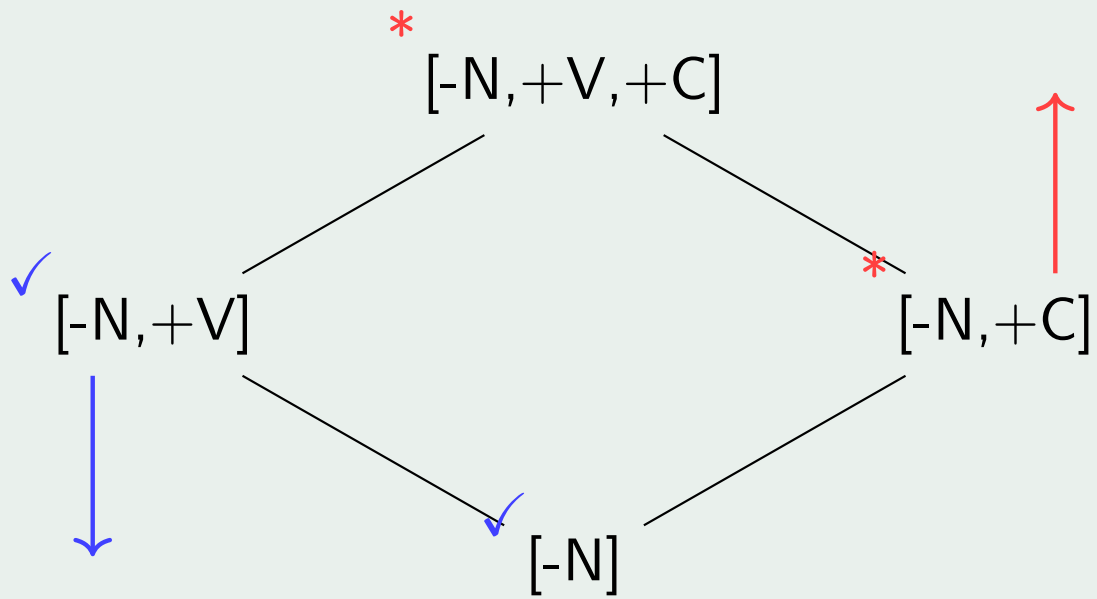


Grammatical Entailments

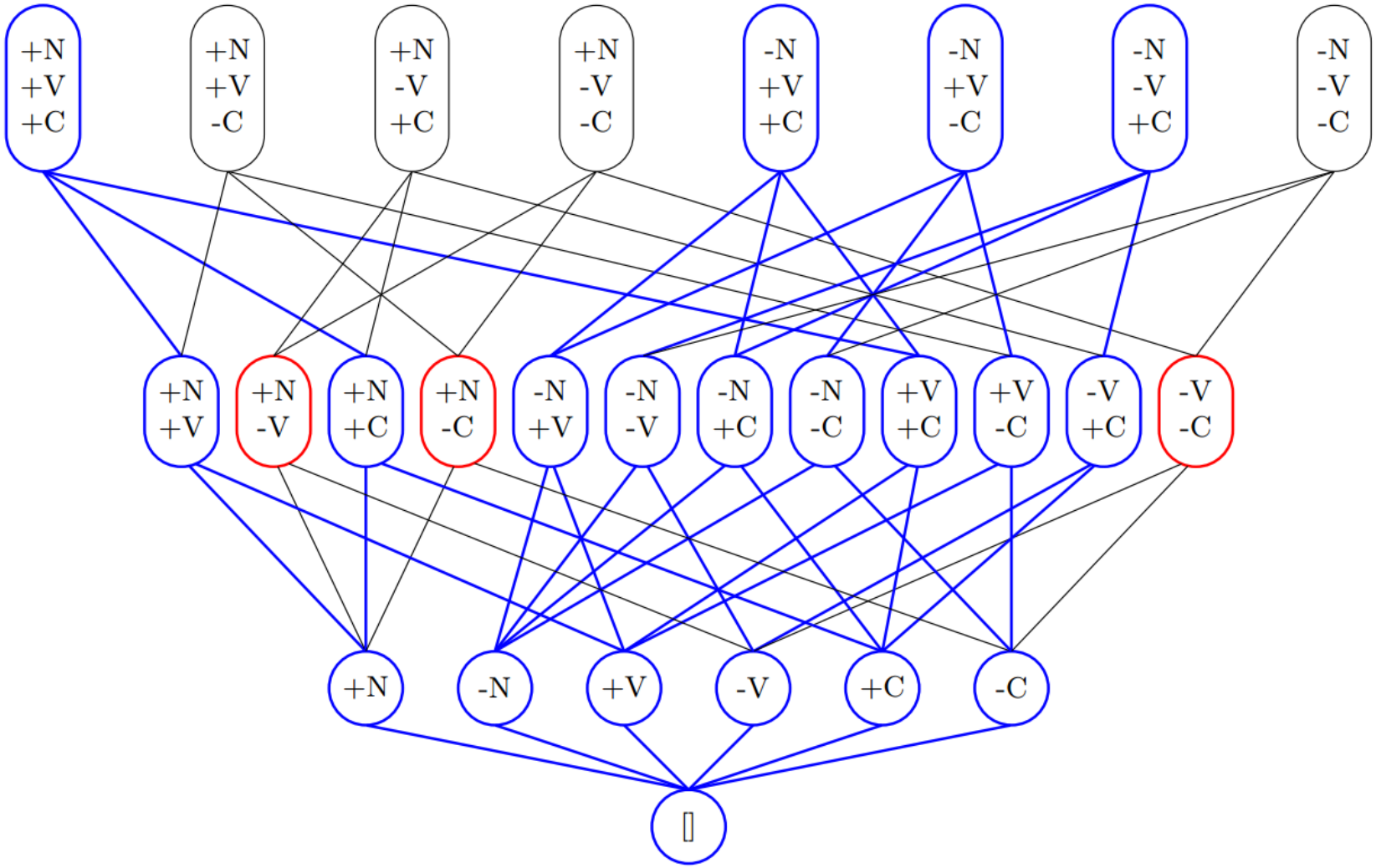
Subfactor Ideals

If **s** is a subfactor of **t** for G and G generates **t**,
then G generates **s**.

Example



Example with Singular Segments



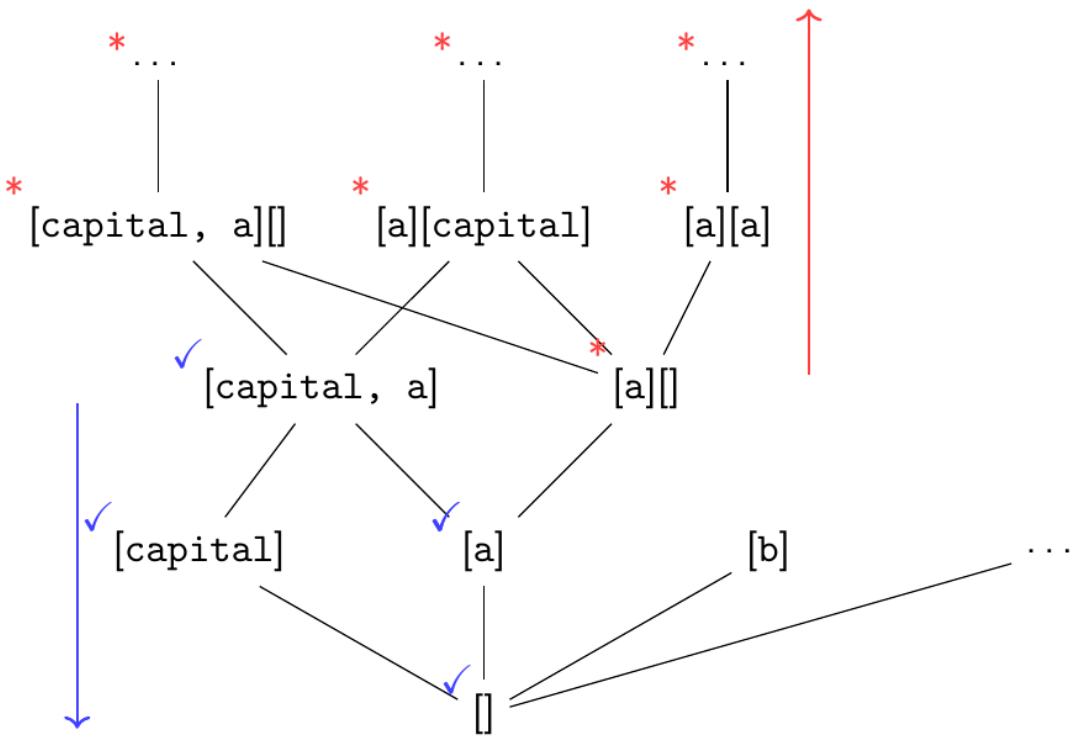
NLP Example

In many NLP applications, text symbols are treated independently

Alphabet $= \{\mathtt{a}, \ldots, \mathtt{z}, \mathtt{A}, \ldots, \mathtt{Z}\} = 52$ symbols

Forbidding maybe all capitals $\rightarrow$ Explosion!

If we use feature [capital], only 27! 26 letters + [capital]



Unbounded Linguistic Dependencies

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

  (1) a. * hasxintilawaʃ
  b. * haʃxintilawas
  c. haʃxintilawaʃ

Example: Samala

*$ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

Unbounded Linguistic Dependencies

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

  (1) a. * hasxintilawaʃ
  b. * haʃxintilawas
  c. haʃxintilawaʃ

Example: Samala

*$ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

Unbounded Linguistic Dependencies

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

  (1) a. * hasxintilawaʃ
  b. * haʃxintilawas
  c. haʃxintilawaʃ

Example: Samala

* $ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

Unbounded Linguistic Dependencies

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

  (1) a. * hasxintilawaʃ
  b. * haʃxintilawas
  c. haʃxintilawaʃ

Example: Samala

* $ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

Unbounded Linguistic Dependencies

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

  (1) a. * hasxintilawaʃ
  b. * haʃxintilawas
  c. haʃxintilawaʃ

Example: Samala

*$ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

- **But:** Sibilants can be arbitrarily far away from each other!

*$ s t a j a n o w o n w a ʃ $

Unbounded Linguistic Dependencies

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

  (1) a. * hasxintilawaʃ
  b. * haʃxintilawas
  c. haʃxintilawaʃ

Example: Samala

* $ h a s x i n t i l a w a ʃ $

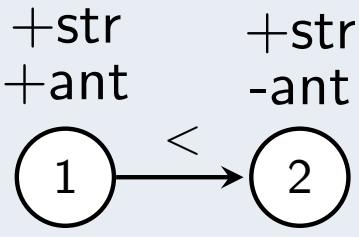
$ h a ʃ x i n t i l a w a ʃ $

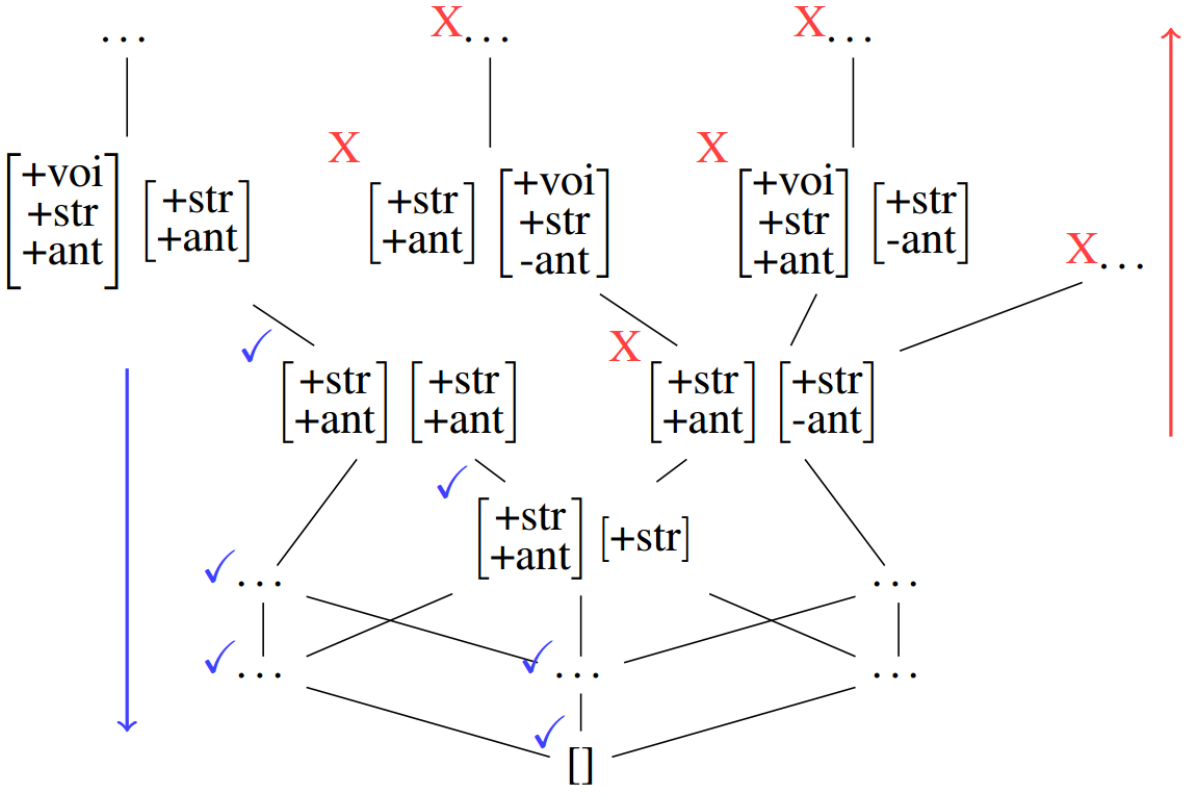
- **But:** Sibilants can be arbitrarily far away from each other!

* $ s t a j a n o w o n w a ʃ $

Example: Samala Long-Distance *sʃ

Banned Structure





Two Ways to Learn (De Raedt 2008)

Specific-to-General Induction

- Start at the most specific points (highest) in the space
- Remove all the subfactors that are present in the data.
- Collect the most general substructures remaining.

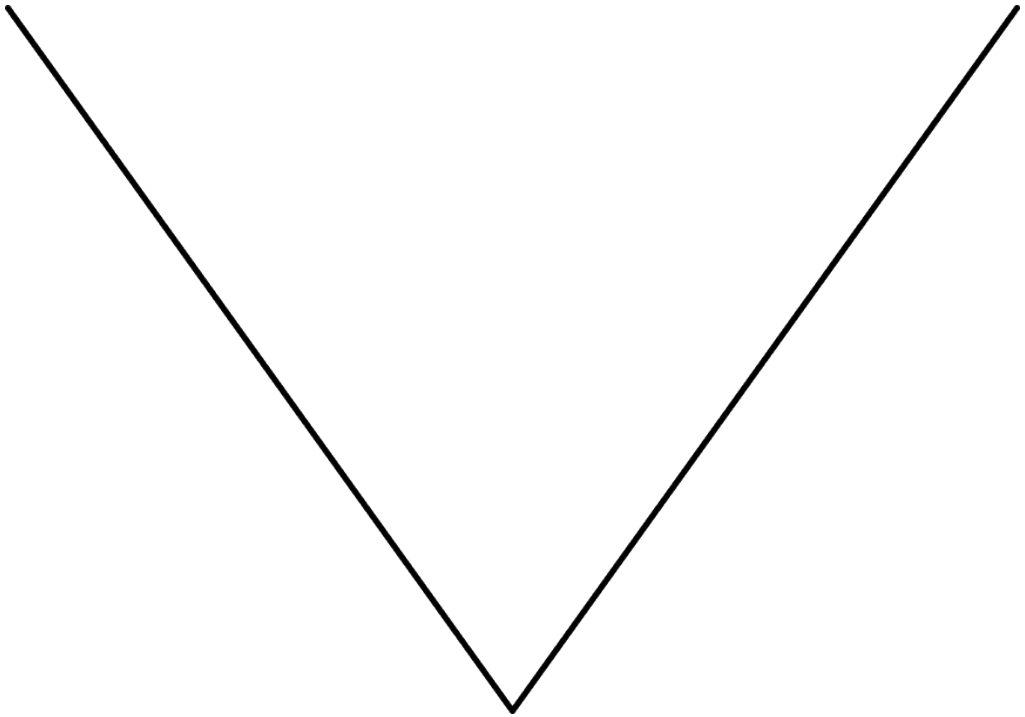
General-to-Specific Induction

- Beginning at the lowest element in the space,
- Check whether this structure is a subfactor of the input data.
- If no, extend the structure by either adding a domain element, or a relation on an existing element and repeat.

Learning Algorithm

Bottom-Up Relational Learner

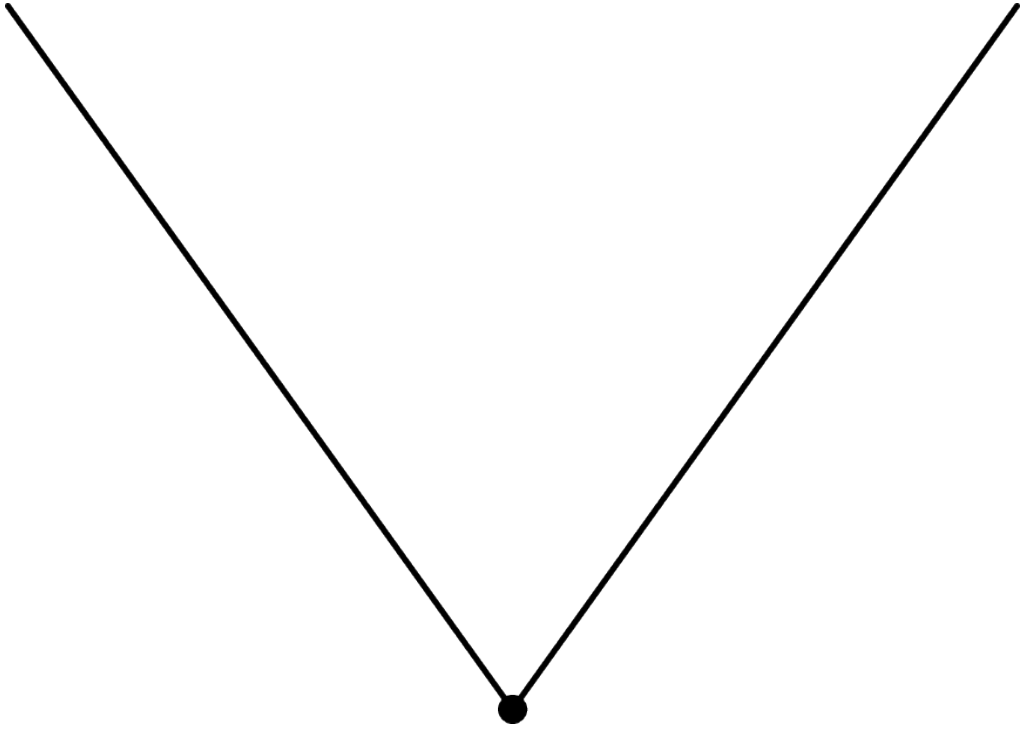
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

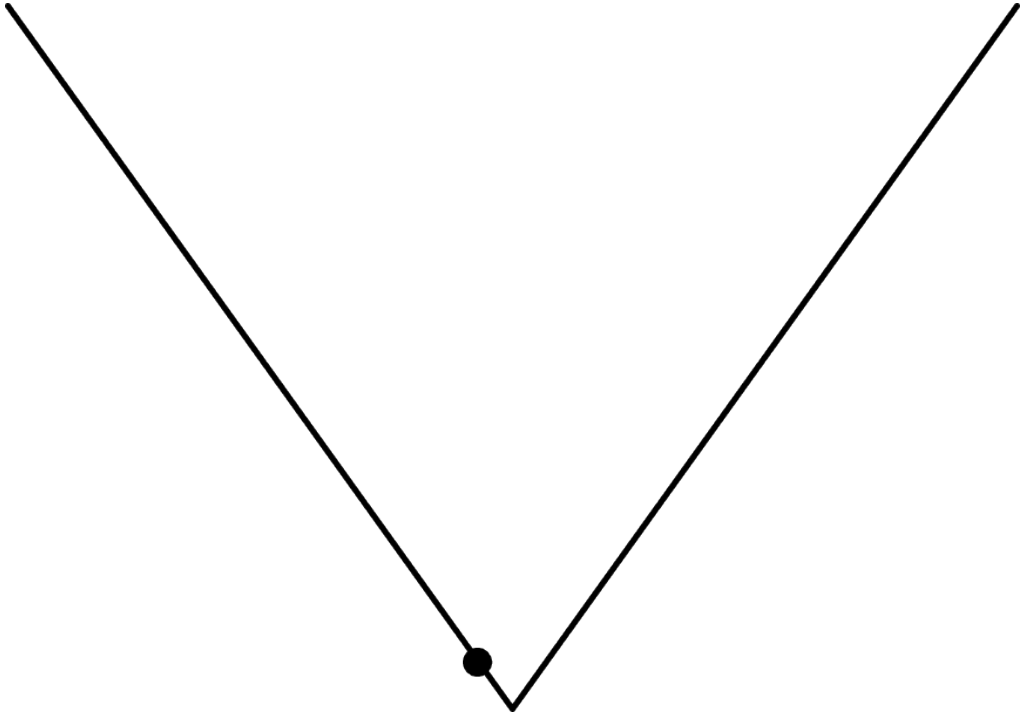
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

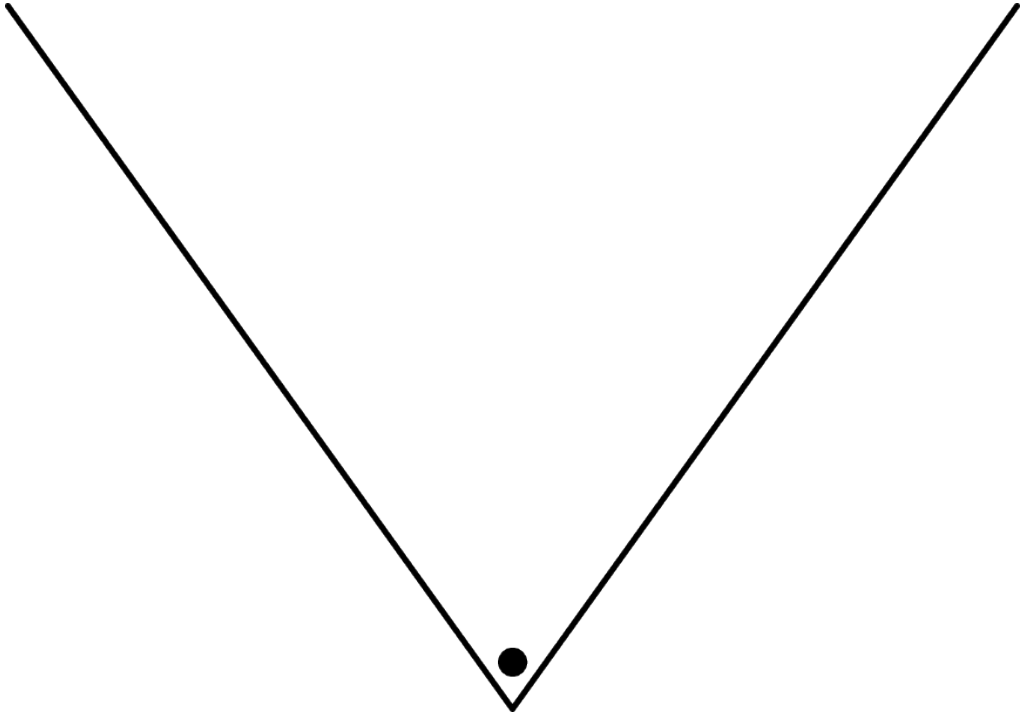
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

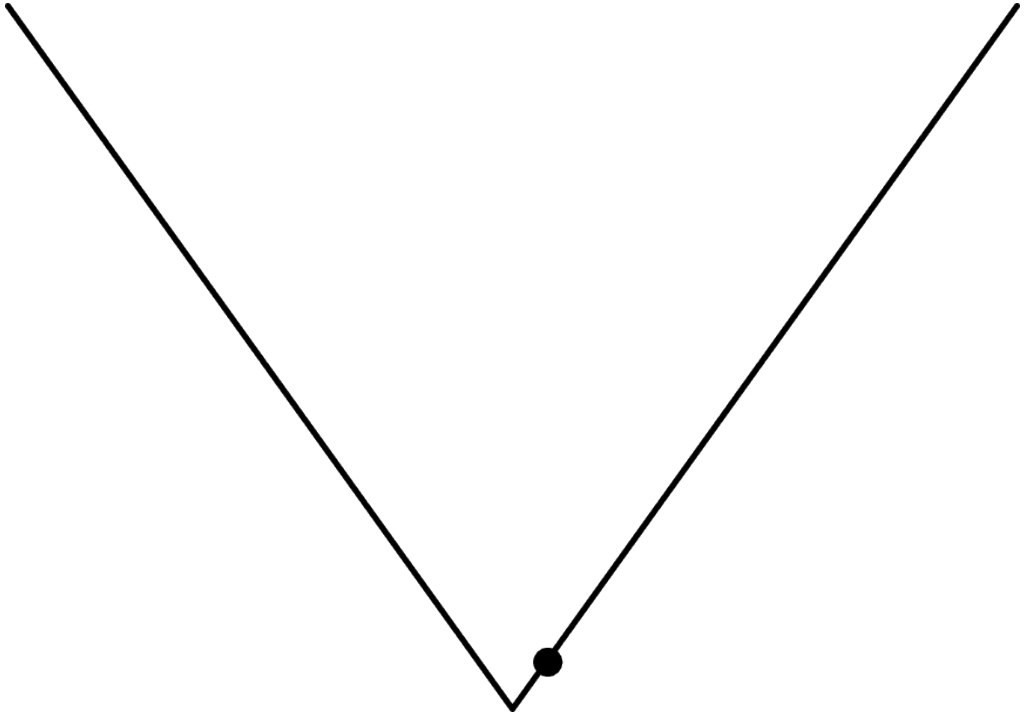
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

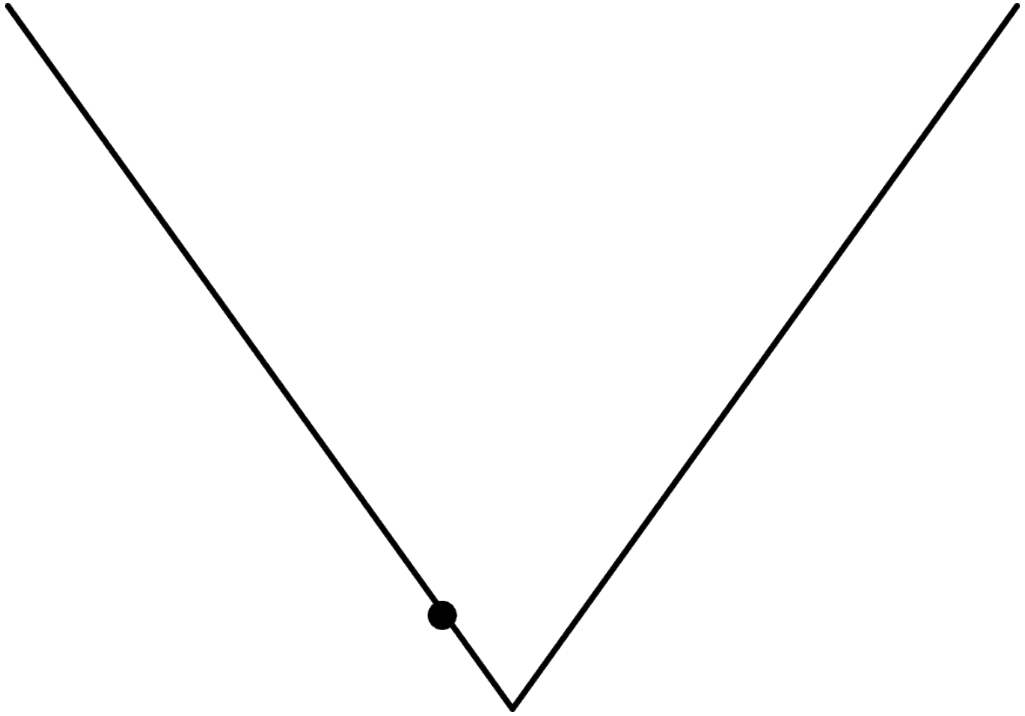
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

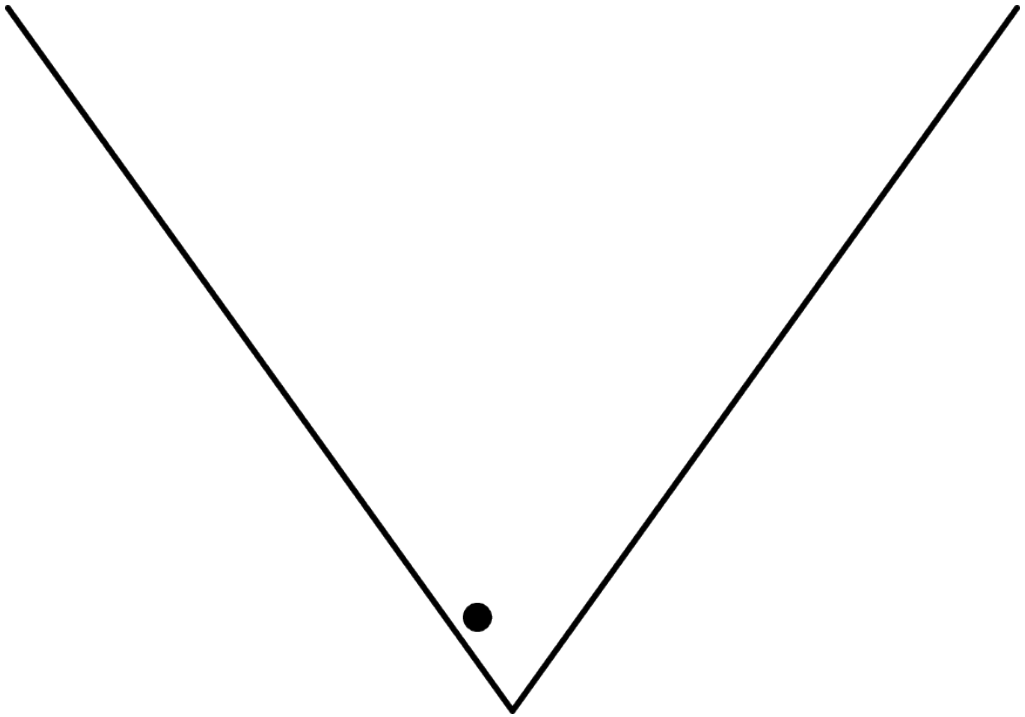
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

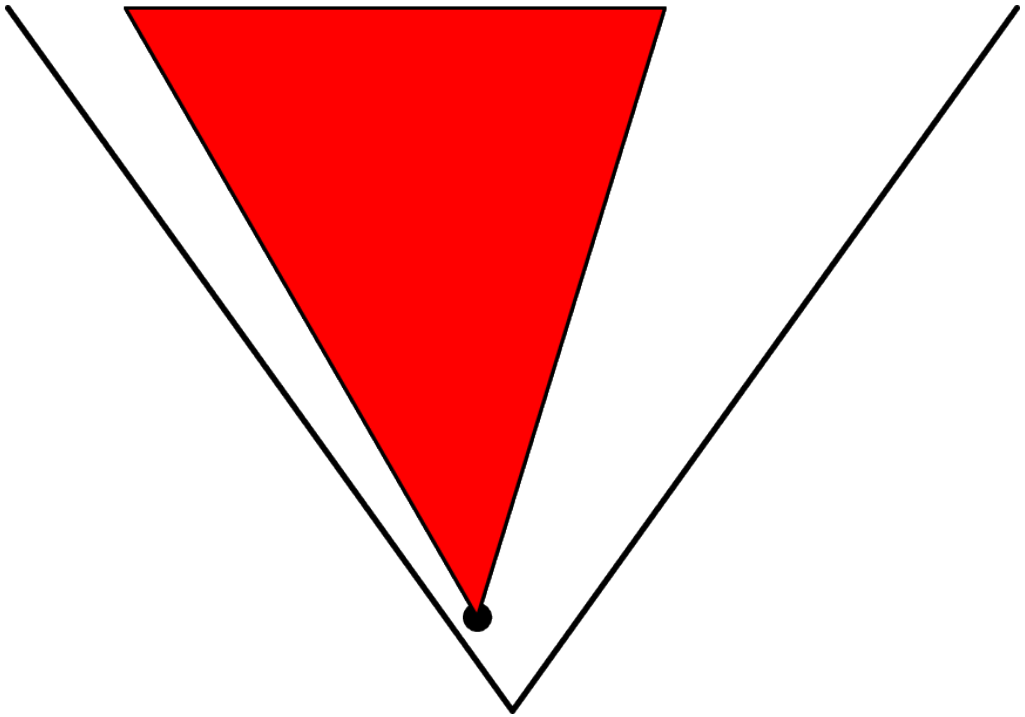
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

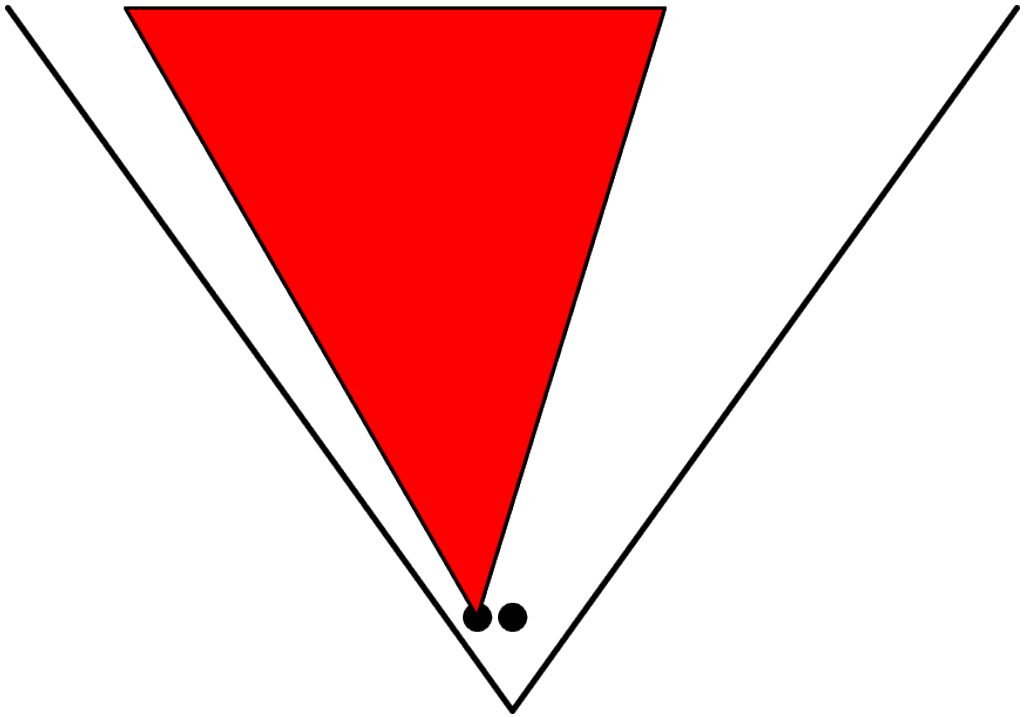
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

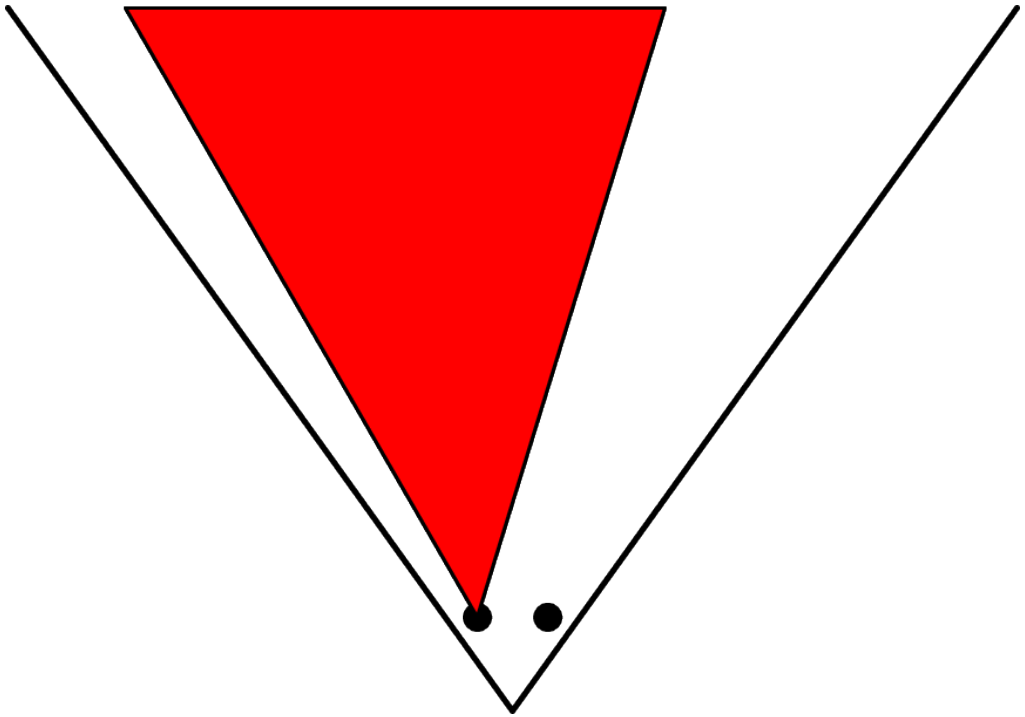
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

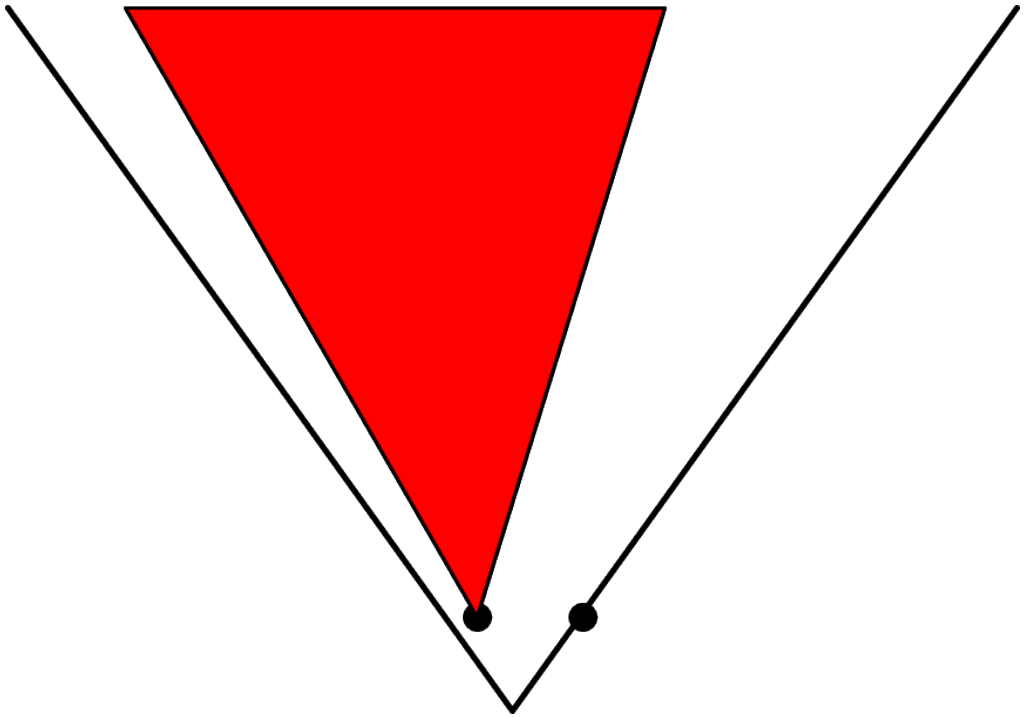
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

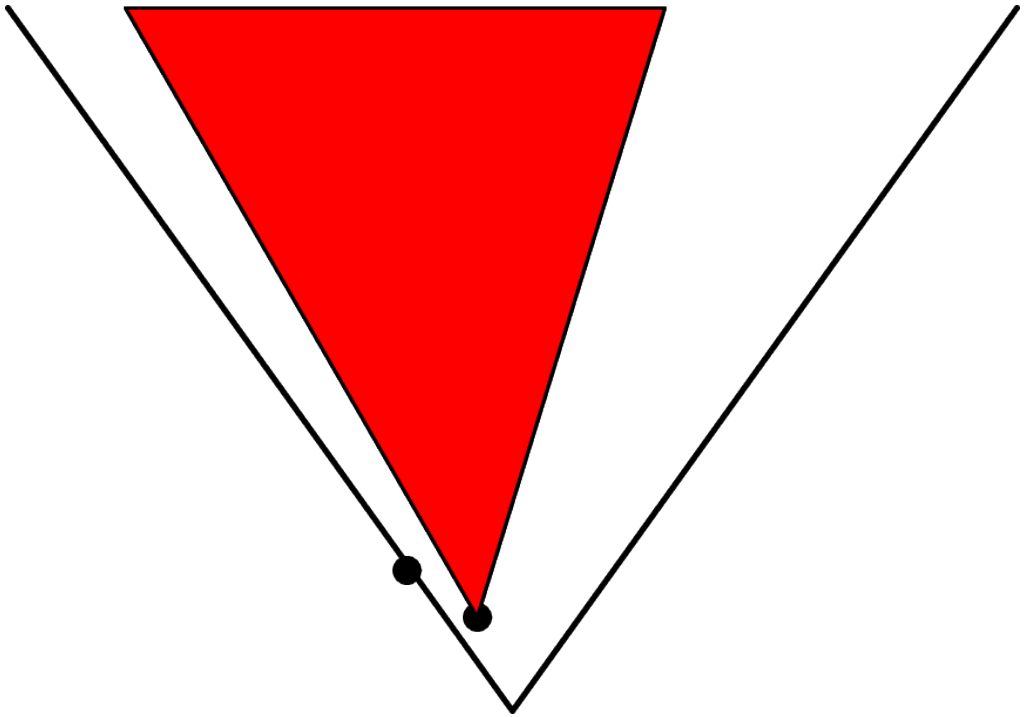
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

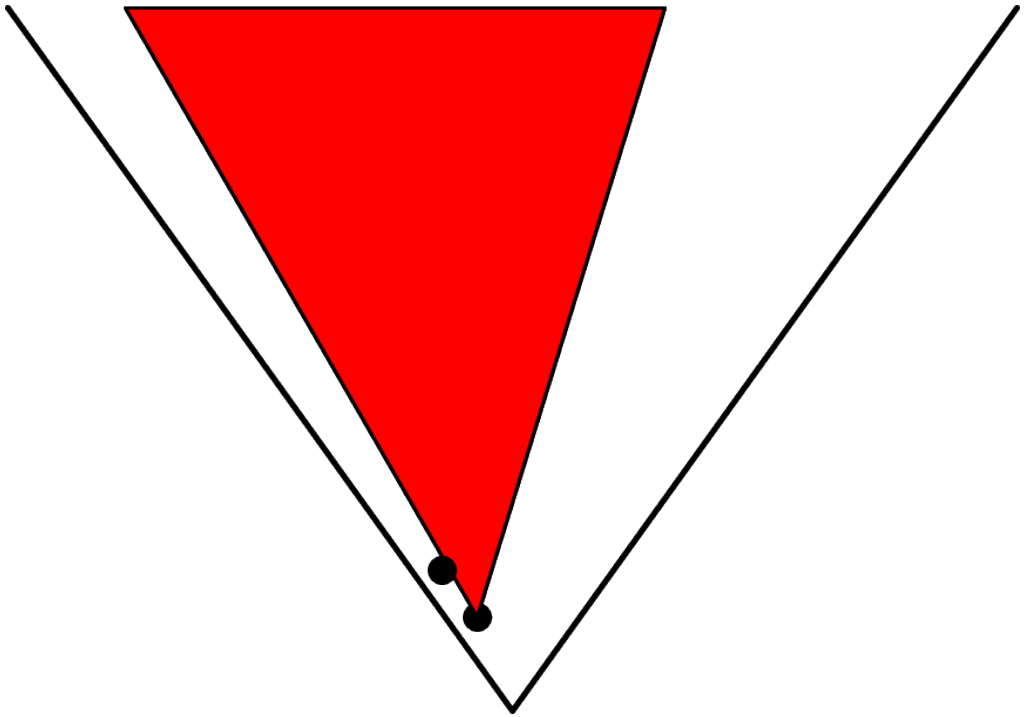
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

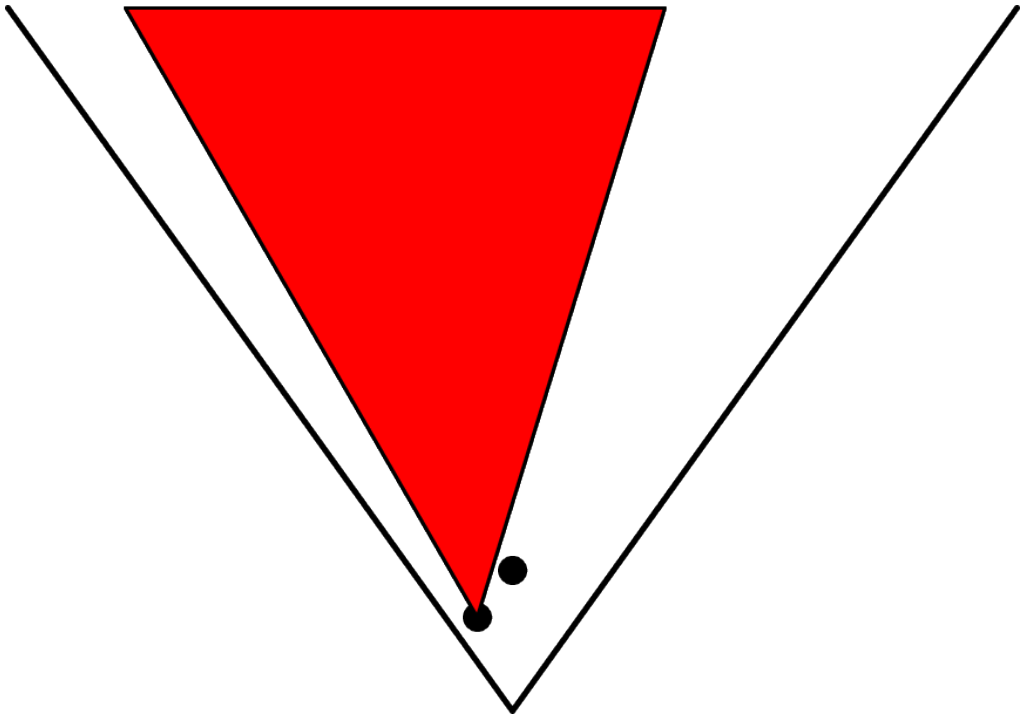
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

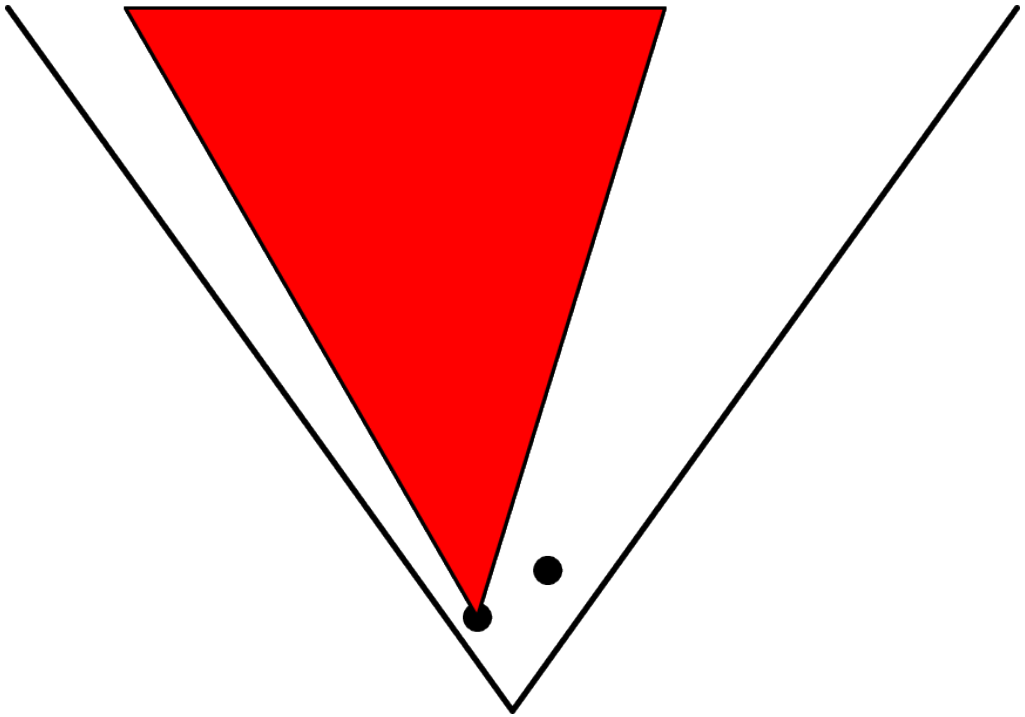
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

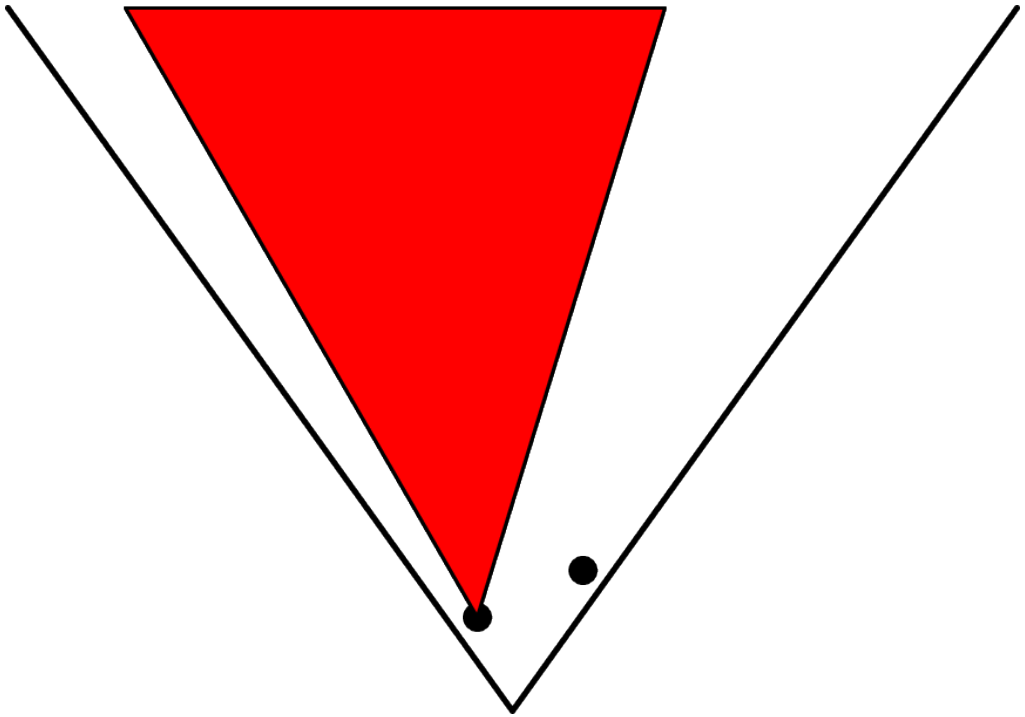
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

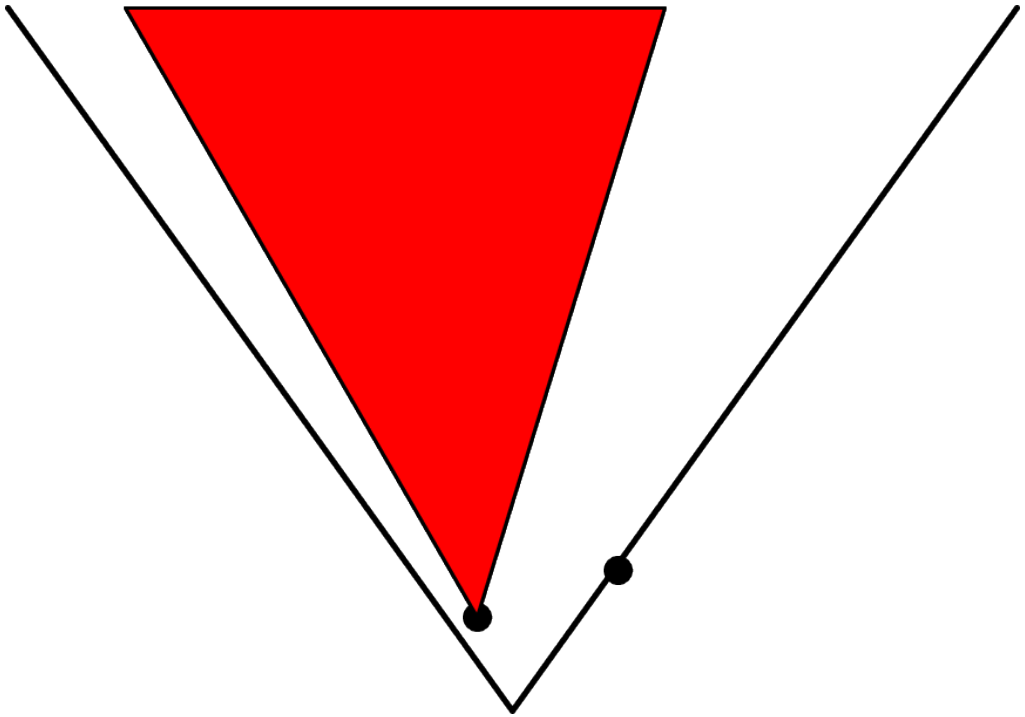
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

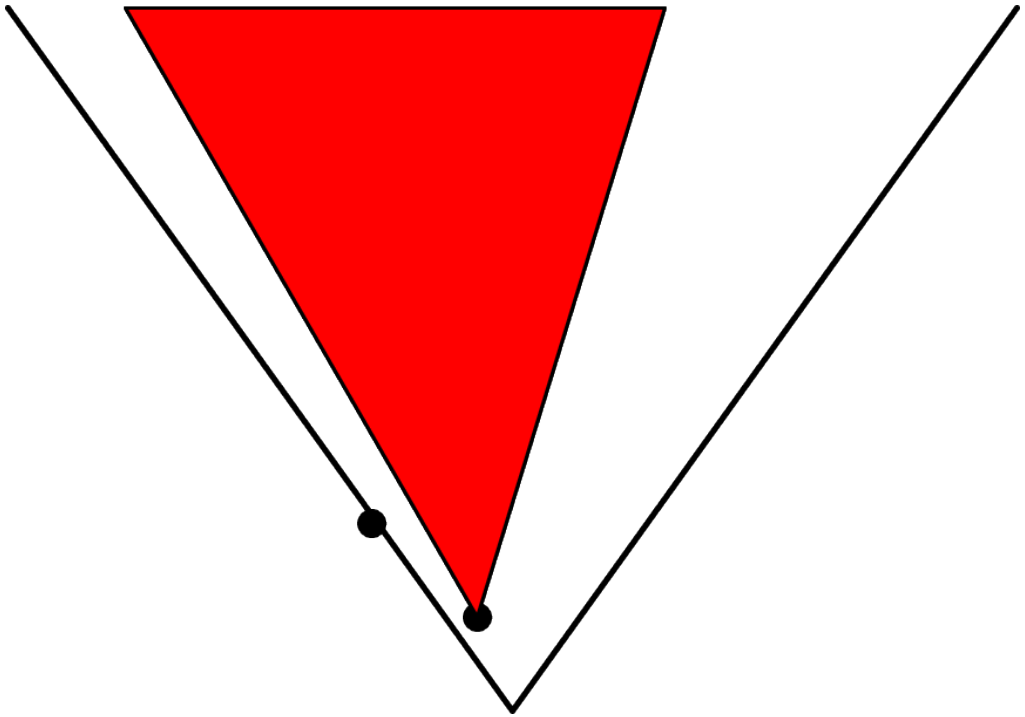
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

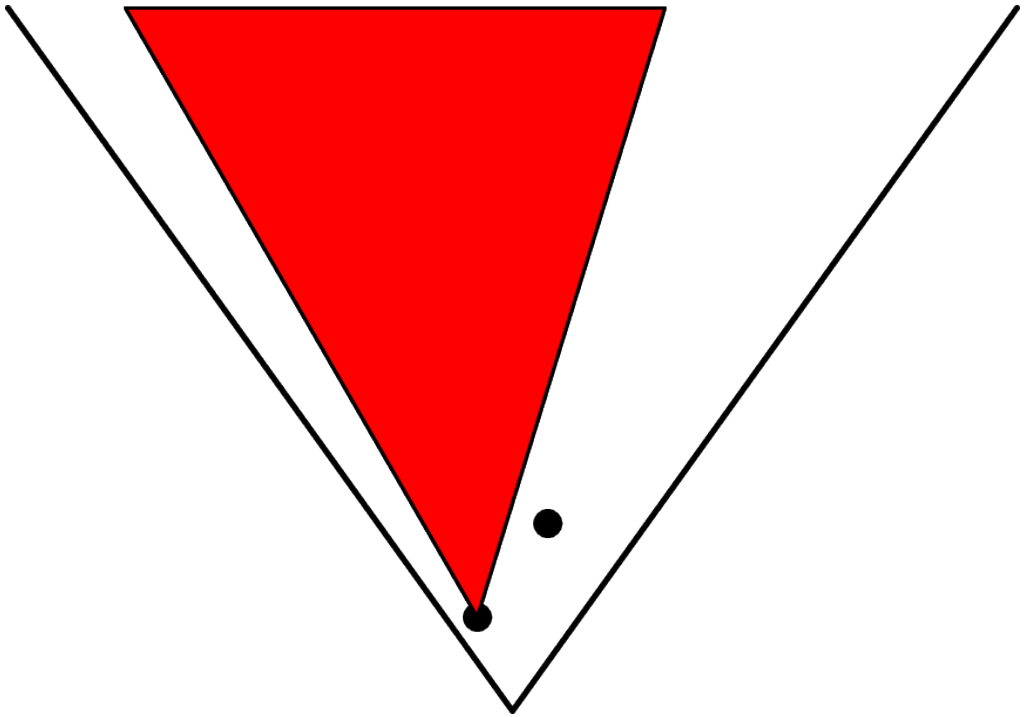
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

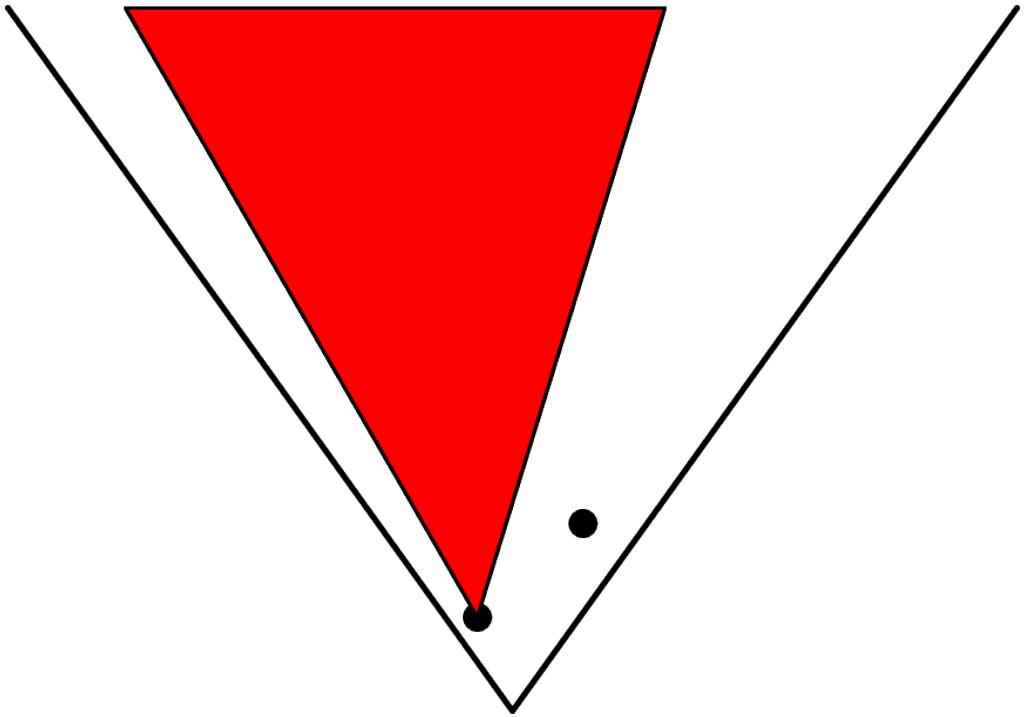
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

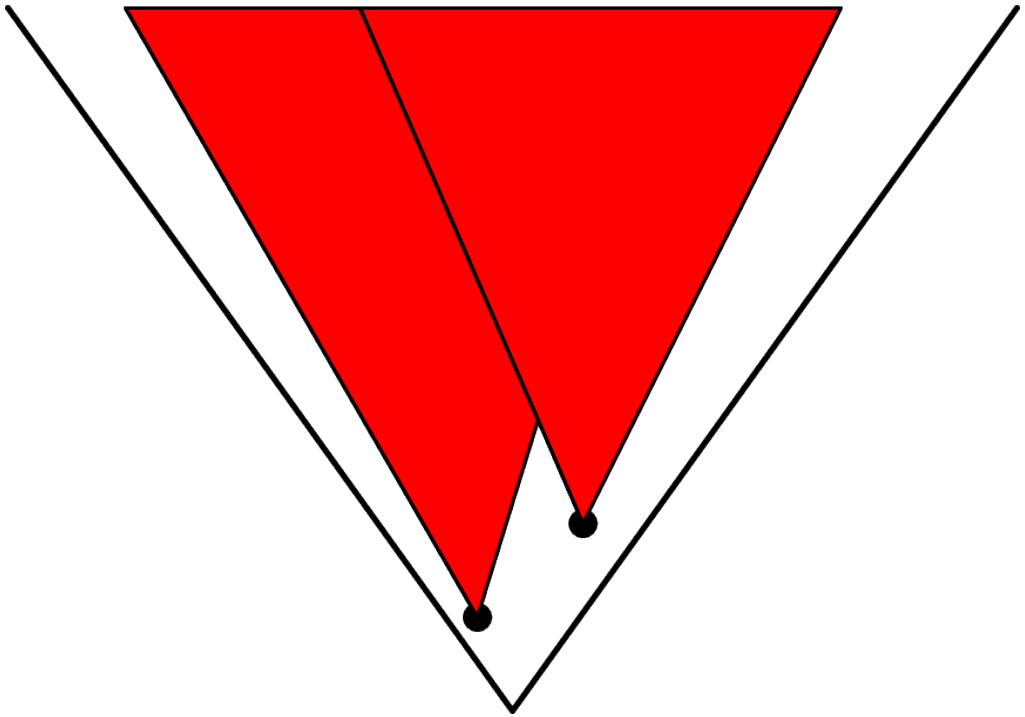
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

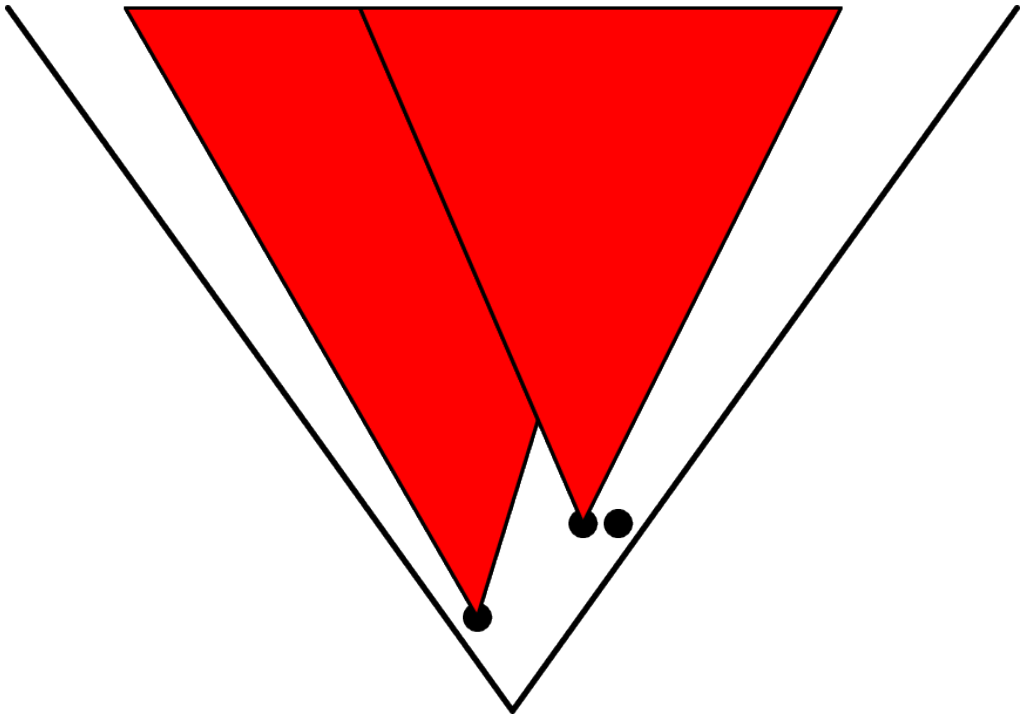
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

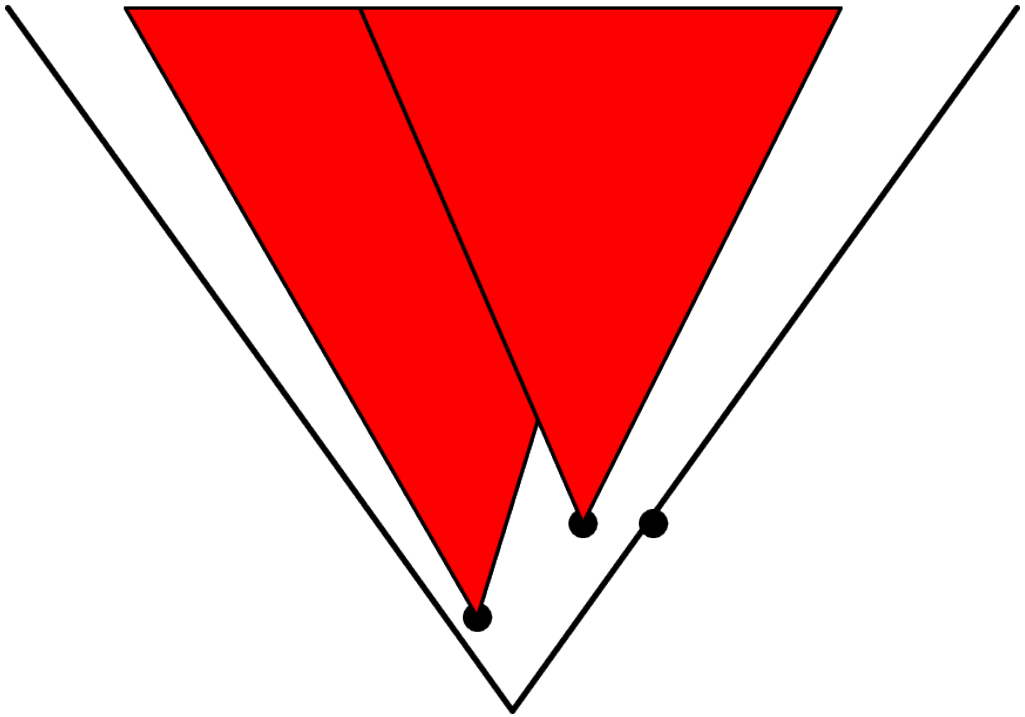
- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Algorithm

Bottom-Up Relational Learner

- Prunes Hypothesis space according to ordering relation
- Provably identifies correct constraints for sequential data
- Uses data sparsity to its advantage!



Learning Guarantees

This learner is provably guaranteed to find the responsible constraints. With What measures?

Theorem

Given a finite positive data sample, the bottom-up learner finds a constraint grammar G such that:

1. G *is consistent, i.e. it covers the data:*
   - $D \subseteq L(G)$
2. $L(G)$ *is the smallest language in* $\mathscr{L}$ *which covers the data*
   - *for all* $L \in \mathscr{L}$ *where* $D \subseteq L$, $L(G) \subseteq L$
3. *the largest forbidden substructure is of size* k
4. G *includes structures* S *that are restrictions of structures* S' *included in other grammars* G' *that also satisfy (1,2,3)*
   - *for all* $S' \in G'$, *there exists* $S \in G$ *such that* $S \sqsubseteq S'$.

Extensions

Things To Do

- Determine the trade-off between data sparsity and time complexity. We hypothesize sparser data should yield faster generalization.
- Extend algorithm to learning subregular functions.
- Incorporation/Comparison to Statistical Learning
  - what is the efficiency tradeoff between statistics and structure?
  - MaxEnt models perform well, can they accommodate structure?

Conclusion

Today's Results

- Learning is due to representations and structured hypothesis spaces
- There is rich structure in features that partially orders the hypothesis space
- These entailments allow bottom-up inference of collections of constraint ideals and filters to succeed

Thanks!

Special thanks to **Jim Rogers** for immensely helpful discussions

This work was supported by NIH under grant #R01HD87133-01

References I

Applegate, R.B. 1972. *Ineseno chumash grammar*. Doctoral Dissertation, University of California,Berkeley.