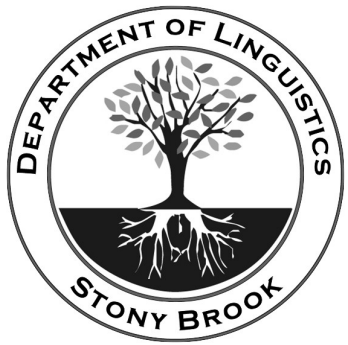
Computational Phonology - Class 1

Jeffrey Heinz (Instructor)
Jon Rawski (TA)



Stony Brook University

LSA Summer Institute
UC Davis
June 24, 2019

Today

1. Organizational Preliminaries
2. What is Phonology?
3. Intensional and Extensional Descriptions
4. Questions in Computational Phonology
5. What are Strings?
6. Models of Strings
7. Overview of Rest of the Course

Part I

Organizational Preliminaries

Getting to know the students

- Please fill out the questionnaire and return to us.

Assignments and Expectations

- There will be 7 assignments in this course plus readings.
  1. Assignments 1-6 will contain short, factual, mathematical/computational exercises motivated by questions in phonology.
  2. Assignment 7 will be an extended assignment – applying what you learned in this class to a phonology problem of your own choosing.
  3. Readings will be made available every class and posted on the course site.
- Assignments 1-6 will be available at the end of classes 1–6 and reviewed at the beginning of classes 2–7.
- During review, your assignment will be scored by another student in the class.
- We will then collect assignments for our own review and return the following class.

Contacting the Instructors

- Office Hours
  - Jeff: TBD
  - Jon: TBD
- Email
  - Jeff: `jeffrey.heinz@stonybrook.edu`
  - Jon: `jon.rawski@stonybrook.edu`

Part II

What is phonology?

The fundamental insight

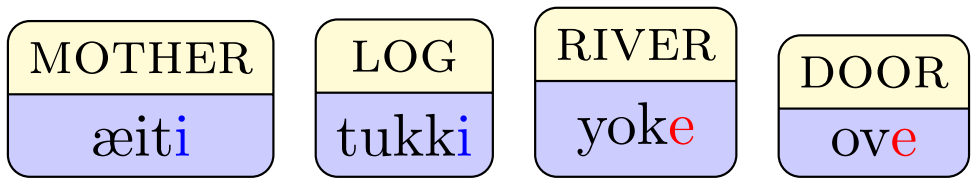
The fundamental insight in the 20th century which shaped the development of generative phonology is that the **best** explanation of the systematic variation in the pronunciation of morphemes is to posit a single underlying mental representation of the pronunciation of each morpheme and to derive its pronounced variants with context-sensitive transformations.

Example from Finnish

Nominative Singular	Partitive Singular	
aamu	aamua	‘morning’
kello	kelloa	‘clock’
kylmæ	kylmææ	‘cold’
kømpelø	kømpeløæ	‘clumsy’
æiti	æitiæ	‘mother’
tukki	tukkia	‘log’
yoki	yokea	‘river’
ovi	ovea	‘door’

Phonological Grammar

I. Mental Lexicon



II. Word-final /e/ raising

- e $\longrightarrow$ [+high] / $\_\_$ #
- *e# $\gg$ Ident(high)

If your theory asserts that . . .

There exist underlying representations of morphemes which are **transformed** to surface representations. . .

Then there are three important questions:

1. **What is the nature** of the more abstract, underlying, lexical representations?
2. **What is the nature** of the more concrete, surface representations?
3. **What is the nature** of the transformations from underlying forms to surface forms?

Theories of Phonology disagree on the answers to these questions, but *they agree on the questions being asked.*

Part III

Intensional and Extensional descriptions

Phonological transformations are infinite objects

Extensions of grammars in phonology are infinite objects in the same way that perfect circles represent infinitely many points.

Word-final /e/ raising

1. e $\longrightarrow$ [+high] / $\_\_$ #
2. *e# >> Ident(high)

Nothing precludes these grammars from operating on words of *any* length. The infinite objects those grammars describe look like this:

(ove,ovi), (yoke,yoki), (tukki,tukki), (kello,kello),. . .
(manilabanile,manilabanili), . . .

Describing circles

Circles are objects containing infinitely many points.
Example:

- Imagine a circle of radius 4 located at the origin of a coordinate system.
- If the system is Cartesian, then the circle is all points (x, y) which satisfy this equation.

$$x^2 + y^2 = 16$$

- If the system is Polar, then the circle is all points (r, θ) which satisfy this equation.

$$r = 4$$

The equations are *intensional* descriptions! But they are *extensionally equivalent* because they describe the same object!

In class exercise

Think of other examples of transformations in phonology. What are they and what are their extensions?

Truisms about transformations

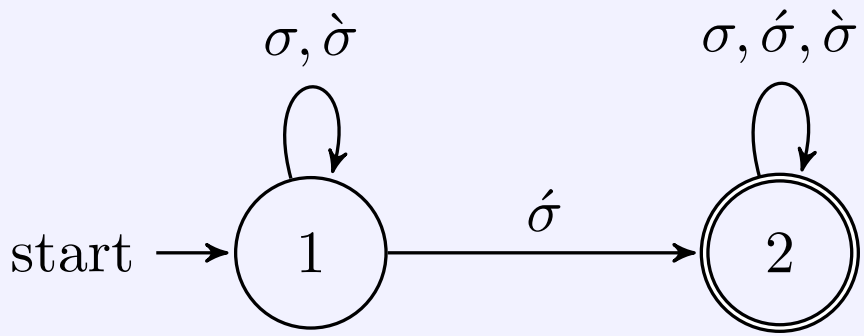
1. Different grammars may generate the same transformation. Such grammars are *extensionally equivalent*.
2. Grammars are finite, *intensional* descriptions of their (possibly infinite) *extensions*.
3. Transformations may have properties *largely independent* of their grammars.
    - Output-driven maps (Tesar 2014)
    - Regular functions (Elgot and Mezei 1956, Scott and Rabin 1959)
    - Subsequential functions (Oncina et al. 1993, Mohri 1997, Heinz and Lai 2013)
    - Input Strictly Local functions (Chandlee 2014, Chandlee et al. 2014, 2015)

Extensionally equivalent formalisms

LOGIC

$$\exists x \left[\texttt{primary\_stress}(x) \right]$$

AUTOMATA



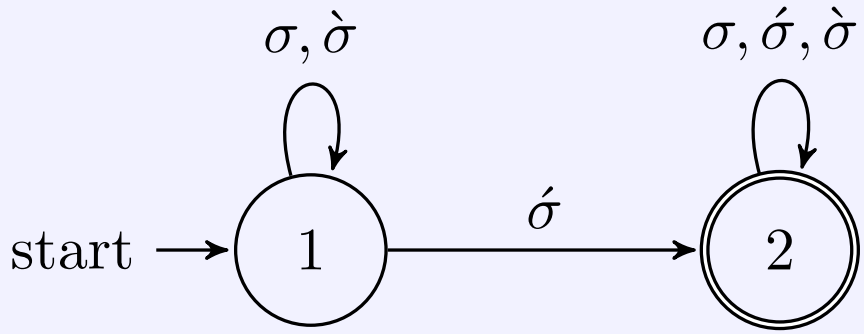
Extensionally equivalent formalisms

LOGIC

$$\exists x \left[\texttt{primary\_stress}(x) \right]$$

high-level

AUTOMATA



low-level

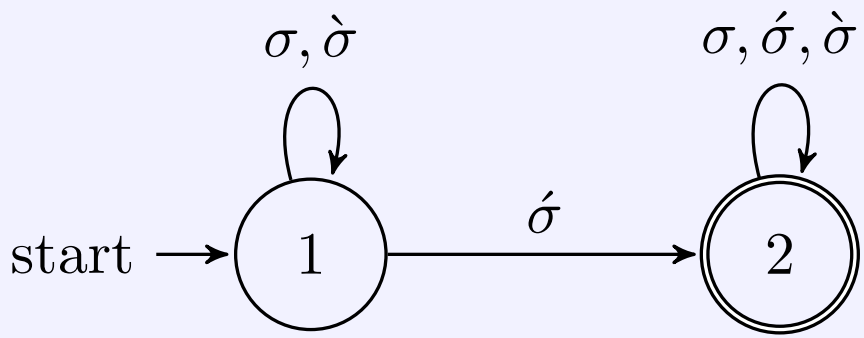
Extensionally equivalent formalisms

LOGIC

$$\exists x \left[\texttt{primary\_stress}(x) \right]$$

high-level; declarative

AUTOMATA



low-level; procedural

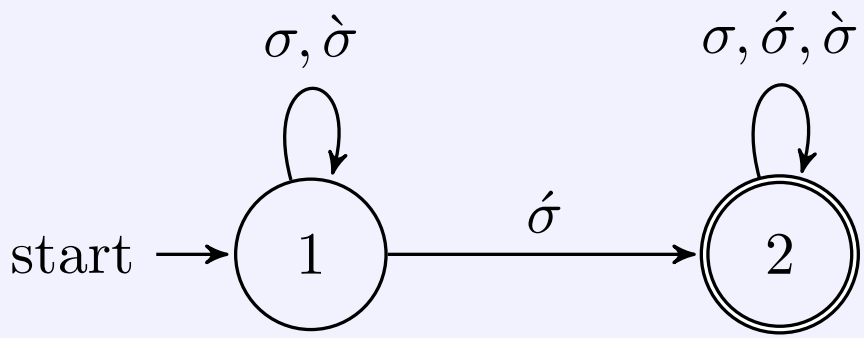
Extensionally equivalent formalisms

LOGIC

$$\exists x \left[\texttt{primary\_stress}(x) \right]$$

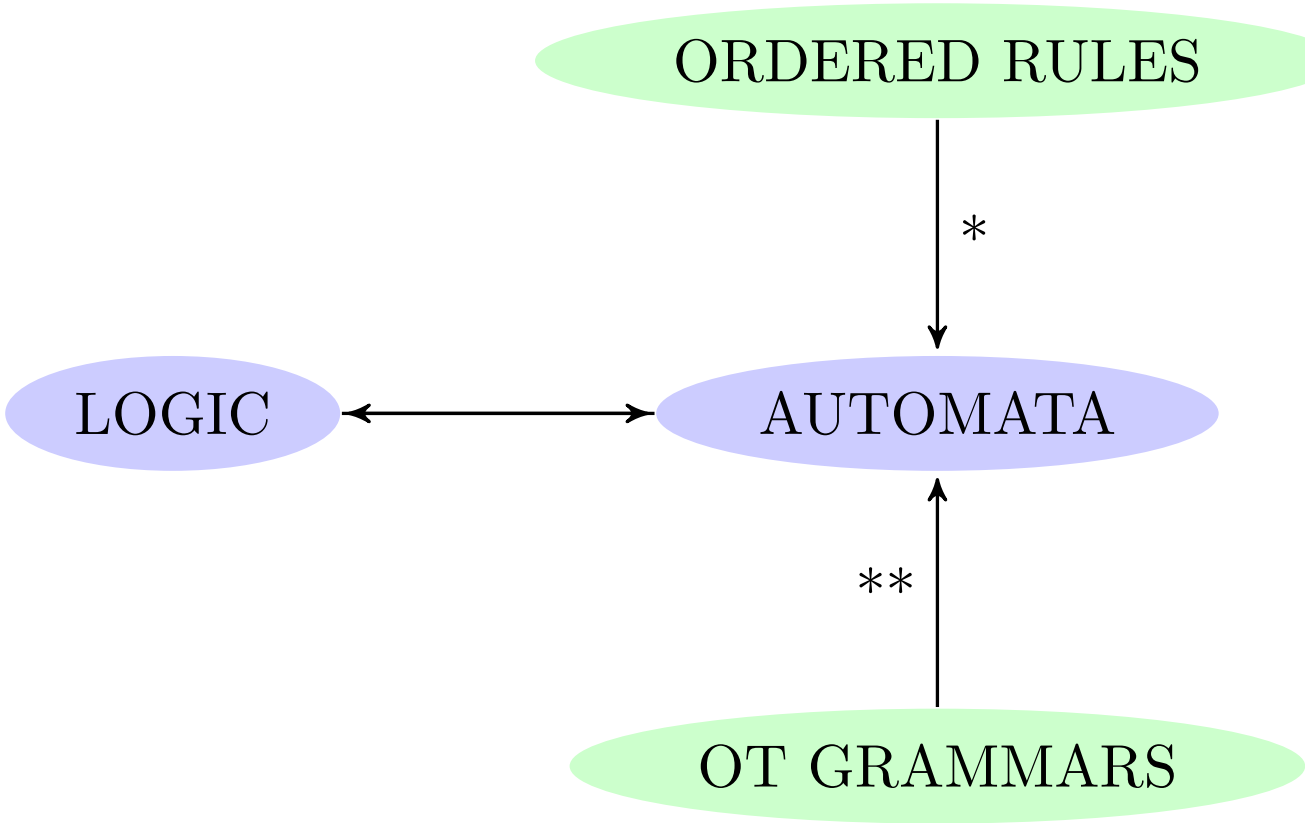
high-level; declarative; **specificies** the computation

AUTOMATA



low-level; procedural; **implements** the computation

Phonological grammars fit here too



*Johnson 1972, Kaplan and Kay 1994, Beesley and Karttunen 2003

**Under certain conditions (Frank and Satta 1998, Kartunnen 1998, Gerdemann and van Noord 2000, Riggle 2004, Gerdemann and Hulden 2012)

Constraints and Transformations as Functions

Function	Description	
$f : \Sigma^* \to \{0, 1\}$	Binary classification	(well-formedness)
$f : \Sigma^* \to \mathbb{N}$	Maps strings to numbers	(counting violations)
$f : \Sigma^* \to [0, 1]$	Maps strings to real values	(gradient well-formedness)
$f : \Sigma^* \to \Delta^*$	Maps strings to strings	(single-valued transformat
$f : \Sigma^* \to \wp(\Delta^* \times [0, 1])$	Maps strings to sets of strings with real values	(multi-valued transformati with gradience)

Table: Functions from strings to various co-domains

Part IV

Questions in Computational Phonology

(and answers)

Computational Phonology

1. What is the computational nature of the phonological *transformations*?

Computational Phonology

2. What is the computational nature of the phonological *constraints*?

3. What is the computational nature of the phonological *representations*?

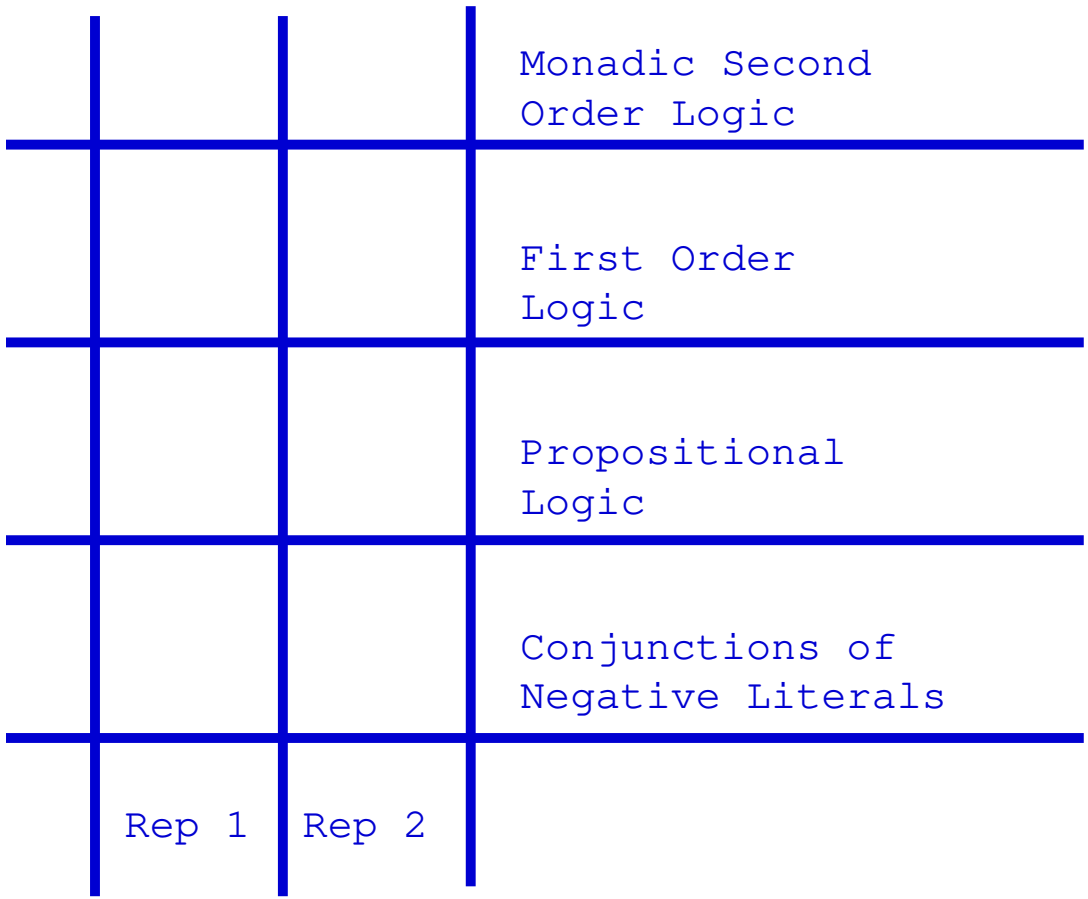
Sketch of advocated method for answering these questions

Identify *properties* of the *extensions*.

The space of all logically possible transformations and constraints can be factored along two dimensions.

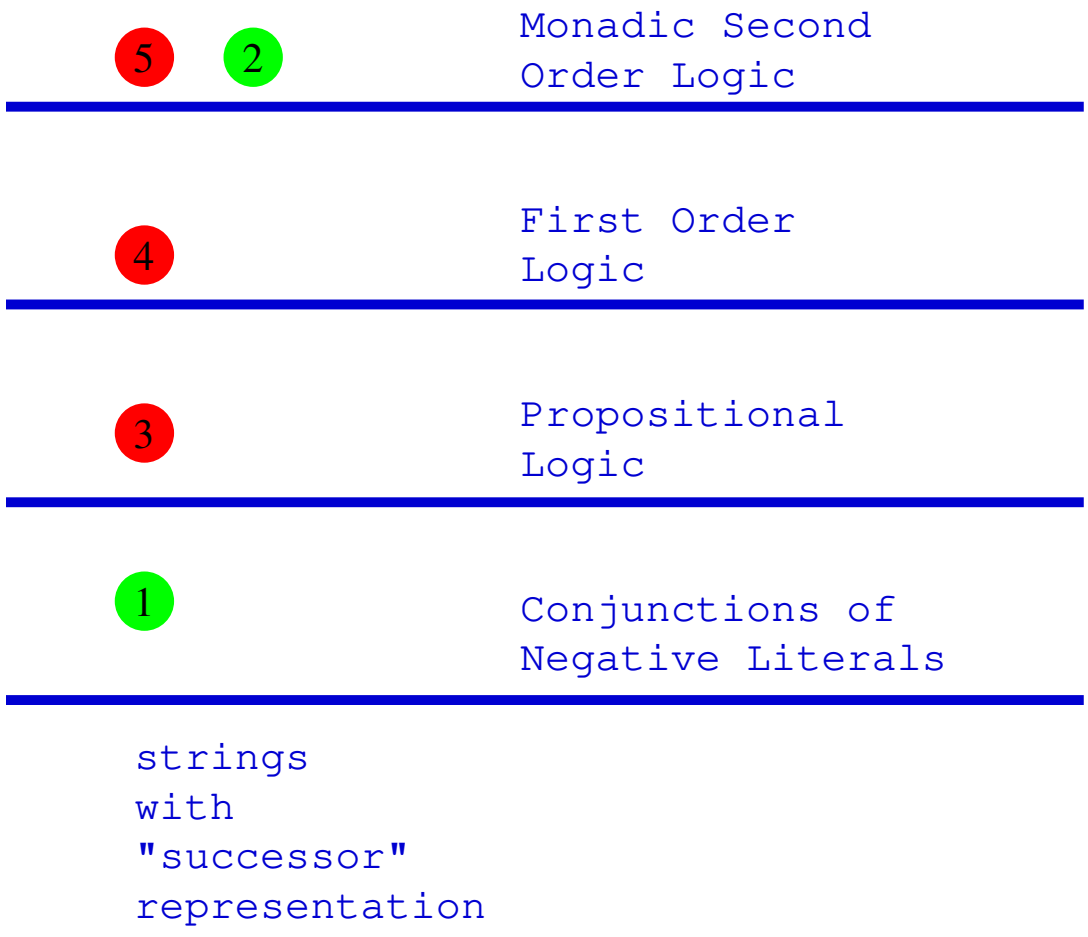
1. Logical Power
2. Representational Primitives

Example: Classifying Constraints w.r.t. Logic and Representation



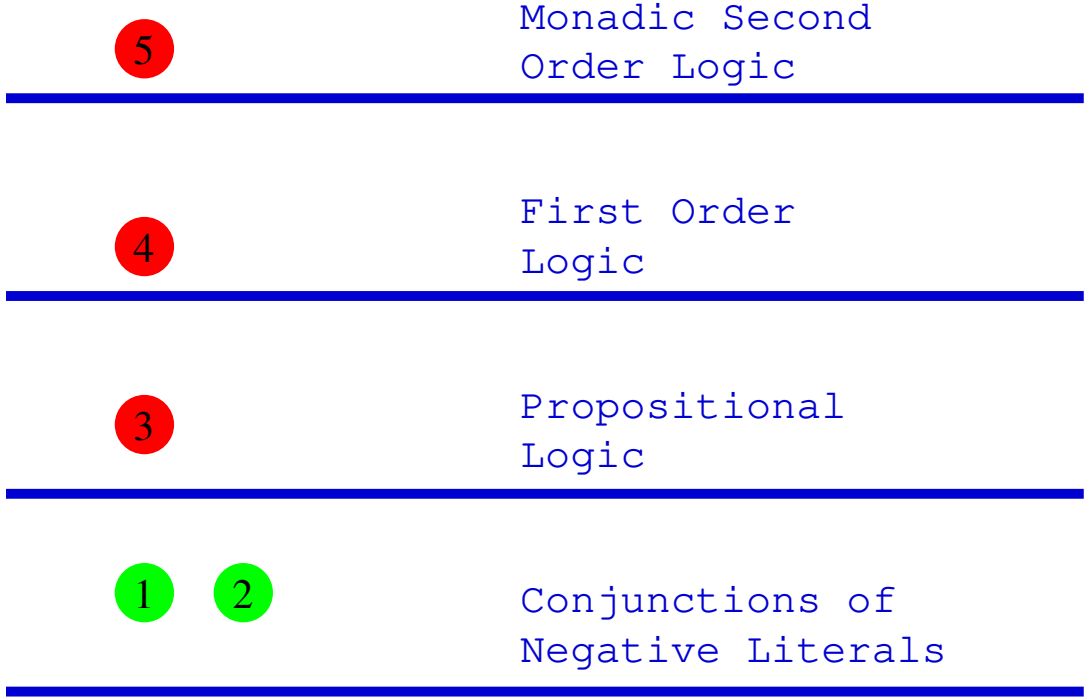
1. *sr
2. *s...S
3. If sr then VV
4. *3sr (but 2 OK)
5. *Even–Sib

Example: Classifying Constraints w.r.t. Logic and Representation



1. *sr
2. *s...S
3. If sr then VV
4. *3sr (but 2 OK)
5. *Even–Sib

Example: Classifying Constraints w.r.t. Logic and Representation



1. *sr
2. *s...S
3. If sr then VV
4. *3sr (but 2 OK)
5. *Even–Sib

Model-Theoretic Approach to Linguistic Structure and Generalizations

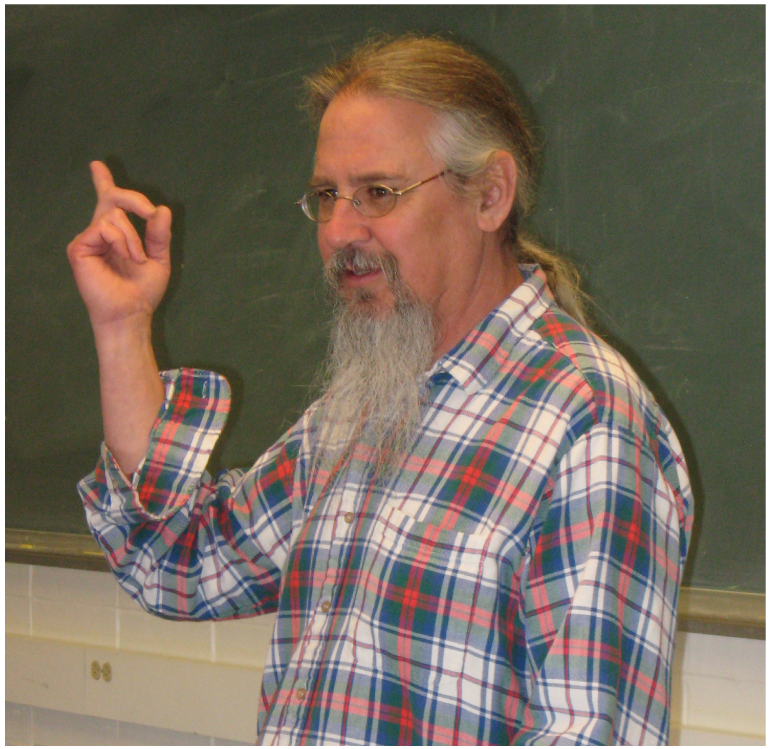
1. Researchers choose the degree of abstraction.
2. Different representations of structures can be shown to be translatable or not, and at what cost.
3. How representations can simplify or complicate a computation is made concrete.
4. Bounds on the complexity of the constraints and transformations of structures in language can be characterized and/or hypothesized.
5. Such bounds have implications for **cognition, psychology, typology, learning**.

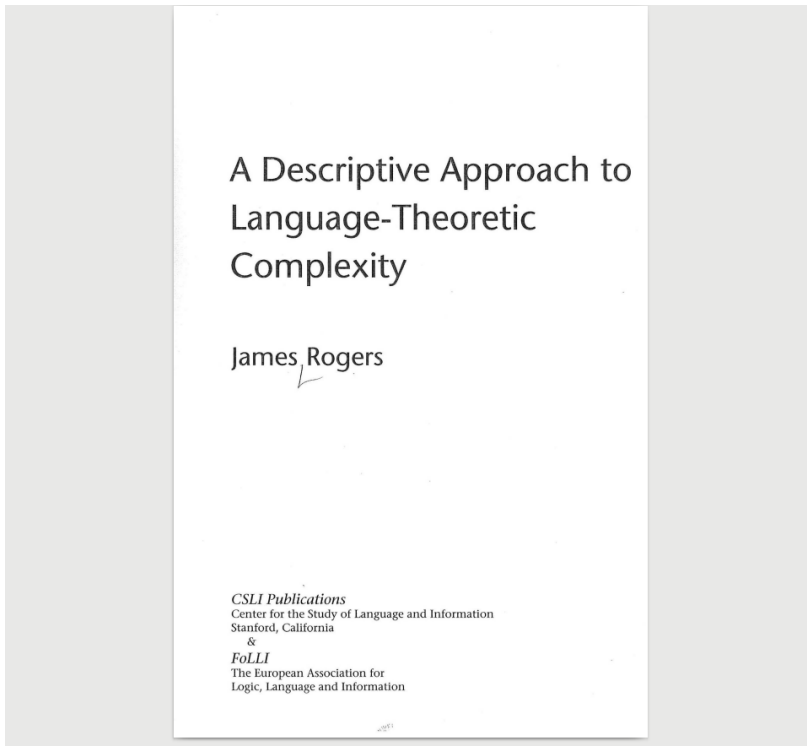
Shoulders of Giants

Linguist, Pullum 2007:7, "The Evolution of Model-Theoretic Frameworks in Linguistics"

> *I have tried to point out in the brief historical review above, however, that the flowering of this work that began in the middle 1990s was related to seeds planted some thirty years before. They were planted in stony ground, only inexpertly tended, and inadequately watered, but they were planted nonetheless. There is now an increasingly luxuriant garden to explore.*

Rogers 1998





The 2017 SIGMOL S.-Y. Kuroda Prize is awarded to James Rogers (Earlham College). James Rogers's 1998 book, "A Descriptive Approach to Language-Theoretic Complexity," was the first comprehensive work to apply monadic second-order logic to the analysis of linguistic theories...`http://molweb.org/mol/award-2017.html`

Channeling Humboldt

This factorization provides an ontology of types—*an encyclopedia of categories*—with which phonological phenomenon—*the encyclopedia of types*—can be identified.

Wilhelm Von Humboldt

Part V

What are strings?

Strings and Stringsets

Assume a finite set of symbols. Traditionally, Σ denotes this set. Strings are built inductively with a non-commutative operation called *concatenation*.

1. Base case: λ is a string.
2. Inductive case: If u is a string and $\sigma \in \Sigma$ then $u \cdot \sigma$ is a string.

- The string λ is the *identity*. So for all strings u: $u \cdot \lambda = \lambda \cdot u = u$.

- We refer to all strings with the notation Σ^*.

A *stringset* is a (possibly infinite) subset of Σ^*.

Part VI

Models

Word Models

We use the word 'word' synonymously with 'string.'

- A *model* of a word is a representation of it.

- A relational model contains two kinds of elements.
  1. **A domain.** This is a finite set of elements.
  2. **Some relations** over the domain elements.

- Guiding principles:
  1. Every word has some model.
  2. Different words must have different models.

What we want to model

Let $\Sigma = \{a, b, c\}$ and suppose we wish to model strings in Σ^*. A model for it

$$\mathbb{W}^{\triangleleft} = \langle \mathcal{D}, \triangleleft, \mathtt{a}, \mathtt{b}, \mathtt{c} \rangle$$

$\mathcal{D}^{\mathbb{W}}$	—	Finite set of elements (positions)
$\triangleleft^{\mathbb{W}}$	—	A binary relation encoding immediate linear precedence on $\mathcal{D}$
$\mathtt{a}, \mathtt{b}, \mathtt{c}$	—	Unary relations (so subsets of $\mathcal{D}$) encoding positions at which a,b,c occurs

Example: $\mathbb{W}^{\triangleleft}$

Consider the string *abbab*.
The model of *abbab* under the signature $\mathbb{W}^{\triangleleft}$ (denoted $\mathcal{M}^{\triangleleft}_{abbab}$) looks like this.

$$\mathcal{M}^{\triangleleft}_{abbab} = \left\langle \begin{array}{l} \{0,1,2,3,4\}, \\ \{(0,1),(1,2),(2,3),(3,4)\}, \\ \{0,3\}, \\ \{1,2,4\} \end{array} \right\rangle$$

In class exercise

1. Give models for these strings.
   1. *abc*
   2. *cacaca*
2. Suppose we removed the unary relations from the signature so the it looks like this: $\mathbb{W}^{\dagger} = \langle \mathcal{D}, \lhd \rangle$. Can models with such a signature distinguish all strings in Σ^*?
3. Suppose we removed the successor relation from the signature so it looks like this: $\mathbb{W}^{\ddagger} = \langle \mathcal{D}, P_a, P_b, P_c \rangle$. Can models with such a signature distinguish all strings in Σ^*?
4. Phonological theories often uses features as representational elements, not segments. How could you define a signature for a model that refers to features? What would the model of *can* [kæn] look like?

Part VII

Summary

Summary

1. We got to know each other a little bit.
2. We covered a lot of conceptual ground.
3. We got into some nitty-gritty with models.

Next class

- We learn to define *constraints* with First-Order (FO) logic with *different* models.

Future classes

- We learn to define *transformations* with First-Order (FO) logic.
- We learn to define Monadic Second-Order constraints and transformations.
- We learn to define *weighted* constraints and transformations with First-Order (FO) logic for specifying computations to count violations, yield probabilities, variation, and so on.
- We study constraints and transformations *below* FO logic!