185b/209b Computational Linguistics: Fragments

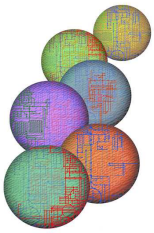
Edward Stabler*

2005-01-05

We will study the basic properties of human linguistic abilities by looking first at much simpler languages, 'language fragments', with particular attention to our ability to learn the semantic values of expressions and the role this could have in 'bootstrapping' the language learner.

* Thanks to many contributions from the 2005 class: Sarah Churng, Eric Fiala, Jeff Heinz, Ben Keil, Ananda Lima, Everett Mettler, Steve Wedig, Christine Yusi.

Contents



Linguistics 185b/209b: computational linguistics 2

To remove any frames surrounding this page click here. Refresh often.Last modified: Tue Mar 15 21:59:39 PST 2005

Lecture: TR12-2, in Public Policy 2292
Lecturer: Prof. Ed Stabler, Campbell 3103f, Office hours T2-3, stop by, or by appt stabler@ucla.edu

Prerequisites: 180/208 and 185a/209a or permission of instructor

Topic:

We will study the basic properties of human linguistic abilities by looking first at much simpler languages, 'language fragments', with particular attention to our ability to learn the semantic values of expressions and the role this could have in 'bootstrapping' the language learner.

UCLA Lx
UCLA Lx Talks

Ladefoged:sounds
phonetic fonts
MIT encyc
Rutgers cog sci
AZ syntax
semantics
lanl
acl
cogprints
Penn Lx tools
logic
dictionaries
Cornell univ
symp
Konstanz univs
ethnologue
IUbio
Lx newslist

corpora
ota
etext
perseus
classics archive
Childes
Susanne
Christine
(spoken)
Lucy (UK)
UMICH HTI
UMICH Middle Eng
LSE www search

aligned corpora
Europarl
Hansard:En-Fr
blinker:En-Fr
crater:En-Fr-Sp
ARTFL bible
more bibles

Contents:

Human languages are sometimes compared to logics: they have a syntax, a semantics, and since we can readily recognize certain entailment relations, it is natural to assume that there is some kind of *derives* relation among sentences. But human languages are surprising in many respects, with properties we would not usually design into a code or formal language used for communication or calculation, and properties that reflect abilities that we might not expect in organisms competing for survival. After a brief survey of some basic properties of human languages (with particular attention to surprising properties!), we will design some fragments, beginning with very simple ones, and study how the language fragments can be understood and learned. It turns out to be extremely easy to design fragments with extremely difficult computational properties; the challenge is to find simple steps towards human languages.

Texts:

Lecture notes together with selections from the current literature, including some of the following (depending in part on student interests):

- Lecture notes
  - 1 2 3 4 5 6 (7-1) 8 9 10 11 12 13 14 15 16 17 18 19 20
  - Stuart Shieber's lecture Towards a universal framework for tree transduction (discussed in our lecture 19)
  - the Mitton files (discussed in lecture 6)
    - original files 1, 2
    - mitton.txt, mitton2.txt, col2tokens.flex, mitton2tokens.txt, bigrams.cnt, bigrams
- Background
  - biological, cross-cultural perspectives
    - Hauser, Chomsky & Fitch (2001) "The faculty of language: what it is, who has it, and how did it evolve?" (Boston Globe 2003 story about this)
    - Pinker & Jackendoff (2004) "The faculty of language: what's special about it?"
    - Hauser, Chomsky & Fitch (2005) "Clarifications and implications", "Appendix"
  - quantifiers in the language of the Piraha and Munduruku?
    - Holden (2004) "Life without numbers",
    - Pica (2004) "Exact and approximate arithmetic in an Amazonian Indigene Group",
    - Everett (1992) "Cultural constraints...",
    - Gelman & Gallistel (2004) "Origin of Numbers",
- psychological perspectives
  - Kuhl (2004) "Early language acquisition: cracking the speech code"
  - Snedeker & Gleitman (2004) "Why it is hard to label our concepts"
- syntactic perspectives
  - Baker (2001) "Syntax"
  - Longobardi (2003) "Results in syntax"
  - Chomsky (2001) "An interview on minimalism"
- philosophical perspectives
  - Putnam (1973) "Meaning and reference"
  - Fodor and Lepore (1998) "The emptiness of the lexicon"
  - Harley (2004) "Wanting, having, and getting: a note on Fodor and Lepore 1998"
- engineering perspectives
  - Seagull & Schubert (2001) "Guiding a linguistically well-founded parser with head patterns"
  - Marcus (1984) "Some inadequate theories of human language processing"

- bootstrapping
  - Carey (2004) "Bootstrapping & the origin of concepts"
  - Snedeker (2000) "Cross-situational observation and the semantic bootstrapping hypothesis"
- Special topics
  - recognizing complex expressions
    - Shieber, Schabes & Pereira (1993) Principles and implementation of deductive parsing.
    - Seki, Matsumura, Fujii & Kasami (1991) "On multiple context-free grammars"
      Albro's MCFG parser (in OCAML) here
    - Stabler (1997) Derivational minimalism
    - Guillaumin (2004) Conversions between mildly sensitive grammars
    - Gibson (1998) "Linguistic complexity"
    - Ford (2002) "Packrat parsing" (ocaml-packrat)
  - recognizing entailments, polarity-based reasoning, and polarity items

    - Moss (2004) "Natural language, natural logic, natural deduction"
    - Pratt-Hartmann (2004) "Fragments of language"
    - Pratt-Hartmann and Third (2004) "More fragments of language"
    - Geurts (2003) "Quantifying Kids"
    - Spade (2002) "Thoughts, Words and Things: An Introduction to Late Mediaeval Logic and Semantic Theory" (Version 1.1a)
    - Szabolcsi (2004) "Positive polarity - negative polarity"
    - Fyodorov, Winter, Francez (2003) "Order-based inference in natural logic"
    - Winter & Altman (2004) "Computing Dominant Readings with Upward Monotone Quantifiers"
    - Su (2001) "Scope and specificity in child language"
  - learning combinatorial structure (morphology, syntax)
    - Harris (1955) "From phoneme to morpheme"
    - Baayen & Schreuder (2000) "Towards a psycholinguistic computational model for morphological parsing"
    - Goldsmith (2004) "An algorithm for the unsupervised learning of morphology"
    - Angluin (1982) "Inference of reversible languages"
    - Kanazawa (1994) "Learnable Classes of Categorial Grammmars"
    - van Zaanen (2001) "Bootstrapping syntax and recursion using alignment-based learning"
  - learning semantic values
    - Feldman (2000) "Minimization of Boolean complexity in human concept learning"
    - Feldman (2004) "How surprising is a simple pattern? Quantifiying "Eureka!""
    - Feldman (2003) "Perceptual grouping by selection of a logically minimal model"
      - DNF minimization demo (compare Feldman's results on Shephard's types)
      - Wegener (1987) "The complexity of Boolean functions" (more than you wanted to know!)
    - Siskind (1996) "A Computational Study of Cross-Situational Techniques for Learning Word-to-Meaning Mappings"
      and Siskind on computing visual percepts
    - Snedeker (2000) "Cross-situational learning and the bootstrapping hypothesis"
    - Phillips (2003) "Linguistics and linking problems"
  - dynamics of learning
    - Niyogi (2004) "The Computational Nature of Language Learning and Evolution"
    - Nowak, Komarova & Niyogi (2002) "Computational and evolutionary aspects of language"
    - Plotkin & Nowak (2001) Major transitions in language evolution
  - idioms and other "multi-word expressions"
    - Nunberg, Wasow & Sag (1994) "Idioms"
    - Sag et al (2002) "Multiword expressions"
    - Westerstahl (1999) "Idioms and compositionality"
  - co-occurence relations a clue to syntactic selection, semantic value?
    - Rooth "Two dimensional clusters in grammatical relations"
    - Carroll & Rooth "Valence induction with a head-lexicalized PCFG"
    - Bekkerman et al (2002) "Distributional Word-Clusters vs. Words for Text Categorization"

- Implementation

  - prolog
    - get SWI prolog
    - check how fast are you going
    - get tcl/tk and make sure you can run: wish
    - Stabler's prolog-based computational linguistics class (185a/209a)
    - tutorial: Blackburn, Bos, and Striegnitz 2001: Learn Prolog now!
  - ocaml
    - get ocaml
    - shootout: how fast does it go
    - John Hale's ocaml-based computational linguistics class
    - Marcus Kracht's ocaml-based computational linguistics class
    - tutorial: Chailloux, Manoury & Pagano (2000) Developing Applications with Objective CAML
      tutorial: Hickey (2002) Introduction to the Objective Caml Programming Language
      tutorial: Remy (2001) Using, Understanding and Unraveling the OCaml Language
  - scheme
    - get MIT/GNU scheme
  - prolog, ocaml and scheme MG parsers on Stabler's page

Requirements:

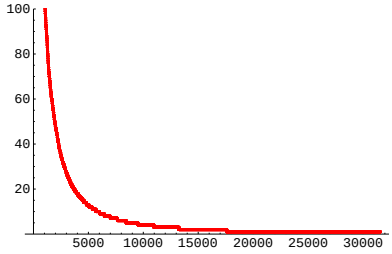
Occasional short exercises (probably 6-7), plus a final 'squib' or project on a relevant topic. The final grade will count the exercises for 75% and the final project for 25%. This is not a programming class, but we will present and discuss programs in class, using the languages SWI prolog, OCAML, and Octave (all freely available -- you should get them!). For the final squib, programming projects are encouraged.

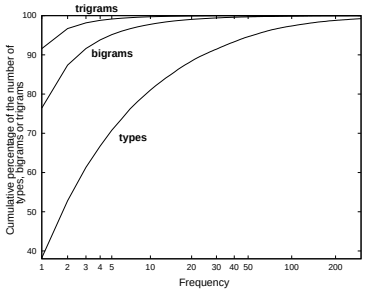
Note:

The material covered in this class varies from year to year, so it can be repeated for credit.

1 Bootstrapping: first questions

(1) Zipf: expression type rank and frequency related by inverse exponentials: most are rare [29]

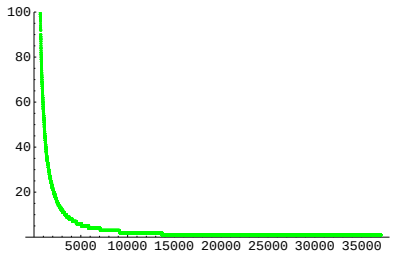




tb2: ≈32,000 words, ≈40% words unique, 75% bigrams, 90% trigrams, 99.7% sentences

(2) Frege: …a thought grasped by a terrestrial being for the very first time can be put into a form of words which will be understood by someone to whom the thought is entirely new. This would be impossible, were we not able to distinguish parts in the thought corresponding to the parts of a sentence, so that the structure of the sentence serves as an image of the structure of the thought. [4]

(3) (naive) 'construction' rank and frequency related by inverse exponentials: most are rare



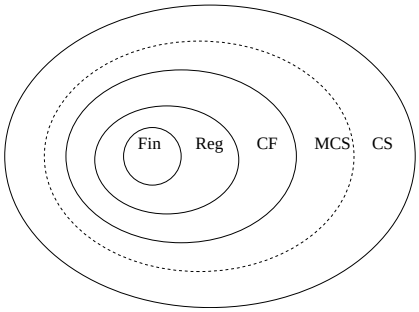
tb2: ≈37,000 constructions, ≈40% unique. We understand this, but… [19]

Q1 Can a better grammar significantly change this picture? How can we get it?

Q2 How can learners converge on similar grammars from diverse linguistic data?

Q3 How are the atoms (morphemes or whatever, the basis of the recursion) identified?

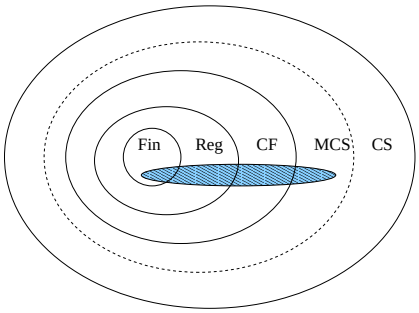
(4) Joshi et al: Human languages are 'mildly context sensitive' (MCS) [8, 25, 26, 13, 7, 14]



[5, 3, 20, 21] Q4 How can MCS recursion be identified? How can MCS languages be parsed?

(5) (Gold) It is impossible to learning arbitrary MCS languages from examples. Subsets possible, but an appropriate subset not yet found…

[6, 17, 1, 9, 22, 15]



This slightly 'antique' perspective has been replaced…

(6) Snedeker & Gleitman, et al: …the learning procedure in some way makes joint use of the structures and situations that cooccur with verbs so as to converge on their meanings. Neither source of evidence is strong or stable enough by itself, but taken together they significantly narrow the search space…

[11]

Could that change the learning situation from Gold's predicament? YES!

Q5 In the human predicament, are meanings required in principle?

(7) Snedeker & Gleitmann To learn that cat is the English-language word for the concept 'cat,' the child need only note that cats are the objects most systematically present in scenes wherein the sound /kat/ is uttered (just as proposed by Augustine (398); Locke (1690); Pinker (1984); and many other commentators).

[24, 18, 16, 23, 10]

Q6 What could 'systematically present' mean? And how far can this get us?

> *When they (my elders) named some object, and accordingly moved towards something, I saw this and I grasped that the thing was called by the sound they uttered when they meant to point it out. Their intention was shown by their bodily movements, as it were the natural language of all peoples: the expression of the face, the play of the eyes, the movement of the other parts of the body, and the tone of the voice which expresses our state of mind in seeking, having, rejecting or avoiding something. Thus, as I heard words repeatedly used in their proper places in various sentences, I gradually learned to understand what objects they signified; and after I had trained my mouth to form these signs, I used them to express my own desires.* (Augustine 398)

> *"The proposition says something" is identical with: it has a particular relation to reality, whatever this may be. And if this reality is given and also that relation, the sense of the proposition is known. $p \vee q$ has a different relation to reality from $p \wedge q$, etc.*
> *The possibility of a proposition is, of course, founded on the principle of signs as* going proxy *for objects…But there is also the common cement. My fundamental thought is that the logical constants are not proxies. That the logic of the fact cannot have anything as its proxy.*
> *A tautology…does not stand in any representing relation to reality.*
> *Roughly speaking, before any proposition can make sense at all the logical constants must have denotation [Bedeutung].* (Wittgenstein 1914)

> *Just as one can never dispense with rules of inferences by enriching one's axioms (though a limited trade-off is possible) so, no matter how interlocking the assignments to interacting expressions are made, one can never obviate the need for some semantic potential to be lodged in the grammatical constructions – even if it is merely that concatenation signifies functional application.* (Evans 1976)

Q7 How could we (or anything) learn the meaning of *and* ($\wedge$), *or* ($\vee$), *all*, or concatenation?

(8) Do we learn the meanings and entailment relations at once? or, one from the other?

(9) Logicians think that the 'entailment' relations among sentences are not all of the same kind…

John sings and John dances	All of the people dance	I know John dances	John is married
John dances	Someone dances	John dances	John is not a bachelor

The entailment on the left is sometimes said to hold 'in virtue of form' while the one on the right holds 'in virtue of meaning'. Is that right? Does anything like that distinction relate to how people recognize entailment relations?

References for Lecture 1

[1] ANGLUIN, D. Inference of reversible languages. *Journal of the Association for Computing Machinery 29* (1982), 741–765.

[2] AUGUSTINE. *Confessions.* 398. Reprinted with commentary by J.J. O'Donnell. NY: Oxford University Press, 1992.

[3] FORD, B. Parsing expression grammars. In *POPL'04* (2004).

[4] FREGE, G. Gedankengefüge. *Beträge zur Philosophie des deutschen Idealismus 3* (1923), 36–51. Translated and reprinted as 'Compound thoughts' in *Mind* **72**(285): 1-17, 1963.

[5] GIBSON, E. Linguistic complexity: Locality of syntactic dependencies. *Cognition 68* (1998), 1–76.

[6] GOLD, E. M. Language identification in the limit. *Information and Control 10* (1967), 447–474.

[7] HARKEMA, H. A characterization of minimalist languages. In *Logical Aspects of Computational Linguistics* (NY, 2001), P. de Groote, G. Morrill, and C. Retoré, Eds., Lecture Notes in Artificial Intelligence, No. 2099, Springer, pp. 193–211.

[8] JOSHI, A. How much context-sensitivity is necessary for characterizing structural descriptions. In *Natural Language Processing: Theoretical, Computational and Psychological Perspectives*, D. Dowty, L. Karttunen, and A. Zwicky, Eds. Cambridge University Press, NY, 1985, pp. 206–250.

[9] KANAZAWA, M. Identification in the limit of categorial grammars. *Journal of Logic, Language, and Information 5* (1996), 115–155.

[10] KOBELE, G. M., RIGGLE, J., COLLIER, T., LEE, Y., LIN, Y., YAO, Y., TAYLOR, C., AND STABLER, E. Grounding as learning. In *Language Evolution and Computation Workshop, ESSLLI'03* (2003).

[11] LIDZ, J., GLEITMAN, H., AND GLEITMAN, L. R. Kidz in the 'hood: Syntactic bootstrapping and the mental lexicon. In *Weaving a Lexicon*, D. Hall and S. Waxman, Eds. MIT Press, Cambridge, Massachusetts, 2004, pp. 603–636.

[12] LOCKE, J. *An Essay Concerning Human Understanding.* 1690. Reprinted. Cleveland Ohio: Meridian Books, 1964.

[13] MICHAELIS, J. Derivational minimalism is mildly context-sensitive. In *Proceedings, Logical Aspects of Computational Linguistics, LACL'98* (NY, 1998), Springer.

[14] MICHAELIS, J., AND KRACHT, M. Semilinearity as a syntactic invariant. In *Logical Aspects of Computational Linguistics* (NY, 1997), C. Retoré, Ed., Springer-Verlag (Lecture Notes in Computer Science 1328), pp. 37–40.

[15] NIYOGI, P. *The Computational Nature of Language Learning and Evolution.* MIT Press, Cambridge, Massachusetts, 2003. Forthcoming.

[16] PINKER, S. *Language Learnability and Language Development.* Harvard University Press, Cambridge, Massachusetts, 1984.

[17] PITT, L. *Probabilistic inductive inference.* PhD thesis, University of Illinois, 1989.

[18] REGIER, T., CORRIGAN, B., CABASAN, R., WOODWARD, A., GASSER, M., AND SMITH, L. The emergence of words. In *Proceedings of the Cognitive Science Society* (2001).

[19] SEAGULL, A. B., AND SCHUBERT, L. K. Guiding a linguistically well-founded parser with head patterns. Technical report 767, Computer Science Department, University of Rochester, 1991.

[20] SHIEBER, S., AND JOHNSON, M. Variations on incremental interpretation. *Journal of Psycholinguistic Research 22* (1994), 287–318.

[21] SHIEBER, S. M., SCHABES, Y., AND PEREIRA, F. C. N. Principles and implementation of deductive parsing. Tech. Rep. CRCT TR-11-94, Computer Science Department, Harvard University, Cambridge, Massachusetts, 1993.

[22] SHINOHARA, T. Inductive inference from positive data is powerful. In *Annual Workshop on Computational Learning Theory* (San Mateo, California, 1990), Morgan Kaufmann, pp. 97–110.

[23] SISKIND, J. M. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition 61* (1996), 39–91.

[24] SNEDEKER, J., AND GLEITMAN, L. R. Why it is hard to label our concepts. In *Weaving a Lexicon*, D. Hall and S. Waxman, Eds. MIT Press, Cambridge, Massachusetts, 2004, pp. 257–293.

[25] STABLER, E. P. Varieties of crossing dependencies: Structure dependence and mild context sensitivity. *Cognitive Science 93*, 5 (2004), 699–720.

[26] VIJAY-SHANKER, K., AND WEIR, D. The equivalence of four extensions of context free grammar formalisms. *Mathematical Systems Theory 27* (1994), 511–545.

[27] WITTGENSTEIN, L. *Tractatus logico-philosophicus*. Routledge and Kegan-Paul, London, 1963, 1922. The German text of Ludwig Wittgenstein's Logisch-philosophische Adhandlung, with a translation by D. F. Pears and B. F. McGuinness, and with an introduction by Bertrand Russell.

[28] WITTGENSTEIN, L. *Philosophical Investigations*. MacMillan, NY, 1958. This edition published in 1970.

[29] ZIPF, G. K. *The Psychobiology of Language: An introduction to dynamic philology*. Houghton-Mifflin, Boston, 1935.

2 The simplest languages

(1) We propose, following Montague and many others:

Each human language is a logic.

(2) What is a logic? A logic has three parts:

i. a language (a set of expressions, sequences of gestures) that has

ii. a "derives" relation $\vdash$ defined for it (a syntactic relation on expressions), and

iii. a semantics: expressions of the language have meanings.

(3) **Notation: Sequences** are written in many ways. I try to choose the notation to minimize confusion.

$$abc \qquad \langle a, b, c \rangle \qquad a, b, c \qquad [\mathsf{a}, \mathsf{b}, \mathsf{c}]$$

(4) **Notation:** Context free grammars are commonly written in a form like this, which we will use extensively:

S → DP VP	
VP → V DP	VP → V
DP → John	DP → Mary
V → saw	V → knew

These grammars are sometimes written in the slightly more succinct Backus-Naur Form (BNF) notation:

S ::= DP VP	VP ::= V NP | V
DP ::= John | Mary	V ::= saw | knew

The categories on the left side of the `::=` are expanded as indicated on the right, where the vertical bar separates alternative expansions. (Sometimes in BNF, angle brackets or italics are used to distinguish category from terminal symbols, rather than the capitalization that we have used here.) Various BNF-like notations are often used by logicians and computer scientists.

2.1 Finite languages

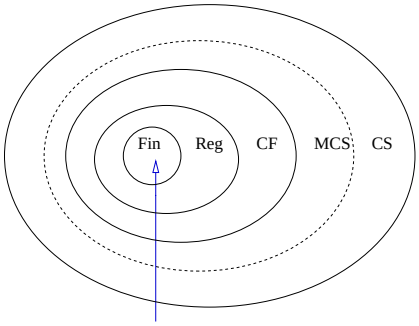
(5) Consider a language that just contains a finite number of syntactic and semantic atoms. So each grammar G has this form, for some finite number of elements $p_1, p_2, \ldots, p_n$ (where each of these elements can be complex):

$$\text{S ::= } p_1 \mid p_2 \mid \ldots \mid p_n$$

As usual, the language of any such grammar G is

$$L(G) = \{p_1, p_2, \ldots, p_n\}.$$

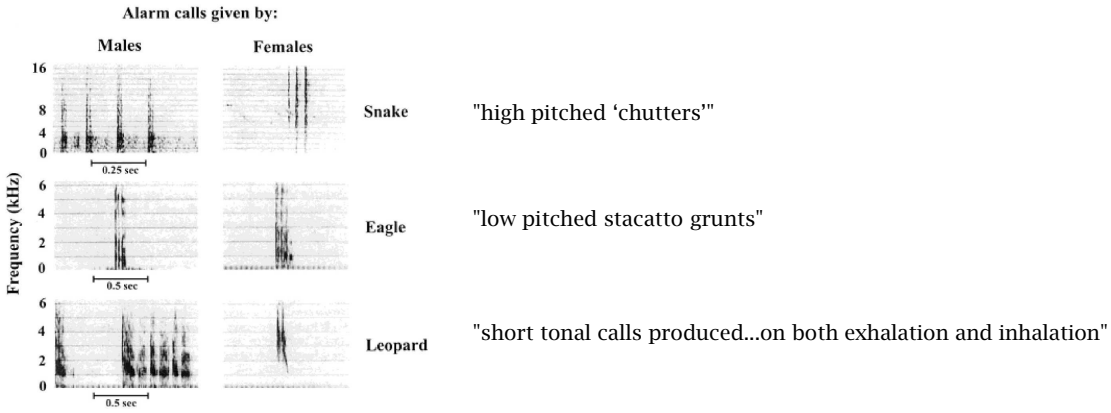
And as we have already seen, these languages have a prominent place in the Chomsky hierarchy, at the bottom:



(6) Obviously, the syntactic atoms can be phonologically, gesturally, orthographically complex, and their denotations can obviously be complex too…

(7) For example, vervet monkeys use different alarm calls for different predators. For example, one group of vervet monkeys was observed to use different calls depending on whether the cause of the alarm was a snake, a leopard or an eagle. Infants do not get these calls right, but use of the appropriate call improves with age and experience.

[6, 4]



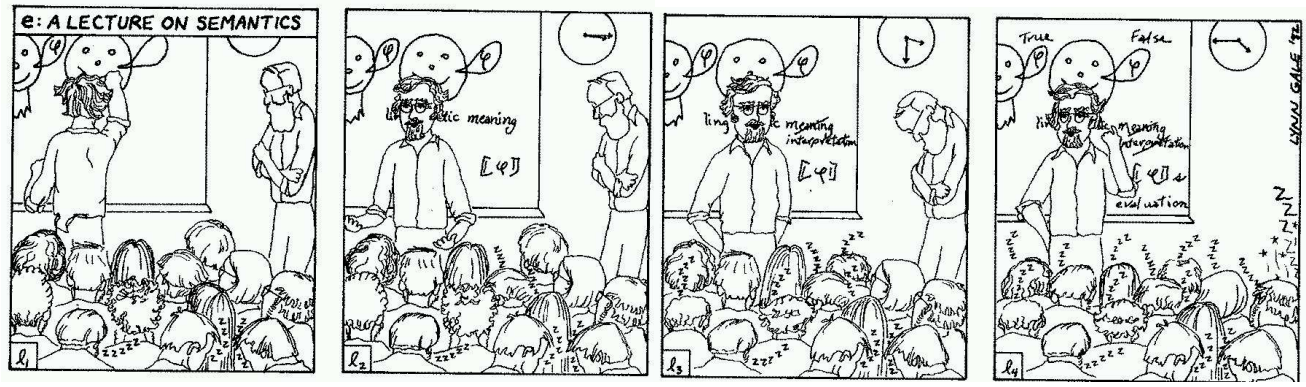
These monkeys look up when they hear the eagle call, down when they hear the snake call, and scurry into the trees when they hear the leopard call.

Although the young misuse these calls and improve with age, surprisingly, even unexperienced infants make these three different calls, and usually in roughly the right circumstances. (strange fact! – we will consider this more carefully later)

[2]

(8) Standardly, the semantic values of the expressions in any such language can be given by some (partial) function: $\mu : L(G) \rightarrow R$, where R is the semantic domain. Let $\mathcal{M}$ be the set of possible interpretation functions μ, the set of "models."

(9) This standard idea about semantic values is problematic. In a natural sense, semantic values can change from one situation to another. The following figure from [1, p.157] illustrates the changing denotation of the predicate sleep:



Barwise and Perry call these kinds of context dependence efficiency, because it allows some aspects of the message to be determined by the situation. The monkey calls mentioned above are efficient: the time and place of utterance are relevant.

(10) The standard way to represent semantic values for languages that are 'efficient' in this way is to think of the values of expressions as functions from contexts $\mathcal{W}$ (or 'worlds' or 'situations') to meanings.

So then we can think of the range M of μ as a set of functions from contexts to situated meanings' S.

(11) Any such semantics induces a partition on the language that we call 'synonymy'. For any $a, b \in L(G)$ and any $\mu \in \mathcal{M}$,

$$a \equiv_\mu b \text{ iff } a, b \in \text{dom}(\mu) \text{ and } \mu(a) = \mu(b).$$

Assuming that the semantic values in the range of μ are functions from contexts to meanings, then two expressions are equivalent in this sense iff they denote the same function from contexts to meanings.

(NB: this is not quite right – we will return to tinker with this later)

(12) Suppose every possible interpretation μ maps some subset of expressions $S \subseteq L$ to functions from contexts into a domain that is ordered by a Boolean relation $\leq$.[1] Then we can say

$$a \leq b \text{ iff } \forall \mu \in \mathcal{M},\ \forall w \in \mathcal{W},\quad \mu(a)(w) \leq \mu(b)(w).$$

When the values $\mu(a)(w)$ are truth values $\{0, 1\}$ and $a \leq b$ If this is the case in all models and all situations, then we say a entails b, $a \vDash b$.

(NB: this is not quite right – we will return to tinker with this later)

[1] We say that a relation $\leq$ on a set S is 'Boolean' iff it is a complemented and distributive lattice order. Remember that $\leq$ is a partial order iff

$x \leq x$	(it is reflexive)
$x \leq y$ and $y \leq x$ implies $x = y$	(it is antisymmetric)
$x \leq y$ and $y \leq z$ imply $x \leq z$	(it is transitive).

Given any partially ordered set $(A, \leq)$, any $S \subseteq A$ and any $x \in A$,

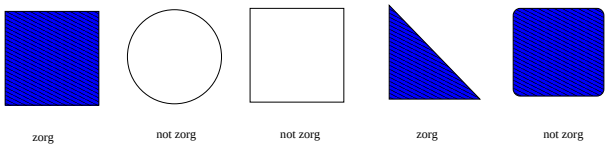
x is a *lower bound* for S iff for all $y \in S$, $x \leq y$
x is a *greatest lower bound* for S iff for all lower bounds y of S, $y \leq x$
x is an *upper bound* for S iff for all $y \in S$, $y \leq x$, and
x is a *least upper bound* for S iff for all upper bounds y of S, $x \leq y$.

A partially ordered set $(A, \leq)$ is a *lattice* iff $\forall x, y \in A$, $\bigwedge\{x, y\}$ and $\bigvee\{x, y\}$ exist. And it is *distributive* iff $\forall x, y, z \in A$, $\bigwedge\{x, \bigvee\{y, z\}\} = \bigvee\{\bigwedge\{x, y\}, \bigwedge\{x, z\}\}$ and $\bigvee\{x, \bigwedge\{y, z\}\} = \bigwedge\{\bigvee\{x, y\}, \bigvee\{x, z\}\}$. And finally, it is *complemented* iff it has a top $\top = \bigvee A$ and a bottom $\bot = \bigwedge A$, and for every $x \in A$ there is a $x' \in A$ (the *complement* of x, $\neg x$) such that $\bigwedge\{x, x'\} = \bot$, and $\bigvee\{x, x'\} = \top$.

2.2 Identifying (possibly complex) semantic values from (perfect) examples

(13) A simple kind of word learning has been studied in many 'concept learning' studies, where the task for the subject is (roughly) to guess the meaning of a word from some data. Here the word is given, and only the meaning needs to be figured out, so let's consider this simple problem first.

One of the simplest experimental paradigms involves showing a subject a sequence of positive and negative examples of some unknown concept 'blick'.



Humans (and also monkeys, rats, crows,...) can learn a concept like "shaded and not rounded"

(14) How can we tell when two concepts are the same? Is the concept "shaded and not rounded" the same as "not rounded and shaded"? Perhaps this is the the same as the linguistic question: are "shaded and not rounded" and "not rounded and shaded" exactly synonymous?

(15) Criterion for synonymy: first idea. Two expressions a, b are exactly synonymous only if one can be substituted for the other in every expression without changing the semantic value of that expression in any possible situation.

Using this criterion, Frege's famous examples show that the names 'Hesperus' and 'Phosphorus' are not synonymous, since the following sentences can easily differ in truth value:

Astronomers know that Phosphorus is visible in the morning
Astronomers know that Hesperus is visible in the morning

For many years, only the former was true, since it was not known that the planet Venus appears both in the morning and in the evening.

(16) The last definition implies that synonymous expressions will have exactly the same syntactic distribution. Is that true?

In a recent paper, Hodges says no on the basis of facts like these:

[3]

the beast ate the meat	the beast devoured the meat
the beast ate	the beast devoured
It's likely that Alex will leave	It's probable that Alex will leave
Alex is likely to leave	Alex is probable to leave

Westerståhl considers facts like these. If 'runs' is a single morpheme, then it is natural to assume it is synonymous with 'run', but clearly these are not intersubstitutable.

[7]

Are any of these examples persuasive? (We should return to this kind of question later when we can bring more considerations to bear)

(17) Another kind of example:

He filled the bucket	he filled the pail
He kicked the bucket	he kicked the pail

(18) Many studies show that a concept like "shaded and not rounded" is easier for people to learn than "shaded or not rounded." Why would this be? In both cases we need to learn all 4 lines of the truth table:

s	r	$s \wedge \neg r$	$s \vee \neg r$
0	0	0	1
1	0	1	1
0	1	0	0
1	1	0	1

Not only that, but any propositional concept like these can be expressed in infinitely many ways.

shaded and not rounded $\leftrightarrow$ neither unshaded nor rounded

$(s \wedge \neg r) \leftrightarrow \neg(\neg s \vee r)$

References for Lecture 2

[1] Barwise, J., and Perry, J. *Situations and Attitudes.* MIT Press, Cambridge, Massachusetts, 1983.

[2] Cheney, D. L., and Seyfarth, R. M. *How Monkeys See the World.* University of Chicago Press, Chicago, 1992.

[3] Hodges, W. Formal features of compositionality. *Journal of Logic, Language and Information 10* (2001), 7–28.

[4] Owren, M. J., and Rendall, D. Sound on the rebound: Bringing form and function back to the forefront in understanding nonhuman primate vocal signaling. *Evolutionary Anthropology 10*, 2 (2001), 58–71.

[5] Seligman, J., and Moss, L. S. Situation theory. In *Handbook of Logic and Language*, J. van Benthem and A. ter Meulen, Eds. Elsevier, NY, 1997.

[6] Seyfarth, R. M., Cheney, D. L., and Marler, P. Monkey responses to three different alarm calls. *Science 210*, 4410 (1980), 801–803.

[7] Westerståhl, D. On the compositional extension problem. *Journal of Philosophical Logic forthcoming* (2004).

3 The simplest languages: concept learning

3.1 Summary

(1) We are considering the following fragment, with the 3 parts of a 'logic'.

grammar G: S ::= p_1 | p_2 | ...| p_n

semantics: $\mu : L(G) \rightarrow R$, where R is the semantic domain.

- R may provide context-dependent values, functions from contexts to meanings.
- each particular interpretation μ determines what expressions are synonymous with each other – a 'partition' of the language.
- the specification of all the possible interpretations (there may be more than one) determines which expressions entail which others.
- synonymy does not necessarily guarantee intersubstitutability, though this is still a relevant consideration for languages like English (for reasons we left unexplored – we should come back to this)

inference: so far, we allow arbitrary binary inference relations. (We will explore inference relations more carefully in fragments where they interact with grammatical structure and semantics.)

So the grammar, the semantics, and the inference relations look rather simple.

(2) Now we consider how such a simple language could be learned – apparently even vervet monkeys can learn certain languages like this. To start, we consider the situation where the language is given: the speaker can recognize the linguistic expressions. And suppose that the speaker is explicitly given positive and negative examples

(cf Augustine's idea about learning what 'cat' means)

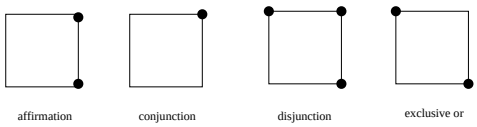
We noted, to start, only the fact that "conjunctive concepts" like "round and red" are easier to learn from examples than concepts like "round or red"

3.2 Identifying Boolean semantic values: humans

(3) Learning that *zorg* means "shaded and not rounded" is easier than learning that it means "shaded or not rounded." Why? Each is specified by a truth table with the 4 possible assignments of truth values to "shaded" and "rounded". [1]

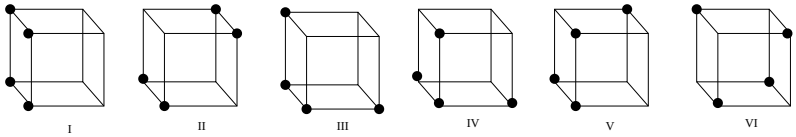
(4) Idea: maybe the relevant thing is explicit in the truth table itself: how many positive instances does the concept have? This turns out not to be the determining factor.

Feldman represents the Boolean functions of two variables in a way that makes the number of positives easy to see, collapsing across the (infinitely many) different formulas that are similar in this respect: [2, 3]



[1] It turns out that these are shown here in order of increasing difficulty, from left to right, so the number of positives is not the determining factor.

[5] A study by Shepard, Hovland and Jenkins complicates matters, showing that if we consider formulas with 3 atomic propositions and exactly 4 positives, we find these reliably differ in difficulty:

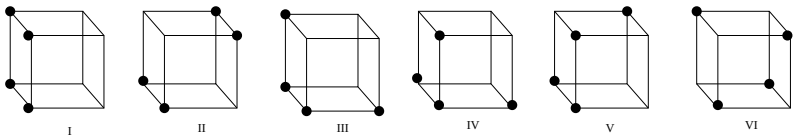


I < II < {III,IV,V} < VI

(5) This typology equates formulas with an equivalence which can be defined this way. First let a **relabeling** be a total 1-1 function f from propositional atoms to literals such that Rng(f) is consistent. Extend f to apply to any formula ϕ, by replacing every atom p in ϕ by $f(p)$. Then,

$$\phi \equiv \phi' \text{ iff there is some relabeling } f \text{ such that } f(\phi) \leftrightarrow \phi'.$$

(6) What's going on here? Following earlier conjectures, Feldman observes that when we calculate minimal representations of each of the 7 concepts – as well as we can – we get these formulas (to save space, Feldman uses + for $\vee$, ' for $\neg$, and concatentation for $\wedge$):



I	<	II	<	{III,	IV,	V}	<	VI
a'		ab+a'b'		a'(bc)'+ab'c	a'(bc)'+ab'c'	a'(bc)'+abc		a(b'c+bc')+a'(b'c'+bc)

This supports the surprising idea that the difficulty of learning a Boolean concept is related to the size of its minimal expression.

The formulas $p \wedge q$ and $p \vee q$ have the same size, so in this case we can stick to some version of the idea that a concept with few positives is easier to learn than a concept with many positives.

Feldman explores a wider range of formula types – 76 of them in [2] – confirming that there is definitely something right about this perspective. He also applies it to perceptual grouping phenomena. So let's explore how this could work.

3.3 Identifying Boolean semantic values: setting the stage

(7) A *literal* is an atomic proposition or its negation.

(And by a *complex* proposition, I do not mean *complicated*, I just mean: not atomic.
So $\neg p$ is a complex proposition.)

If there are n atomic propositions, an *assignment* is a vector of n truth values, an element of $\{0,1\}^n$.

A *n-ary Boolean function f* is a function from $\{0,1\}^n$ to $\{0,1\}$.

(So for each $n \geq 0$, there are 2^{2^n} n-ary Boolean functions.)

For any propositional formula ϕ, let function f_ϕ be the Boolean function it defines.

(8) A vector v *satisfies* or *verifies* f iff $f(v) = 1$.
$f \Rightarrow g$ iff for all vectors v, if $f(v) = 1$ then $g(v) = 1$. In this case we say f *implies* g.

(9) A propositional formula is in **conjunctive normal form** (CNF) iff it is a conjunction of disjunctions of literals. It is in **disjunctive normal form** (DNF) iff it is a disjunction of conjunctions of literals.

(10) Since $\wedge$ and $\vee$ commute, a CNF or DNF formula can be represented as a set of sets of literals. The empty disjunction is sometimes written $\bot$ or $\Box$ and is always false; the empty conjunction is sometimes written $\top$ and is always true.

We can convert any formula into CNF using the following algorithm.

Input: An arbitrary formula ϕ (over $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$)
Output: A CNF formula ϕ'
while any of the following equivalences can be applied to any subexpression in the $\rightarrow$ direction **do**
$(p \leftrightarrow q) \leftrightarrow ((p \rightarrow q) \wedge (q \rightarrow p))$
$(p \rightarrow q) \leftrightarrow (\neg p \vee q)$
$\neg(p \vee q) \leftrightarrow (\neg p \wedge \neg q)$
$\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$
$\neg\neg p \leftrightarrow p$
$p \vee (q \wedge r) \leftrightarrow (p \vee q) \wedge (p \vee r)$
$(q \wedge r) \vee p \leftrightarrow (q \vee p) \wedge (r \vee p)$
end while

Algorithm 2CNF

We can convert any formula into CNF using the following algorithm.

Input: An arbitrary formula ϕ (over $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$)
Output: A DNF formula ϕ'
while any of the following equivalences can be applied to any subexpression in the $\rightarrow$ direction **do**
$(p \leftrightarrow q) \leftrightarrow ((p \rightarrow q) \wedge (q \rightarrow p))$
$(p \rightarrow q) \leftrightarrow (\neg p \vee q)$
$\neg(p \vee q) \leftrightarrow (\neg p \wedge \neg q)$
$\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$
$\neg\neg p \leftrightarrow p$
$p \wedge (q \vee r) \leftrightarrow (p \wedge q) \vee (p \wedge r)$
$(q \vee r) \wedge p \leftrightarrow (q \wedge p) \vee (r \wedge p)$
end while

Algorithm 2DNF

3.4 Identifying Boolean semantic values: exact learning in the limit

(11) A *propositional learner* PL is a (total) mapping from finite sequences of total vectors (corresponding to the trials) to Boolean functions (corresponding to the guesses about the target).
Given a function f, a *positive text* t *for* f is a sequence of assignments that contains all and only the assignments that verify f.
We will represent each assignment by the set of literals that the assignment verifies.
We will call this function f the *target* function of the text.

An (informant) text for f is a sequence of all the assignments together with the value of f on those assignments.

PL *converges* on text t iff there is a function F that is the value of PL for all but finitely many initial segments of t. That is, f eventually settles on a value that new data will never change. In this case we say $\text{PL}(t) = f$.

PL *identifies* text t iff PL converges on t and t is a text for $\text{PL}(t)$.

PL *identifies* function f (from positive text) iff PL identifies every (positive) text for f.

PL *identifies* a class of functions $\mathcal{F}$ (from positive text) iff it identifies every $f \in \mathcal{F}$ (from positive text).

A class of functions $\mathcal{F}$ is *identifiable* (from positive text) iff there is a learning function that identifies it (from positive text).

(12) **Theorem:** The class of Boolean functions of n variables is identifiable from positive text (for any n). We will show this by presenting algorithms that provably learn any Boolean function, in the sense we have just defined.

Input: A j-element initial segment of a positive text t for target function f.
Output: A DNF formula ϕ_j represented as a set of sets of literals
  $\phi_0 := \emptyset$ the (always false) empty disjunction
  for $i = 1$ to j **do**
    Select $v_i \in t$ (represented by the set of literals that v_i verifies)
    $\phi_i := \{v_i\} \cup \phi_{i-1}$
  end for

Algorithm PL-DNF

Call the function computed by this algorithm "PL-DNF." The following proposition is obvious.

(13) PL-DNF identifies the class of Boolean functions from positive text

Proof: By definition, every text for f contains all of the assignments verifying f, and since there are only finitely many of them, they all must occur in some j-element initial segment of the text, for finite j. For any such j, then, it is clear that $\{v | f(v) = 1\} = \{v | f_{\phi_j}(v) = 1\}$, i.e. $f = f_{\phi_j}$. Furthermore, it is clear that for all $k \geq j$, $\phi_k = \phi_j$. □

(14) We can use CNF instead (and we'll see later how this CNF learner is more easily made efficient!).

Input: A j-element initial segment of a positive text t for target function f
Output: A CNF formula ϕ_j represented as a set of sets of literals
  ϕ_0 := the conjunction of all disjunctions of literals
  for $i = 1$ to j **do**
    Select $v_i \in t$ (represented by the set of literals that v_i verifies)
    Let ϕ_i be the result of deleting every conjunct of ϕ_{i-1} that does not contain an element of v_i
  end for

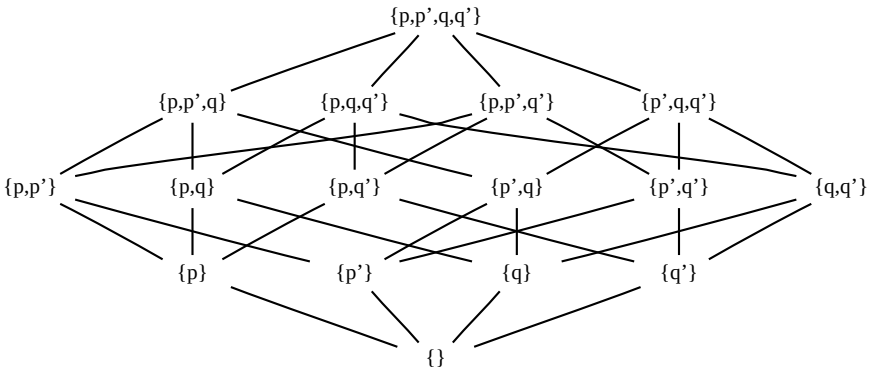
Algorithm PL-CNF

Call the function computed by algorithm PL-CNF simply "PL-CNF."

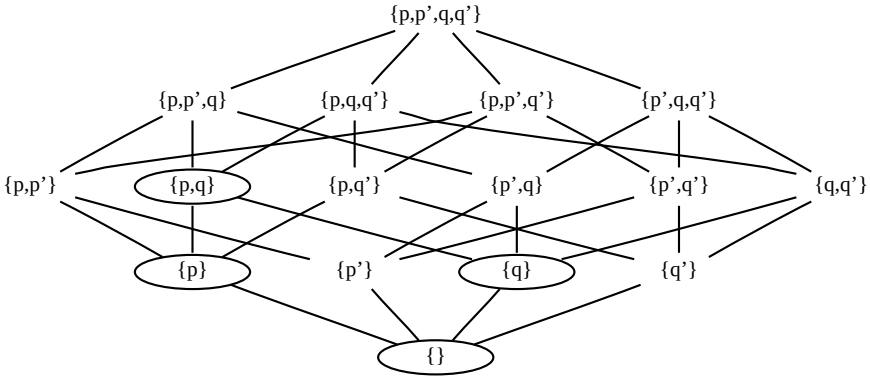
(15) **Example.** Suppose that we are trying to learn the function f represented by this truth table, which we considered in class:

p	q	f
0	0	1
1	0	0
0	1	0
1	1	1

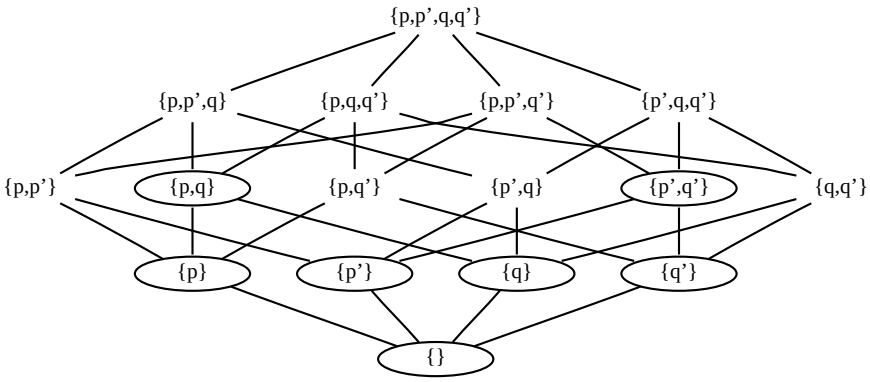
The PL-CNF learner begins with the whole set of 16 disjunctions:



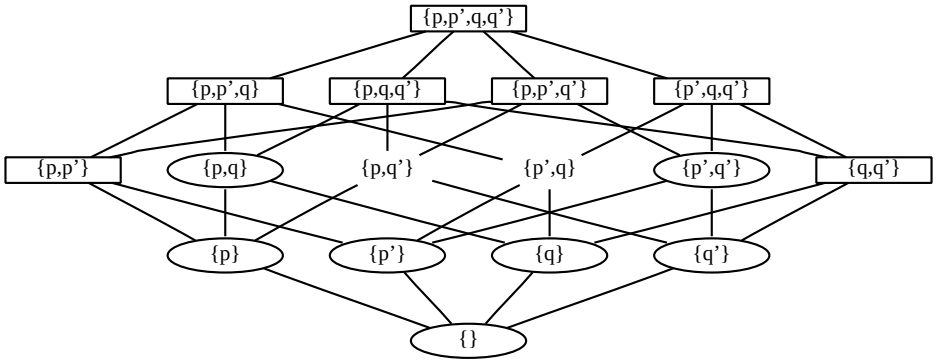
Suppose the learner first sees that f holds when $\neg p \wedge \neg q$, and so removes every disjunction that does not contain one of $\{\neg p, \neg q\}$. So we remove the elements circled here:



Now suppose that the learner sees that f holds when $p \wedge q$, and so removes every disjunction that does not contain either of $\{p, q\}$, leaving:



Some of the disjunctions here are tautologies, which are redundant in a CNF formula, so let's can mark them with boxes:



Now it is easy to see that this is the desired result. The CNF formula $(p \vee \neg q) \wedge (\neg p \vee q)$ defines the target function.

(16) **Theorem:** PL-CNF identifies the class of Boolean functions from positive text.[1]

Proof: Let f be an arbitrary target Boolean function and let t be an arbitrary text for f. We must show that there is a finite j such that for all $k \geq j$, the k-element initial segment t_k of t is such that PL-CNF$(t_k) = \phi_j$ where $f_{\phi_j} = f$. This result follows if we establish the following claims:

(1) At every step, $f_{\phi_i} \Rightarrow f$.

(2) There is a finite j such that for all $k \geq j$, $f_{\phi_j} = f_{\phi_k}$ and $f \Rightarrow f_{\phi_k}$.

(Proof of Claim 1) Initially, for $i = 0$, it is obvious that $\{v | \phi_i(v)\} = \emptyset$, and so $f_{\phi_i} \Rightarrow f$ is trivially true.

Let B be the product of all the disjunctions c such that, for all assignments v, if $f(v) = 1$ then $f_c(v) = 1$.

Obviously, the algorithm will never delete any disjunction in B from ϕ_i, and so $\{v | \phi_i(v) = 1\} \subseteq \{v | f_B(v) = 1\}$. Thus it suffices to prove that $f_B = f$. This is established by showing $f_B \Rightarrow f$ and $f \Rightarrow f_B$.

By definition, every element of B is implied by every assignment that implies f, so $f \Rightarrow f_B$.

If some disjunction c' verified by f did not occur in B, then, by the definition of B, there would have to be an assignment v such that $f(v) = 1$ but $f_{c'}(v) = 0$. But this is impossible since if c' is verified by f and if $f_{c'}(v) = 0$, then $f(v) = 0$.

(Proof of Claim 2) At each step, algorithm PL-CNF deletes all of the disjunctions in ϕ_{i-1} that do not contain a literal that is verified by v_i. Thus, $f_{\phi_i}(v_j) = 1$ for all $j \leq i$. By definition, every text for f contains all of the assignments verifying f, and since there are only finitely many of them, they all must occur in some j-element initial segment of the text, for finite j. For any such j, then, we have $\{v | f(v) = 1\} \subseteq \{v | f_{\phi_j}(v) = 1\}$, i.e. $f \Rightarrow f_{\phi_j}$. Furthermore, it is clear that for all $k \geq j$, $\phi_k = \phi_j$. □

(17) Unfortunately, these learners are not feasible because of the size of the conjectures. In the case of PL-DNF, the conjectured formula contains every assignment seen so far. If there are n propositional variables, there are $2n$ literals, and so there are 2^{2n} disjuncts in the initial conjecture. In the case of PL-CNF, the problem arises in the initialization step. If there are n propositional variables, there are $2n$ literals, and the 2^{2n} subsets of these literals correspond to the 2^{2n} conjuncts of ϕ_0 in PL-CNF.

(18) More importantly, PL-DNF and PL-CNF do not have anything like the behavior that Shepard et al. discovered on Boolean functions of the forms I-VI.

[1] Cf. Valiant's [6, Thm A] and Gold's proof of the learnability of the finite languages from positive text [4, Thm I.6].

Exercise 1 [a]

(1) a. The formula $p \oplus q$ is equivalent to $(p \vee q) \wedge \neg(p \wedge q)$.
Use the algorithm 2CNF to put $(p \vee q) \wedge \neg(p \wedge q)$ into CNF.

b. Use the algorithm 2DNF to put $(p \vee q) \wedge \neg(p \wedge q)$ into DNF.

c. Represent the function $f_{p \oplus (q \oplus (r \oplus \neg p))}$ in Feldman's graphical notation.
Is it one of Shepard's types I-VI?

d. What is the shortest formula ϕ that expresses $f_{p \oplus (q \oplus (r \oplus \neg p))}$?

(2) For practice with the set notation for CNF:

a. Represent $p \oplus q$ in set CNF notation (using the result from above).

b. How do we know that if any element of a CNF set S contains both p and $\neg p$ for any p, then some proper subset S' of S is equivalent to S?

(An element C of a CNF set S that does not contain both p and $\neg p$ for any p is sometimes called a **cube**.)

c. Explain why every nonempty cube is satisfiable.

d. Explain why it is appropriate to call S **redundant** if it contains two cubes C, C' where $C \subseteq C'$, and how can you transform any redundant set S of cubes to an equivalent nonredundant set?

(Every Boolean function can be represented by a nonredundant set of cubes.)

(3-**optionally!**) code up one or more of the algorithms you used in these exercises, so that you don't have to do these things by hand!

[a] Warning! I sometimes cannot resist giving tricky exercises – sometimes ridiculously easy, sometimes impossible. The right response to an exercise that seems impossible or unreasonable is to explain why it's impossible or unreasonable!

References for Lecture 3

[1] BOURNE, L. E. Knowing and using concepts. *Psychological Review 77*, 6 (1970), 546–556.

[2] FELDMAN, J. Minimization of complexity in human concept learning. *Nature 407* (2000), 630–633.

[3] FELDMAN, J. How surprising is a simple pattern? Quantifying 'Eureka!'. *Cognition 93* (2004), 199–224.

[4] GOLD, E. M. Language identification in the limit. *Information and Control 10* (1967), 447–474.

[5] SHEPARD, R., HOVLAND, C., AND JENKINS, H. Learning and memorization of classifications. *Psychological Monographs: General and Applied 75*, 13 (1961), 1–42.

[6] VALIANT, L. A theory of the learnable. *Communications of the Association for Computing Machinery 27*, 11 (1984), 1134–1142.

4 The simplest languages: concept learning 2

4.1 Summary

(1) We are considering the following fragment,

grammar G: S ::= p_1 | p_2 | ...| p_n (each expression is just given, in this "lexicon")

semantics: $\mu : L(G) \rightarrow R$ (each expression just assigned some meaning)

inference: so far, arbitrary

Now we are considering how such a simple language could be learned. Apparently even vervet monkeys can learn certain languages like this.

(2) We can distinguish two parts to the learning problem, parts that might be at least partially separable:

a. recognizing the expressions $p_1, p_2, \ldots, p_n$

b. identifying the meanings of these expressions

We first consider only the second step, part b.

(3) For b, a simplistic interpretation of Augustine is obviously not right: we cannot get the meaning of a word p_i simply by simply intersecting everything we were percieving or thinking on occasions when we hear p_i to get at some common factor. Some of those occasions might be ones where we are being told "this is NOT p_i." In those cases, we do not expect any p_i to be present.

(4) **Exact learning in the limit.** Think of the learner as a function from data to hypotheses, where the data is (a finite part of) a "text" of data, and where this text can either be just positive instances ("positive text"), or else labeled positive and negative instances ("informant text").

We saw that, given a specification of what propositional parameters to attend to, it is easy to define learners in this sense for arbitrary Boolean concepts.

(5) For the learners PL-CNF and PL-DNF, it is easy to see that they will succeed, but only after seeing all the positive instances.

- This would fit with the idea that concepts with few positive instances are easier to learn – but we saw that this idea is not right!

  The work of Bourne, Shepard, Feldman and many others shows that the complexity of a concept seems to depend on how hard it is to represent the concept in some sense.

- Also, neither PL-CNF nor PL-DNF are feasible in general, since a Boolean function with n variables can have 2^n positive instances – exponentially many. Waiting to see them all is not feasible when n gets larger. We want learners that will generalize, ideally in something like the way described by the psychologists.

(6) After doing the first exercise, you will not be surprised to learn that converting to CNF, or to DNF, is not feasible either. The standard conversion of $(p_1 \wedge q_1) \vee (p_2 \wedge q_2) \vee \cdots \vee (p_n \wedge q_n)$ to CNF yields a conjunction of 2^n disjunctions, each containing 2 literals.

There are standard tricks for avoiding the explosion in these conversions: [11, 3]

4.2 Identifying semantic values: PAC learning

(7) We see that it is a trivial matter to design algorithms that can, in principle, "learn" any Boolean function. So Valiant points out that, if we are interested in real cases of learning, we should confine our attention to classes of functions that can be learned with a feasible amount of computation:

[16, 15]

> *In this model the impediment to learnability is computational complexity. If members of a class can be acquired by learning only in exponentially many steps then this class will not be learnable in practice.*

No surprise: It turns out that this criterion is very hard to meet!

So Valiant adds that for many purposes we do not need to require perfect learning. It's often enough to decrease the probability of error to an arbitrarily small amount: probably approximately correct (PAC) learning.

(8) We consider learners who draw samples from a sample space $X = \{0,1\}^n$, and so in our definitions we restrict attention to countable spaces X.

(9) Consider any distribution μ on X and any target class $\mathcal{L} \subseteq \wp X$. Then for any target $L \in \mathcal{L}$ and any hypothesis $h \subseteq X$, the **error** of the hypothesis is the measure of the symmetric difference between the hypothesis and the target:

$$error_\mu(L,h) = \mu((L \backslash h) \cup (h \backslash L)).$$

A distribution μ on X induces a distribution μ^m on X^m obtained by regarding each element of each sequence in X^m as drawn randomly and independently from X according to distribution μ. For any target L, let $S(m,L)$, the samples of length m, be the sequences $x_1, \ldots, x_m \in X^m$ labeled according to whether each element is in L:

$$((x_1, F_L(x_1)), \ldots, (x_m, F_L(x_m))).$$

(10) $\mathcal{L}$ is **efficiently PAC-learnable** iff there is a learning algorithm A such that, for any real numbers $0 < \epsilon, \delta < 1$, there is a positive integer $m_0 = m_0(\delta, \epsilon)$ such that for any target $L \in \mathcal{L}$ and any probability distribution μ on X, whenever $m \geq m_0$,

$$\mu^m\{s \in S(m,L) |\ error_\mu(L, A(s,\delta,\epsilon)) \leq \epsilon\} \geq 1 - \delta.$$

and the time required to compute $A(s,\delta,\epsilon)$ is bounded by a polynomial function of $(\frac{1}{\epsilon}, \frac{1}{\delta}, size(L))$.

(11) Let's consider concepts which can be coded as a set of n binary parameters, so each can be represented by a conjunction literals, a **monomial**.

Since conjunction is associative and commutative, and redundancies have no effect on semantic value, each monomial can be represented by a set of literals, and we use the overline notation for negation. So for example, $p_1 \wedge \overline{p_2} \wedge p_4$ is represented by $\{p_1, \overline{p_2}, p_4\}$. This formula is verified by vectors, each of which can be represented as a sequence from $\{0,1\}^4$ or again by a set of literals

$$\begin{array}{ll} \langle 1,0,0,1\rangle & \{p_1, \overline{p_2}, \overline{p_3}, p_4\} \\ \langle 1,0,1,1\rangle & \{p_1, \overline{p_2}, p_3, p_4\}. \end{array}$$

To eliminate double negations, for any literal l, let

$$\overline{l} = \begin{cases} \overline{p_i} & \text{if } l = p_i \\ p_i & \text{if } l = \overline{p_i} \end{cases}$$

We extend this function to sets pointwise: if s is a set of literals, $\overline{s} = \{\overline{l} \mid l \in s\}$.

(12) There are 2^{2^n} n-ary Boolean functions, but only 3^n monomials, since in a monomial each of the n propositional atoms can be positive, negated, or absent.

(13)

Input: A j-element sample t of labeled vectors for target function f, drawn according to μ. (Each vector represented by the set of literals it verifies.)
Output: A monomial ϕ_j (represented as a set of literals)
  $\phi_0 := \{p_1, \overline{p_1}, \ldots, p_n, \overline{p_n}\}$ (the conjunction of all literals – always false)
  for $i = 1$ to j **do**
    Select labeled $v_i \in t$
    if label=1 **then**
      $\phi_i := \phi_{i-1} \backslash \overline{v_i}$
    else
      $\phi_i := \phi_{i-1}$ (we ignore negative instances of the concept)
    end if
  end for

Algorithm PL-MON

(14) **Theorem:** Monomials are efficiently PAC learnable from labeled samples.

Proof: We show that PL-MON efficiently PAC learns the monomials.

For any literal $l \in (h - c)$, define the probability of counterevidence to l,

$$\mu(l) = \mu(\{x \in \{0,1\}^n \mid x \in c,\ \overline{l} \in x\}),$$

and so $error_p(c,h) \leq \sum_{l \in (h-c)} \mu(l)$. Notice that when there are n variables, there are exactly $2n$ literals. Say literal l is **bad** if $\mu(l) \geq \frac{\epsilon}{2n}$, so if h contains no bad literals,

$$error_p(c,h) \leq \sum_{l \in (h-c)} \mu(l) \leq 2n\frac{\epsilon}{2n} = \epsilon.$$

Now we consider how many samples we need to look at in order to have a hypothesis which, to the desired degree of confidence, has no bad literals.

Where h_i is the hypothesis after i samples, for any particular $l \in (h-c)$, $\mu(l \in (h_i - c)) \leq (1 - \frac{\epsilon}{2n})^i$ since each sample will cause bad l to be deleted with probability at least $\frac{\epsilon}{2n}$.

Then the probability that a particular literal is not deleted in a given trial is $1 - \frac{\epsilon}{2n}$
and the probability that it's not deleted after i trials is $(1 - \frac{\epsilon}{2n})^i$,
and since there are $2n$ literals, the probability that some bad literal remains after i samples is not more than $2n(1 - \frac{\epsilon}{2n})^i$. So what i (how many samples) brings this value below δ?

Since $1 - x \leq e^{-x}$, it suffices to find an i such that $2n(1 - \frac{\epsilon}{2n})^i \leq 2ne^{\frac{-i\epsilon}{2n}} \leq \delta$.

Now solving for i,

$$
\begin{array}{ll}
2ne^{\frac{-i\epsilon}{2n}} \leq \delta & \\
2n\frac{1}{\delta}e^{\frac{-i\epsilon}{2n}} \leq 1 & \text{dividing} \\
e^{\log_e 2n}e^{\log_e \frac{1}{\delta}}e^{\frac{-i\epsilon}{2n}} \leq 1 & \text{since } \forall x, y,\ x = y^{\log_y x} \\
e^{\log_e 2n+\log_e \frac{1}{\delta}}e^{\frac{-i\epsilon}{2n}} \leq 1 & \text{since } \forall x, y, z,\ x^y x^z = x^{y+z} \\
e^{\frac{-i\epsilon}{2n}} \leq \frac{1}{e^{\log_e 2n+\log_e \frac{1}{\delta}}} & \text{dividing} \\
e^{\frac{i\epsilon}{2n}} \geq e^{\log_e 2n+\log_e \frac{1}{\delta}} & \text{taking inverses} \\
\frac{i\epsilon}{2n} \geq (\log_e 2n + \log_e \frac{1}{\delta}) & \text{taking } \log_e \\
i \geq \frac{2n}{\epsilon}(\log_e 2n + \log_e \frac{1}{\delta}) & \text{multiplying by } \frac{2n}{\epsilon}
\end{array}
$$

Since $\log_e 2n$ is bounded by n, and $\log_e \frac{1}{\delta}$ by $\frac{1}{\delta}$, the number of samples i that we need is bounded by a polynomial function with respect to $n, \frac{1}{\epsilon}, \frac{1}{\delta}$. If the learner ϕ gets at least i examples, it will have reached the PAC criterion.

Since in addition each sample can be processed in linear time, the class is efficiently PAC learnable w.r.t. dimension n. □

(15) **Theorem:** The class of disjunctions of 2 conjunctions of literals is not efficiently PAC learnable w.r.t. n from positive and negative samples.

[10, 7]

The 'naturalness' of the PAC characterization of learning is supported by converging characterizations of the classes of real valued functions for which Empirical Risk Minimization (ERM) is consistent: they are the uniform Glivenko-Cantelli (uGC) classes [1, 9]. And it was known earlier that for the special case of simple classifiers – discrete $\{0, 1\}$ valued functions – the uGC classes are exactly those with "finite VC dimension" [17, 4].

4.3 Identifying semantic values: Feldman

(16) Feldman takes the results of Shepard et al seriously and also finds further evidence for the idea that we are best at learning simple things, things that can be defined with simple propositional formulas. How could this be?

(17) In a famous pioneering paper on the subject published in 1952, Quine says: [12]

> *…there remains one problem which, despite the trivial character of truth-function logic, has proved curiously stubborn; viz., the problem of devising a general mechanical procedure for reducing any formula to its simplest equivalent.*

Now we know that computing the minimimum size formula is almost certainly intractable in principle. The famous Quine-McCluskey algorithm simplifies DNF formulas, but it is intractable in general and even when it can be successfully executed it fails to yield the simplest results in many cases. [12, 13, 8, 18]

(18) Feldman proposes that people prefer concepts that have low complexity when represented as a kind of 'power series expansion'. [5, 6]

Simplifying his presentation slightly, for any Boolean function f, let's define the following series of conjunctions of negations of conjunctions of literals. Let's use the notation l_i for any literal, and represent conjunctions as sets. Then for all $k \geq 0$ we define

$$\Phi^k_f = \{\neg\{l_0, \ldots l_k\} \mid f \Rightarrow \neg\{l_0, \ldots, l_k\} \text{ and } \textstyle\bigwedge_{0 \leq j < k} \Phi^j_f \not\Rightarrow \neg\{l_0, \ldots l_k\}\}$$

(19)

Input: Boolean function f
Output: A 'power series expansion' g of f
  $i := 0$
  $g := \Phi^i_f$
  while $g \not\Leftrightarrow f$ **do**
    $i := i + 1$
    $g := f \cup \Phi^i_f$
  end while

Algorithm POWERSERIES

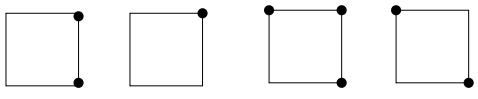
(20) **Theorem:** POWERSERIES always terminates, with $g \Leftrightarrow f$

Proof: f will have $i \leq 2^n$ negative instances, and each negative instance can be represented by the conjunction c_j of literals that it verifies, so

$$\begin{array}{rcl} \neg f & \Leftrightarrow & c_1 \vee \ldots \vee c_i \\ f & \Leftrightarrow & \neg(c_1 \vee \ldots \vee c_i) \\ f & \Leftrightarrow & \neg c_1 \wedge \ldots \wedge \neg c_i \end{array}$$

Each $\neg(c_j)$ is in Φ^{n-1}_f unless it is entailed by earlier conjunctions, and so POWERSERIES will halt at Φ^{n-1}_f or earlier. □

(21) Let's return to the earlier examples. The classic result is that people learn the leftmost concepts in the following order most easily:



f=affirmation	f=conjunction	f=disjunction	f=xor
$\Phi^0_f = \{\neg(\overline{p})\}$	$\Phi^0_f = \{\neg(\overline{p}), \neg(\overline{q})\}$	$\Phi^0_f = \varnothing$ $\Phi^1_f = \{\neg(\overline{pq})\}$	$\Phi^0_f = \varnothing$ **exercise!**

References for Lecture 4

[1] Alon, N., Ben-David, S., Cesa-Bianchi, N., and Haussler, D. Scale-sensitive dimensions, uniform convergence, and learnability. *Journal of the Association for Computing Machinery 44*, 4 (1997), 615–631.

[2] Bourne, L. E. Knowing and using concepts. *Psychological Review 77*, 6 (1970), 546–556.

[3] Boy de la Tour, T. Minimizing the number of clauses by renaming. In *Proceedings of the 10th Conference on Automated Deduction, CADE-10* (1990).

[4] Dudley, R., Giné, E., and Zinn, J. Uniform and universal Glivenko-Cantelli classes. *Journal of Theoretical Probability 4*, 3 (1991), 485–510.

[5] Feldman, J. Minimization of complexity in human concept learning. *Nature 407* (2000), 630–633.

[6] Feldman, J. How surprising is a simple pattern? Quantifying 'Eureka!'. *Cognition 93* (2004), 199–224.

[7] Kearns, M. J., and Vazirani, U. V. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, Massachusetts, 1994.

[8] McCluskey, E. J. Minimization of Boolean functions. *Bell Systems Technical Journal 35*, 5 (1956), 1417–1444.

[9] Mukherjee, S., Niyogi, P., Poggio, T., and Rifkin, R. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of Empirical Risk Minimization. *Advances in Computational Mathematics* (2004). forthcoming.

[10] Pitt, L., and Valiant, L. Computational limitations on learning from examples. *Journal of the Association for Computing Machinery 35* (1988), 965–984.

[11] Plaisted, D. A., and Greenbaum, S. A structure-preserving clause form translation. *Journal of Symbolic Computation 2* (1986), 293–304.

[12] Quine, W. The problem of simplifying truth functions. *American Mathematical Monthly 59* (1952), 521–531.

[13] Quine, W. A way to simplify truth functions. *American Mathematical Monthly 62* (1955), 627–631.

[14] Shepard, R., Hovland, C., and Jenkins, H. Learning and memorization of classifications. *Psychological Monographs: General and Applied 75*, 13 (1961), 1–42.

[15] Valiant, L. Deductive learning. *Philosophical Transactions of the Royal Society of London, Series A 312* (1984), 441–446.

[16] Valiant, L. A theory of the learnable. *Communications of the Association for Computing Machinery 27*, 11 (1984), 1134–1142.

[17] Vapnik, V., and Chervonenkis, A. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications 16* (1971), 264–280.

[18] Wegener, I. *The Complexity of Boolean Functions*. Wiley, NY, 1997.

5 The simplest languages: concept learning 3

5.1 Summary

(1) We are considering the following fragment,

grammar G: S ::= p_1 | p_2 | ...| p_n (each expression is just given, in this "lexicon")

semantics: $\mu : L(G) \rightarrow R$ (each expression just assigned some meaning)

inference: so far, arbitrary

Now we are considering how such a simple language could be learned. Apparently even vervet monkeys can learn certain languages like this.

(2) We can distinguish two parts to the learning problem, parts that might be at least partially separable:

a. recognizing the expressions $p_1, p_2, \ldots, p_n$

b. identifying the meanings of these expressions

Today we conclude our initial consideration of the second step, part b.

(3) **Augustine's proposal.** A simplistic interpretation of Augustine is obviously not right: we cannot get the meaning of a word p_i simply by simply intersecting everything we were perceiving or thinking on occasions when we hear p_i. Some of those occasions might be ones where we are being told "this is NOT p_i."

(4) **Exact learning in the limit.** Think of the learner as a function from data to hypotheses, where the data is (a finite part of) a "text" of data, and where this text can either be just positive instances ("positive text"), or else labeled positive and negative instances ("informant text").

Given a specification of what propositional parameters to attend to, it is trivial to define learners in this sense for arbitrary Boolean functions from examples in which the properties $p_1, \ldots, p_n$ defining the concept are given, but these learners of arbitrary Boolean functions are intractable.

(5) **Efficient PAC learning.** It is possible to efficiently learn monomials (even though the number of monomial functions is exponential in n). But it is not possible to PAC learn an arbitrary Boolean function, and so we are stuck if we want to learn, e.g., all the 76 functions that Feldman used in the psychological study reported in *Nature*

(6) **Feldman's proposal.** Going back to the psychological studies (Bourne, Shepard, Feldman and many others), what we might like to do is to find the simplest formula that fits with the evidence so far, but this is again intractable. Feldman proposes instead that people learn most easily those functions that are represented by low 'powers' $k \geq 0$ in the following sequence: [5, 6]

$$\Phi_f^k = \{\neg\{l_0, \ldots l_k\} \mid f \Rightarrow \neg\{l_0, \ldots, l_k\} \text{ and } \bigwedge_{0 \leq j < k} \Phi_f^j \not\Rightarrow \neg\{l_0, \ldots l_k\}\}$$

(7)

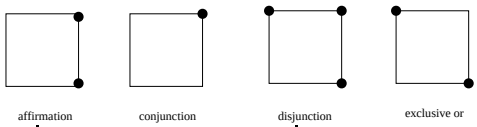
Input: Boolean function f (e.g. represented by its truth table)
Output: A 'power series expansion' g that defines f exactly
  $i := 0$
  $g := \Phi_f^i$
  while $g \not\Leftrightarrow f$ **do**
    $i := i + 1$
    $g := f \cup \Phi_f^i$
  end while

Algorithm POWERSERIES

Clearly, this is a simple variant of CNF, with some redundancies eliminated and conjuncts listed in order of size.

5.2 Concept learning and Feldman's proposal

(8) Considering the classic result again:



f=affirmation	f=conjunction	f=disjunction	f=xor
$\Phi^0_f = \{\neg(\overline{p})\}$	$\Phi^0_f = \{\neg(\overline{p}), \neg(\overline{q})\}$	$\Phi^0_f = \varnothing$ $\Phi^1_f = \{\neg(\overline{pq})\}$	**exercise!**

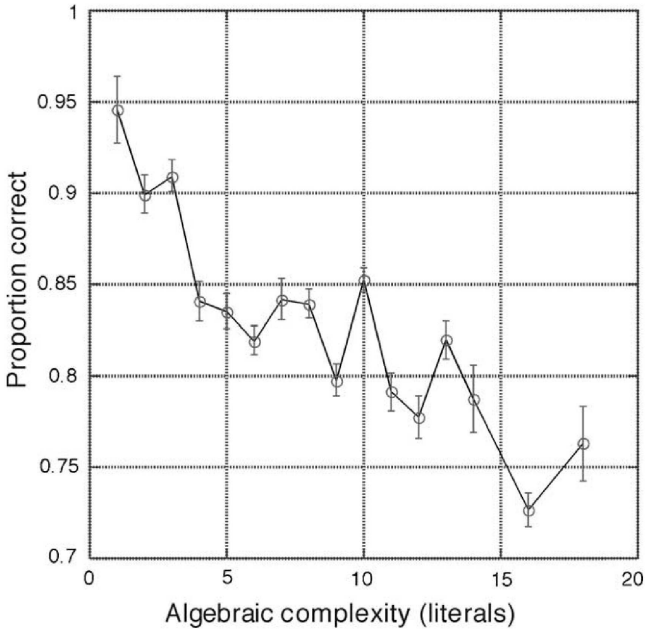
(9) Feldman considers a couple of ideas about weights for each power from $k = 1$ to n. One idea is simply for each element of each Φ^k_f let the weight be the number of literals. The **number of literals in the 'power series' of function** f, the **"algebraic complexity"** is

$$C(f) = \sum_{0 \le k \le n-1} (k+1)|\Phi^k_f|.$$

Applying this to the classic examples above, we find:

f=affirmation	f=conjunction	f=disjunction	f=xor
$C(f) = 1$	$C(f) = 2$	$C(f) = 2$	$C(f) = 4$

In his study of 76 different types of concepts, Feldman finds that this measure correlates significantly with the psychological complexity of a learning task: [5, 6]



So Feldman's proposal is interesting. Not only does it fit the data fairly well, but it might provide a natural framework for understanding 'default-like' simplifications of the data that we sometimes see in human performance as tendencies to stick to low 'powers', $k \le 1$.

There are many other issues to consider: how to deal with 'noise', etc. But let's return now to questions about how this kind of model might relate to our larger problem of learning a simple language.

5.3 The nature of lexical concepts

(10) There are various worries about the applicability of concept learning theories like those above to language learning, at least if we're interested in learning of the sort found humans and other biological organisms. Besides a below, the literature has these other prominent ideas:

a. most lexical concepts have Boolean structure (as phrases also do)

b. most lexical concepts are structured but not Booleanly – they're analogs of prototypes

c. most lexical concepts are structured but not Booleanly – they're analogs of exemplars

d. most lexical concepts do not have structure – they're semantically atomic

e. lexical concepts are fictions of 'folk psychology' that will not play any role in the right theory of language learning; there is nothing useful to be said about meaning in any scientific account of language.

We won't have much to say about 10e – this dispute is so fundamental, the proof will have to be 'in the pudding'. A careful consideration of 10a-10d would take more time than we have, but it is important to think about these basic issues to avoid confusions! We'll consider them very briefly. I'll argue that some points of consensus can be wrestled out of the apparently opposed 10a-10d.

(11) **H: most lexical concepts have Boolean structure (as phrases also do).** Phrases of natural language have a Boolean structure not only in the sense that in very many categories there is a possibility of negation, coordination and disjunction, but also in that combinations of semantic values can be characterized in a Boolean framework. We notice, for example,

[18, 29, 30, 4]

$$
\begin{array}{rcl}
(q \vee r) \wedge p & \leftrightarrow & (q \wedge p) \vee (r \wedge p) \\
\text{Quincy or Roy saw Paul} & \leftrightarrow & \text{Quincy saw Paul or Roy saw Paul} \\
\text{saw Quincy or Roy} & \leftrightarrow & \text{saw Quincy or saw Roy} \\
\text{with Quincy or Roy} & \leftrightarrow & \text{with Quincy or with Roy} \\
\text{description of Quincy or Roy} & \leftrightarrow & \text{description of Quincy or description of Roy}
\end{array}
$$

The logical structure of a subject or object is often inherited in a certain sense. (These expressions denote 'Boolean homomorphisms' – we'll talk more about this when we study how inference works.)

So the hypothesis H stated above is that lexical concepts have Boolean structure in the way that phrases of human languages do. This hypothesis has been very popular with linguists.

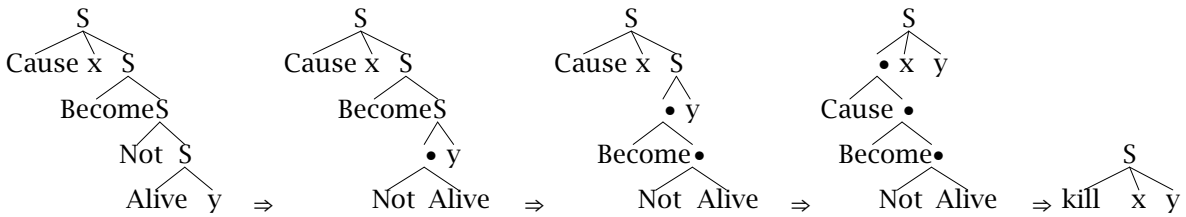
[17, p155]

For example, it has been proposed that

$$
\begin{array}{rcl}
\text{bachelor} & \equiv & \text{(Physical Object)} \wedge \text{(Living)} \wedge \text{(Human)} \wedge \text{(Male)} \wedge \text{(Adult)} \wedge \text{(Never married).} \\
x \text{ chase } y & \equiv & x \text{ follow } y \wedge x \text{ intend } (x \text{ catch } y) \\
x \text{ seeks } y & \equiv & x \text{ try } (x \text{ find } y)
\end{array}
$$

[22, p108]

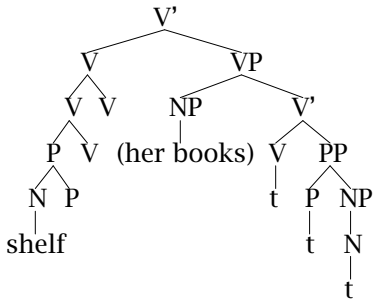
And it is proposed that the verb *kill* might underlyingly be *cause to become not alive*



Many causative verbs have been similarly analyzed:

$$
\begin{array}{rcl}
x \text{ caused } [y \text{ melted}] & \Rightarrow & x \text{ melted } y \\
x \text{ caused } [y \text{ P } z] & \Rightarrow & x \text{ put } y \text{ P } z \\
x \text{ caused } [y \text{ intends } z] & \Rightarrow & x \text{ persuaded } y \text{ to } z \\
x \text{ caused } [y \text{ believes } z] & \Rightarrow & x \text{ persuaded } y \text{ that } z \\
x \text{ caused } [y \text{ to } z] & \Rightarrow & x \text{ forced } y \text{ to } z
\end{array}
$$

It has also been proposed that the verb *shelve* is derived from something like *put on a shelf* by "incorporation of the head noun" *shelf*, leaving unpronounced traces (t) behind: [14, p58]



Similar analyses can be proposed for verbs like *paint, butter, saddle,*…. Others propose that the properties of a lexical item like *bake* are specified by complexes like this: [24, p123]

$$
\left[\begin{array}{l}
\textbf{bake} \\
\text{EVENTSTR} = \left[\begin{array}{c} E_1 = \mathbf{e_1} : \textbf{process} \\ \text{HEAD} = \mathbf{e_1} \end{array}\right] \\
\text{ARGSTR} = \left[\begin{array}{l} \text{ARG1} = \boxed{1}\left[\begin{array}{c} \textbf{animate\_ind} \\ \text{FORMAL} = \textbf{physobj} \end{array}\right] \\ \text{ARG2} = \boxed{2}\left[\begin{array}{c} \textbf{mass} \\ \text{FORMAL} = \textbf{physobj} \end{array}\right] \end{array}\right] \\
\text{QUALIA} = \left[\begin{array}{c} \textbf{state\_change\_lcp} \\ \text{HEAD} = \textbf{bake\_act}(\mathbf{e_1}, \boxed{1}, \boxed{2}) \end{array}\right]
\end{array}\right]
$$

One odd thing about these approaches is that a formal looking notation is used, but without a formal syntax and semantics for the notation, which leaves us to guess about what exactly is meant. But it is typically assumed (and sometimes asserted) that the meanings of the lexical items are given by compositional interpretation of these structures.

(12) **Problems for (11): linguistic evidence.** Substitutions that fail to preserve semantic value: [9, 10]

John caused Mary to die, and it surprised me that he/she did so
John killed Mary, and it surprised me that he/*she did so

John caused Bill to die on Sunday by stabbing him on Saturday
??John killed Bill on Sunday by stabbing him on Saturday

John caused Bill to die on Sunday by swallowing his tongue
≠John killed Bill by swallowing his tongue

John caused the brush to become covered with paint
≠John painted the brush

Where did John put the book on the shelf?
≠Where did John shelve the book?

Where did John put paint on the wall?
≠Where did John paint the wall?

(13) **Problems for (11): psychological evidence.** Predictions about psychological complexity not confirmed!:

[19, 7, 8]

John is a bachelor > John is unmarried
John killed Mary > John kissed Mary
Cats chase mice > Cats catch mice
They seek shelter > They find shelter

Experimental studies show:

morphological negatives < explicit negatives (weak tendency)
'pure definitional negatives' like *kill, bachelor* < explicit negatives

Fodor et al. conclude "it appears practically mandatory to assume that meaning postulates mediate whenever entailment relations turn upon their lexical content." In other words, what decompositional theories are pointing out is things we know about our what our words denote, but these things do not need to be provide a full, nontrivial definition of the word, and they do not need to be reconstructed online whenever the word is recognized.

(14) **H: lexical concepts are (mainly) analogs of prototypes.** This idea has been very popular in the psychology textbooks for more than 25 years:

> *…An important question is whether…natural categories are …structured so that some category members are more central than others …Three major lines of research support this interpretation:*
>
> 1. *People reliably rate some category members as more "typical" than others*
> 2. *When people are asked to list exemplars of a category, they reliably produce some items both earlier and more frequently than others. Furthermore, items that are produced most readily tend to be those that people consider also most typical of the category.*
> 3. *Both of these measures (typicality ratings and frequency of production) predict the speed with with subjects can classify instances as members of a category. For example, people can verify the truth of the sentence* A robin is a bird *more quickly than they can verify* A goose is a bird. [12, p337]

> *There is a wealth of evidence supporting prototype theory over feature theory. Rosch and Mervis (1975) measured family resemblance among instances of concepts such as fruit, furniture, and vehicles by asking participants to list their features. Although some features were given by all participants for particular concepts, these were not technically defining features, as they did not distinguish the concept from other concepts. For example, all participants might say of "birds" that "they're alive", but then so are all other animals. The more specific features that were listed were not shared by all instances of a concept – for example, not all birds fly.* [15, p289]

> *The concepts that people use to categorize all sorts of objects have fuzzy boundaries. For example, we respond "yes" to both "Is a robin a bird?" and "Is a penguin a bird?" But we are significantly faster at responding if the question is about a robin. As we saw…the reason is that a robin better fits our notion of the prototypical "bird" category.* [32, p156]

(15) **H: lexical concepts are (mainly) analogs of examplars.** This is another idea popular in some psychology textbooks.

> *…This approach overlaps with the prototype view in several regards. According to each of these views, you categorize objects by comparing them to a mentally represented "standard." For prototype theory, the standard is the prototype; for exemplar theory, the standard is provided by whatever example of the category comes to mind…a prototype can be though of as the average for a category…but much information is lost in an average. For example, an average, by itself, does not tell you how variable a set is…since people do seem sensitive to variability information, this seems a strike against prototype theory. …Exemplar storage provides an easy way of talking about our knowledge of variability within a category, as well as our sensitivity to patterns of correlations among a category's features. Exemplar storage also provides an easy way of accounting for category pliability and ad hoc categories.* [26, pp280,282,286]

(16) **Challenging the evidence (14-15): analysis vs application.** Everyone agrees that having a concept does not imply that you know whether it applies in every case. For example, knowing [25]

$$\text{prime} \equiv \text{an integer having no integer factors except itself and 1}$$

does not imply that you can answer the question

$$\text{Is 2351 prime?}$$

If I tell you the answer is "yes," you have some evidence, but maybe not conclusive. The same goes

for Boolean concepts like:

$$\text{zorg} \equiv (r \vee q \vee p) \wedge ((r \wedge s) \vee \neg p) \wedge \neg(\neg r \wedge q)$$
$$\text{burf} \equiv r$$

Even given these definitions (not you but) many people would have some trouble saying whether every zorg situation is also burf. The same goes for other familiar concepts like *dog, pencil, mother, water, elm tree,…* There are many situations in which you don't know whether these familiar concepts apply to an object.

Once we recognize that the relation between a concept and evidence about whether the concept applies is not simple, does anything about 'the structure of concepts' follow from the fact that you are faster at answering such questions for some instances of concepts than others? It all depends on your account of how evidence of the applicability of a concept is applied. It seems (14-15) have fallen for an inadequate view of this: the use of "prototypes" and "exemplars" may be relevant to determining applicability sometimes, but that does not mean that the concept is determined by its prototype, nor that prototypes are always used. Obviously: sometimes we determine applicability by doing a calculation, sometimes by asking an expert, sometimes by comparing with past instances,…

(17) **Problems for (14-15): conceptual combination, psychological evidence.**

> *…Concepts can't be prototypes,* pace *all the evidence that everybody who has a concept is likely to have a prototype as well…In a nutshell, the trouble with prototypes is this. Concepts are productive and systematic. Since compositionality is what explains systematicity and productivity, it must be that concepts are compositional. But it's as certain as anything ever gets in cognitive science that concepts don't compose…So, for example, a goldfish is a poorish example of a fish, and a poorish example of a pet, but it's a prototypical example of a pet fish.* [10, pp93-4,102]

The previous argument is the killing one; this one is open-ended. For some debate see [23, 16] and a reply [11, §2].

(18) **H: lexical concepts are (mainly) atomic.** Consider:

- Plainly, it is extremely hard to find very many perfect, non-trivial definitions for lexical items.
- [3] Sustained attempts to define theoretical concepts in observational terms ('positivism') failed, for principled reasons.
- [2, 21, 13, 20] Language learners tend to assume that new words will be non-synonymous with words already learned

5.4 A reasonable view

(19) In human languages, most lexical concepts are semantic atoms

(20) Humans learn logical relations among lexical concepts, much as the decompositional theories say, but (contrary to decompositional theories) these logical relations typically do not provide perfect, nontrivial definitions

(21) It is possible that some lexical items trigger the construction of 'underlying' linguistic structure. This is entirely compatible with (19) if we recognize that the interpretation of these complexes may be (and often is) idiomatic. (We will say much more about idioms later)

(22) Humans remember examples of concepts, and at least in some cases may use these examples in deciding whether a new instance falls under the concept, much as the prototype and exemplar theories say. (Contrary to prototype and exemplar theories, we do not say that the concept is the prototype, or anything like that)

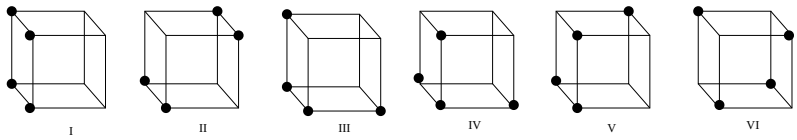
On this view, the learning theories we have been considering for the past 2 weeks relate to filling out the story about (20). The experimental settings in these 'concept learning' studies are carefully devised so that we are testing the recognition of relationships among concepts.

Exercise 2

(1) In standard treatments of Boolean complexity like [31], it is common to let the complexity of a formula (or Boolean circuit) be the number of connectives (or gates). If we count the implicit conjunction symbols and the negations in the series expansion of a Boolean function f, how does this number compare to $C(f)$ defined in (9) on page 35 above?

(2) [a] We provided a (rough) algorithm POWERSERIES on page 1 for computing the 'power series expansion' of a concept, but we did not define a 'learner' based on this representation. Define a 'Feldman learner' that is guaranteed to identify any target function in the limit, and explain whether your learner predicts the 'classic' psychological complexity result shown in (8) on page 35. If not, do you think this is fixable?

(3-**optionally!**) Calculate the series expansion complexity $C(f)$ for each of Shepard's 6 types of functions:



Do these values $C(f)$ correspond to the psychological results we discussed earlier?

(4-**optionally!**) Code up one or more of the algorithms you used in these exercises, so that you don't have to do these things by hand!

[a] **Warning!** This is the first slightly challenging exercise so far! Come see me if you get stuck.
Also note: I record any **optional** exercises you do in the special 'extra credit' column of the grade book!
Some of the optional exercises are also suitable for elaboration into term projects...For example, (4) coding up POWERSERIES in a reasonable way is quite tricky and would be a good project. If you want to take on something like this, talk to me first!
Extra credit is also given for email about typos or improvements I should make in the lecture notes.

References for Lecture 5

[1] BOURNE, L. E. Knowing and using concepts. *Psychological Review 77*, 6 (1970), 546–556.

[2] CLARK, E. V. Meaning and concepts. In *Handbook of Child Psychology*, J. Flavell and E. Markman, Eds. Wiley, NY, 1983, pp. 787–840.

[3] COFFA, J. A. *The Semantic Tradition from Kant to Carnap : To the Vienna Station*. Cambridge University Press, NY, 1993.

[4] DAVEY, B., AND PRIESTLEY, H. *Introduction to Lattices and Order*. Cambridge University Press, NY, 1990.

[5] FELDMAN, J. Minimization of complexity in human concept learning. *Nature 407* (2000), 630–633.

[6] FELDMAN, J. How surprising is a simple pattern? Quantifying 'Eureka!'. *Cognition 93* (2004), 199–224.

[7] FODOR, J., FODOR, J., AND GARRETT, M. The psychological unreality of semantic representations. *Linguistic Inquiry 6* (1975), 515–532.

[8] FODOR, J., GARRETT, M., WALKER, E., AND PARKES, C. Against definitions. *Cognition 8* (1980), 263–367.

[9] FODOR, J. A. Three reasons for not deriving "kill" from "cause to die". *Linguistic Inquiry 1* (1971), 429–438.

[10] FODOR, J. A. *Concepts: Where Cognitive Science Went Wrong*. Clarendon Press, Oxford, 1998.

[11] FODOR, J. A., AND LEPORE, E. *The Compositionality Papers*. Clarendon, Oxford, 2002.

[12] GLASS, A. L., HOLYOAK, K. J., AND SANTA, J. L. *Cognition*. Addison-Wesley, Menlo Park, California, 1979.

[13] GOLINKOFF, R., MERVIS, C., AND HIRSH-PASEK, K. Early object labels: The case for a developmental lexical principles framework. *Journal of Child Language 21* (1990), 125–155.

[14] HALE, K., AND KEYSER, S. J. On argument structure and the lexical expression of syntactic relations. In *The View from Building 20*, K. Hale and S. J. Keyser, Eds. MIT Press, Cambridge, Massachusetts, 1993.

[15] HARLEY, T. *The Psychology of Language, Second Edition*. Psychology Press, New York, 2001.

[16] KAMP, H., AND PARTEE, B. Prototype theory and compositionality. *Cognition 57* (1995), 129–191.

[17] KATZ, J. J. *The Philosophy of Language*. Harper and Row, NY, 1966.

[18] KEENAN, E. L., AND FALTZ, L. M. *Boolean Semantics for Natural Language*. Reidel, Dordrecht, 1985.

[19] KINTSCH, W. *The Representation of Meaning in Memory*. Lawrence Erlbaum, Hillsdale, New Jersey, 1974.

[20] LIDZ, J., GLEITMAN, H., AND GLEITMAN, L. R. Kidz in the 'hood: Syntactic bootstrapping and the mental lexicon. In *Weaving a Lexicon*, D. Hall and S. Waxman, Eds. MIT Press, Cambridge, Massachusetts, 2004, pp. 603–636.

[21] MARKMAN, E. M. *Categorization and Naming in Children: Problems of Induction*. MIT Press, Cambridge, Massachussetts, 1989.

[22] MCCAWLEY, J. D. English as a VSO language. *Language 46* (1970), 286–299. Reprinted in *The Logic of Grammar*, edited by D. Davidson and G. Harman, Dickenson: Encino, California.

[23] OSHERSON, D., AND SMITH, E. On the adequacy of prototype theory as a theory of concepts. In *Readings in Cognitive Science*, A. Collins and E. Smith, Eds. Morgan Kaufman, San Mateo, California, 1988.

[24] PUSTEJOVSKY, J. *The Generative Lexicon*. MIT Press, Cambridge, Massachusetts, 1995.

[25] PUTNAM, H. The meaning of 'meaning'. In *Mind, Language and Reality: Philosophical Papers, Volume 2*. Cambridge University Press, NY, 1975.

[26] REISBERG, D. *Cognition: Exploring the Science of Mind, Second Edition*. Norton, NY, 2001.

[27] ROSCH, E., AND MERVIS, C. Family resemblances: Studies in the internal structure of categories. *Cognitive Psychoogy 7* (1975), 573–605.

[28] SHEPARD, R., HOVLAND, C., AND JENKINS, H. Learning and memorization of classifications. *Psychological Monographs: General and Applied 75*, 13 (1961), 1–42.

[29] VAN BENTHEM, J. *Essays in Logical Semantics*. Reidel, Dordrecht, 1986.

[30] VAN BENTHEM, J. Review of Keenan and Faltz, B*oolean semantics for natural language*. *Language 62* (1986), 908–914.

[31] WEGENER, I. *The Complexity of Boolean Functions*. Wiley, NY, 1997.

[32] WHITNEY, P. *The Psychology of Language*. Houghton Mifflin, NY, 1998.

6 The simplest languages: recognizing the symbols

6.1 Summary

(1) We are considering the following trivial fragment,

grammar G: S ::= $p_1 \mid p_2 \mid \ldots \mid p_n$ (each expression is just given, in this "lexicon")

semantics: $\mu : L(G) \rightarrow R$ (each expression just assigned some meaning)

inference: so far, arbitrary

(2) How could such a simple language could be learned?

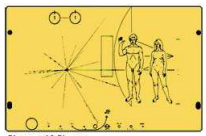
a. how to learn what the expressions are?

b. how to learn the meanings of these expressions?

Now we turn now to a.

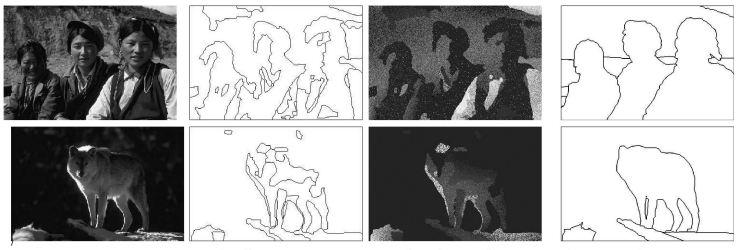
6.2 Identifying signals and components

(3) What is SETI (`www.seti.org`) looking for? What are we broadcasting for others to find?



Pioneer 10 Plaque

(4) What parts do we find in visual scenes? [5, 22, 23]



a. Input image b. Segmentation c. Synthesized image d. Manual segmentation

(5) How do we decide when we've segmented properly? [2]

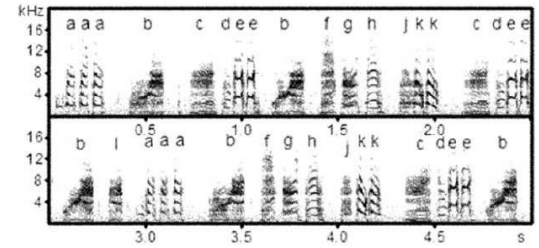
```
EMUFPHZLRFAXYUSDJKZLDKRNSHGNFIVJ
YQTQUXQBQVYUVLLTREVJYQTMKYRDMFD
VFPJUDEEHZWETZYVGWHKKQETGFQJNCE
GGWHKK?DQMCPFQZDQMMIAGPFXHQRLG
TIMVMZJANQLVKQEDAGDVFRPJUNGEUNA
QZGZLECGYUXUEENJTBJLBQCRTBJDFHRR
YIZETKZEMVDUFKSJHKFWHKUWQLSZFTI
HHDDDUVH?DWKBFUFPWNTDFIYCUQZERE
EVLDKFEZMOQQJLTTUGSYQPFEUNLAVIDX
FLGGTEZ?FKZBSFDQVGOGIPUFXHHDRKF
FHQNTGPUAECNUVPDJMQCLQUMUNEDFQ
ELZZVRRGKFFVOEEXBDMVPNFQXEZLGRE
DNQFMPNZGLFLPMRJQYALMGNUVPDXVKP
DQUMEBEDMHDAFMJGZNUPLGEWJLLAETG
```

```
ABCDEFGHIJKLMNOPQRSTUVWXYZABCD
AKRYPTOSABCDEFGHIJLMNQUVWXZKRYP
BRYPTOSABCDEFGHIJLMNQUVWXZKRYPT
CYPTOSABCDEFGHIJLMNQUVWXZKRYPTO
DPTOSABCDEFGHIJLMNQUVWXZKRYPTOS
ETOSABCDEFGHIJLMNQUVWXZKRYPTOSA
FOSABCDEFGHIJLMNQUVWXZKRYPTOSAB
GSABCDEFGHIJLMNQUVWXZKRYPTOSABC
HABCDEFGHIJLMNQUVWXZKRYPTOSABCD
IBCDEFGHIJLMNQUVWXZKRYPTOSABCDE
JCDEFGHIJLMNQUVWXZKRYPTOSABCDEF
KDEFGHIJLMNQUVWXZKRYPTOSABCDEFG
LEFGHIJLMNQUVWXZKRYPTOSABCDEFGH
MFGHIJLMNQUVWXZKRYPTOSABCDEFGHI
```

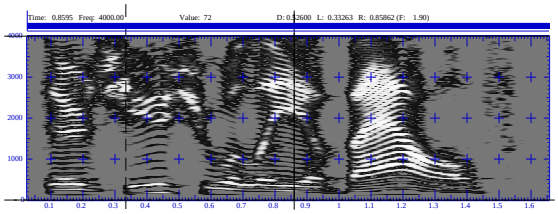
```
ENDYAHROHNLSRHEOCPTEOIBIDYSHNAIA
CHTNREYULDSLLSLLNOHSNOSMRWXMNE
TPRNGATIHNRARPESLNNELEBLPIIACAE
WMTWNDITEENRAHCTENEUDRETNHAEOE
TFOLSEDTIWENHAEIOYTEYQHEENCTAYCR
EIFTBRSPAMHHEWENATAMATEGYEERLB
TEEFOASFIOTUETUAEOTOARMAEERTNRTI
BSEDDNIAAHTTMSTEWPIEROAGRIEWFEB
AECTDDHILCEIHSITEGOEAOSDDRYDLORIT
RKLMLEHAGTDHARDPNEOHMGFMFEUHE
ECDMRIPFEIMEHNLSSTTRTVDOHW?OBKR
UOXOGHULBSOLIFBBWFLRVQQPRNGKSSO
TWTQSJQSSEKZZWATJKLUDIAWINFBNYP
VTTMZFPKWGDKZXTJCDIGKUHUAUEKCAR
```

```
NGHIJLMNQUVWXZKRYPTOSABCDEFGHIJL
OHIJLMNQUVWXZKRYPTOSABCDEFGHIJL
PIJLMNQUVWXZKRYPTOSABCDEFGHIJLM
QJLMNQUVWXZKRYPTOSABCDEFGHIJLMN
RLMNQUVWXZKRYPTOSABCDEFGHIJLMNQ
SMNQUVWXZKRYPTOSABCDEFGHIJLMNQU
TNQUVWXZKRYPTOSABCDEFGHIJLMNQUV
UQUVWXZKRYPTOSABCDEFGHIJLMNQUVW
VUVWXZKRYPTOSABCDEFGHIJLMNQUVWX
WVWXZKRYPTOSABCDEFGHIJLMNQUVWXZ
XWXZKRYPTOSABCDEFGHIJLMNQUVWXZK
YXZKRYPTOSABCDEFGHIJLMNQUVWXZKR
ZZKRYPTOSABCDEFGHIJLMNQUVWXZKRY
 ABCDEFGHIJKLMNOPQRSTUVWXYZABCD
```

[19] (6) Where are the signals, and the segments of the signal here?



[15, 6, 13]



6.3 Identifying utterance parts

(7) Given the phonemes, how do we find the meaningful units in utterances?

[10] Zellig Harris: look for sequences of events which are more predictable than you would expect, segmenting at points where the entropy hits a maximum:

> *The basic procedure is to ask how many different phonemes occur (in various utterances) after the first n phonemes in some test utterance…We segment the test utterance at the points where the number of successors reaches a peak.* (p.192)

A simplistic measure of entropy at each point in an utterance is given by the number of possible sounds that can follow, and so the idea is to segment at the "local maxima" of this function; the entropy decreases inside a word.

(8) Let's consider one of Harris' examples,

/hizkwɪkɹ̩/.

[4] We can analyze this using the following inventory of sounds from the Fromkin textbook:

24 consonants voice?			bilabial	labio-dental	dental	alveolar	palato-alveolar	palatal	velar	glottal
stops	-		/p/			/t/	/ʧ/		/k/	
	+		/b/			/d/	/ʤ/		/g/	
fricatives	-			/f/	/θ/	/s/	/ʃ/			/h/
	+			/v/	/ð/	/z/	/ʒ/			
nasals	+		/m/			/n/			/ŋ/	
approximant	+	lateral				/l/				
	+	central	/w/			/r/		/j/		

11 vowels	front unrounded	central unrounded	back unrounded	back rounded
upper high	/i/			/u/
lower high	/ɪ/			/ʊ/
upper mid	/e/	/ə/		/o/
lower mid	/ɛ/		/ʌ/	
low	/æ/		/a/	

3 diphthongs
/aɪ/
/aʊ/
/oɪ/
1 syllabic consonant
/ɻ̩/

Now consider how many sounds can come after [h], how many sounds can come after /hi/, after /hiz/, and so on through the whole sequence. For /h/, we can have the following continuations:

sounds that can follow [h] in English:

	as in:
j	humans act like simians
ɪ	his ship's in
i	he's out
ɛ	hell, what's the use?
æ	have you got it
a	Harping on it won't help
o	hope is all we have
ə	Havanna
ʌ	hunting and fishing are regulated
u	who is it?
ʊ	hooks and ladders are needed
total: 11	

After [hi], the situation is different. Here, I can think of easy examples for all of the 39 sounds except the ones that never (or almost never) begin words: namely ŋ and ʒ, so the total is 37. The count for [hiz] is the same: 37. For [hizk], though, no stop or affricate or fricative or nasal can follow, so I count just these:

sounds that can follow [hizk] in English:

	as in:		as in:
w	he's quiet	ɹ	he's cranky now
l	he's clinching it	j	he's cute for a boy
i	he's keeping it	u	he's cool
ɪ	he's kicking	ʊ	he's cooking
e	he's capable	ə	he's commanding
o	he's coding the message	ɛ	he's kept it
æ	he's cackling like a bird	ʌ	he's coming home
ɻ̩	he's colonel of the regiment	a	he's cocking the gun
oɪ	he's coy	aɪ	he's kite flying
total: 18			

sounds that can follow [hizkw] in English:

	as in:		as in:
aɪ	he's quiet	ɛ	he's quelling the riot
e	he's quaking	æ	he's quacking
i	he's queen of the ball	a	he's quaffing his ale
ɪ	he's quicker	o	he's quoting Marx
ɻ̩	he's quirky		
total: 9			

We can list these results in a table like this (leaving a few ?s for you to consider).

	h	i	z	k	w	ɪ	k	ɻ̩
counts of following sounds:	11	37	37	18	9	?	?	?

If we put breaks at the points where the successor count starts to decrease – after the /z/ – we get an appropriate break. If we put breaks at the points where the successor count starts to decrease or stays the same, we get the two breaks indicated above.

(9) I just filled these tables of data quickly, and surely missed some possibilities. How could we give this simple proposal a more serious test? We take some first steps, partly just to have the practice for use with other proposals.

6.4 GNU text tools for corpus analysis

(10) Here I use the GNU text tools `wc, more, sort, uniq, gawk, paste,` and the GNU C compiler `gcc` and lexer `flex` to dissect the Mitton dictionary. These are standard in linux, available through cygwin `www.cygwin.com` on Windows, and using `fink` on MacOS-X.

[17] (11) Mitton's publically available phonetic dictionary is from the *Oxford Advanced Learner's Dictionary*, and provides a kind of phonetic transcription for British English.

1. **edit out extra stuff at beginning and end of the file.** I also used global substitutions to replace all blanks in entries by -. I think most people do this kind of thing with `emacs` or `xemacs` or `vi`.

   ```
   % emacs ascii_0710-1.txt
   ```

   I saved the result in `mitton.txt`

2. **to see how many lines, words, and bytes are in the file**

   ```
   % wc mitton.txt
     70646  283547 9113334 mitton.txt
   ```

   This shows that the file has 70646 lines in it, after the beginning and end are removed. We can use `more` to look at the beginning of the file, like this,

   ```
   % more mitton.txt
   'em                  @m                   Qx$                  1
   'neath               niT                  T-$                  1
   'shun                SVn                  W-$                  1
   'twas                twOz                 Gf$                  1
   'tween               twin                 Pu$,T-$              1
   'tween-decks         'twin-deks           Pu$                  2
   'twere               tw3R                 Gf$                  1
   'twill               twIl                 Gf$                  1
   'twixt               twIkst               T-$                  1
   'twould              twUd                 Gf$                  1
   'un                  @n                   Qx$                  1
   A                    eI                   Ki$                  1
   A's                  eIz                  Kj$                  1
   A-bombs              'eI-bOmz             Kj$                  2
   A-level              'eI-levl             K6%                  3
   A-levels             'eI-levlz            Kj%                  3
   AA                   ,eI'eI               Y>%                  2
   ABC                  ,eI,bi'si            Y>%                  3
   --More--
   q
   ```

   The second column is a phonetic transcription of the word spelled in the first column. (Columns 3 and 4 contain syntactic category, number of syllables.)

3. The phonetic transcription has notations for 43 sounds. My guesses on the translation:

Mitton	IPA	example	Mitton	IPA	example
i	i	bead	N	ŋ	sing
I	ɪ	bid	T	θ	thin
e	ɛ	bed	D	ð	then
&	æ	bad	S	ʃ	shed
A	a	bard	Z	ʒ	beige
0(zero)	ɒ	cod	O	ɛə	cord
U	ʊ	good	u	u	food
p	p		t	t	
k	k		b	b	
d	d		g	g	
V	ʌ		m	m	
n	n		f	f	
v	v		s	s	
z	z		3	ɜ	bird
r	r		l	l	
w	w		h	h	
j	j		@	ə	about
eI	eɪ	day	@U	oʊ	go
aI	aɪ	eye	aU	aʊ	cow
oI	oɪ	boy	I@	ɪə	beer
e@	ɛə	bare	U@	ʊə	tour
R	ɹ	far			

The phonetic entries also mark primary stress with an apostrophe, and secondary stress with an comma. Word boundaries in compound forms are indicated with a +, unless they are spelled with a hyphen or space, in which case the phonetic entries do the same.

4. **get rid of columns 3 and 4, since we will not be using them for the moment.**

```
% gawk '{print $1 "\t" $2};' < mitton.txt > mitton12.txt
% gawk '{print $2};' < mitton.txt > mitton2.txt
```

We can check how many lines, words and characters in the resulting files:

```
% wc mitton12.txt
 70646  141292 1347453 mitton12.txt
% wc mitton2.txt
 70646  70646 680908 mitton2.txt
```

And we can take a look at the beginning at the file with `more`, typing `q` to end it:

```
% more mitton2.txt
@m
niT
SVn
twOz
twin
'twin-deks
tw3R
twIl
twIkst
twUd
@n
eI
eIz
'eI-bOm
'eI-bOmz
'eI-levl
'eI-levlz
,eI'eI
,eI,bi'si
--More--
q
```

5. **put every symbol on a new line.** This is slightly tricky because there are several 2-character symbols, and we don't want to split them up. This kind of translation is often done with `perl` or `python` or even `ocaml` – really you can use almost any program to build a finite transducer for this task, but I find it easiest to use flex. So I wrote this simple flex program

```
%{ /* col2tokens.flex */
%}

%%
eI      {printf("eI\n");}
@U      {printf("@U\n");}
aI      {printf("aI\n");}
aU      {printf("aU\n");}
oI      {printf("oI\n");}
I@      {printf("I@\n");}
e@      {printf("e@\n");}
U@      {printf("U@\n");}
i       {printf("i\n");}
N       {printf("N\n");}
I       {printf("I\n");}
T       {printf("T\n");}
e       {printf("e\n");}
D       {printf("D\n");}
&       {printf("&\n");}
S       {printf("S\n");}
A       {printf("A\n");}
Z       {printf("Z\n");}
O       {printf("O\n");}
O       {printf("O\n");}
U       {printf("U\n");}
u       {printf("u\n");}
p       {printf("p\n");}
t       {printf("t\n");}
k       {printf("k\n");}
b       {printf("b\n");}
d       {printf("d\n");}
g       {printf("g\n");}
V       {printf("V\n");}
m       {printf("m\n");}
n       {printf("n\n");}
f       {printf("f\n");}
v       {printf("v\n");}
s       {printf("s\n");}
z       {printf("z\n");}
3       {printf("3\n");}
r       {printf("r\n");}
l       {printf("l\n");}
w       {printf("w\n");}
h       {printf("h\n");}
j       {printf("j\n");}
@       {printf("@\n");}
R       {printf("R\n");}
'       {} /* remove primary stress mark */
,       {} /* remove secondary stress mark */
\+      {} /* remove compounding + */
-       {} /* remove compounding + */
\n      {printf("\n");}
.       {printf("%s",yytext);}

%%
int
main()
        {
        yylex();
        return 0;
        }
```

Flex will use the first match in this list of rules, and so we will always get the 2-symbol tokens where possible. Then I compile this code with the following commands

```
% flex col2tokens.flex
% gcc -c lex.yy.c
% gcc -o col2tokens lex.yy.o -lfl
```

Then I can apply this to `mitton2.txt` like this:

```
% col2tokens < mitton2.txt > mitton2tokens.txt
```

We can look at the beginning of the resulting file with `more`, typing `q` to end the display:

```
% more mitton2tokens.txt
@
m

n
i
T

S
V
n

t
w
0
z

t
w
i
--More--
q
```

6. **We can use tail to print out all except the first line, like this:**

```
% tail +2 mitton2tokens.txt | more
m

n
i
T

S
V
n

t
w
0
z

t
w
i
n
--More--
q
% tail +2 mitton2tokens.txt > mitton2tokens1.txt
```

7. **We can then paste the two files together, like this:**

```
% paste mitton2tokens.txt mitton2tokens1.txt > mitton2bigrams.txt
% more mitton2bigrams.txt
@       m
m
        n
n       i
i       T
T
        S
S       V
V       n
n
        t
t       w
w       0
0       z
--More--
q
% wc mitton2bigrams.txt
 571829 1002525 2225671 mitton2bigrams.txt
```

8. **We can now sort (in "dictionary order") and count these bigrams, like this:**

```
% sort -d mitton2bigrams.txt > bigrams.srt
% more bigrams.srt
@
@
@
@
@
@
@
@
@
@
--More--
q
% uniq -c bigrams.srt > bigrams.cnt
% more bigrams.cnt
    746 @
      4 &
      1
   1794         @
   1338         &
    458         0
    306 0       b
--More--
q
```

We can see here that @ ends 746 words, and begins 1794 words. And so on…

And if we sort the counts numerically in reverse order, we can get a list of the bigrams from most to least frequent:

```
% sort -nr bigrams.cnt > bigrams.nr
% more bigrams.nr
  15723 z
   8454 s
   8272         s
   7979 I       N
   7356 t       I
   7169 s       t
   7030 d
   6858 l       I
   6444 r       I
   6440 t
   6369         k
   6315 N
   5735 @       n
   5602 I
   5574         p
   5510         d
--More--
q
```

For Harris, all we need is a sorted list of bigrams without the counts:

```
% uniq bigrams.srt > bigrams
% more bigrams
@
&

        @
        &
        0
0       b
0       d
0       D
--More--
q
```

9. **Now let's do the previous exercise again, using corpus data.** We considered the sequence

/hizkwɪkɻ̩/.

Now let's just count successors of each sound as the number of bigrams beginning with that sound. First, in case you are not used to the Mitton notation yet, we can look up the Mitton notation for these words using `grep`:

```
% grep quicker mitton12.txt
quicker 'kwIk@R
% grep "he's" mitton12.txt
he's    hiz
she's   Siz
```

To see how many phones follow /h/, we can use `grep` again:

```
% grep "^h" bigrams
h       @
h       &
h       0
h       3
h       A
h       aI
h       aU
h       e
h       e@
h       eI
h       i
h       I
h       I@
h       j
h       O
h       oI
h       u
h       U
h       @U
h       U@
h       V
h       w
%grep "^h" bigrams | wc -l
22
```

So we can begin filling in this table again, based on the corpus:

	h	i	z	k	w	ɪ	k	r̥
counts of following sounds:	22	?	?	?	?	?	?	?

Now looking at what follows /i/:

```
% grep "^i" bigrams
i
i       @
i       &
i       0
i       3
i       A
i       aI
i       aU
i       b
i       d
i       D
i       e
i       eI
i       f
i       g
i       h
i       i
i       I
i       j
i       k
i       l
i       m
i       n
i       N
i       O
i       p
i       r
i       s
i       S
i       t
i       T
i       @U
i       v
i       V
i       w
i       z
i       Z
% grep "^i" bigrams | wc -l
37
```

Can that really be true? What word has i followed by eI?

```
% grep ieI mitton12.txt
d_eshabill_e    ,deIz&'bieI
```

Ah, a borrowing.

deshabille/dishabille
1. The state of being partly undressed, or dressed in a negligent or careless style; undress. Usually in phr. in dishabille (= Fr. en déshabillé).
2. concr. A garment worn in undress; a dress or costume of a negligent style.

So we continue to fill in the table.[1] Continuing in this way,

```
% grep "^z" bigrams | wc -l
39
% grep "^k" bigrams | wc -l
39
% grep "^w" bigrams | wc -l
19
% grep "^I" bigrams | wc -l
58
% grep "^@" bigrams | wc -l
64
% grep "^R" bigrams | wc -l
1
```

	h	i	z	k	w	I	k	@	R
counts of following sounds:	22	37	39	39	19	58	39	64	1

6.5 Doing better than Harris 1955

(12) Rather than considering all of Harris's variations on the basic theme, let's jump ahead 50 years and look at what Goldsmith proposes just recently. First, he reflects on Harris' idea [8]

> *Harris proposed that peaks in successor frequency would be suitable detectors for the discovery of morpheme breaks. As Hafer and Weiss (1974) note, Harris's apparent proposal is actually a family of closely related proposals, and none of them works anywhere close to perfectly, for various reasons...*[For example] *while consonants tend to follow vowels, there is a strong tendency for the successor frequency to be larger after a vowel than after a consonant within the first 3 letters of a word, and hence for this algorithm to find a (spurious) morpheme break after any vowel in the first 5 letters of a word.*

He then provides an algorithm for segmenting written English (and similar languages), which uses a Harris-like criterion in its first step, with the following main steps:

[1] Since the bigrams listed above show that /i/ can end a word, maybe should we assume that it can in principle be followed by any sound that can begin a word?

(13)

Input: A lexicon of sequences of characters
Output: A lexicon with 'signatures'

1. Segment at successor peaks after $i > 5$ characters
   and earlier if the successor frequency $sf(i-1) = sf(i+1) = 1$
2. Compute *signatures*: maximal sets of 1st pieces (stems)
   with common 2nd pieces (suffixes). E.g.

{...mainly nouns...}	$\{\emptyset, s\}$
{...mainly verbs...}	$\{\emptyset, ed, ing, s\}$
{...mainly adjectives...}	$\{\emptyset, er, est, ly\}$

3a. Eliminate signature ⟨stems,suffixes⟩ if |stems| < 3
3b. Eliminate signature if |stems| < 25 and $|\{x \in$suffixes,$|x| > 1\}| < 2$
4. In each signature, if entropy(stem-final chars)< 1.4
   then move stem chars to suffixes if that decreases entropy

⟨{...Xis ...},$\{m, t\}$⟩ ⇒ ⟨{...X ...},$\{ism, ist\}$⟩

5-8. Extend known suffixes, signatures, stems
9. Find singleton signatures when that is most probable analysis
10. Find single-char allomorphy. E.g. $e \rightarrow \emptyset / \_\_\_\_ + ing$

⟨{...lov ...},$\{er, ing, e, es, ed\}$⟩

Algorithm GOLDSMITH

(14) The Goldsmith algorithm is rather complex, and has many language-specific and orthography-specific parameters.

- Are these particulars and parameters right? Are Goldsmith's criteria of accuracy appropriate?
- Do the algorithm have a phonetic analog? How is it determined?
- Are the particulars and parameters of the algorithm valid in the sense that an English user would deploy them on new words, or do they just characterize accidental properties that could easily shift?
- What is the principle that determines how are they set, and how could a learner figure this out? [7]

Goldsmith says,

> *The underlying model that is utilized invokes the principles of the minimum description length (MDL) framework (Rissanen 1989), which provides a helpful perspective for understanding the goals of traditional linguistic analysis. MDL focuses on the analysis of a corpus of data that is optimal by virtue of providing both the most compact representation of the data and the most compact means of extracting that compression from the original data. It thus requires both a quantitative account whose parameters match the original corpus reasonably well (in order to provide the basis for a satisfactory compression) and a spare, elegant account of the overall structure.*

$$\left\{ \begin{array}{ccc} laughed & laughing & laughs \\ walked & walking & walks \\ jumped & jumping & jumps \end{array} \right\} \quad \text{total: 57 characters}$$

$$\left\{ \begin{array}{c} laugh \\ walk \\ jump \end{array} \right\} \left\{ \begin{array}{c} ed \\ ing \\ s \end{array} \right\} \quad \text{total: 19 characters}$$

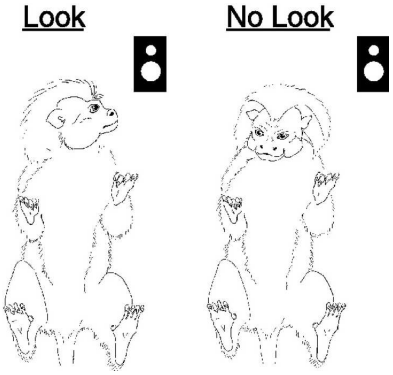
(what is the cost, the "size" of associating two sets of various sizes, like this?)

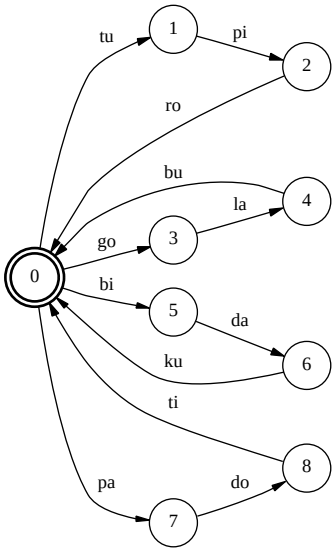
(15) Many learning algorithms use the MDL idea, restricted to a particular measure of description complexity. We saw this idea already in Feldman's proposal, and we will see it again. The choice of the appropriate measure for the domain is analogous to the choice of priors in Bayesian theories. [1, 14]

6.6 Even monkeys and human infants can do it

[12, 21, 11, 3, 16]

(16) In a series of experiments, Hauser and others have shown that word- and sound-transition probabilities are noticed by 7-month-old humans and by capuchin monkeys. In a number of studies they used a spontaneous looking paradigm, playing the monkeys 20 minutes of speech generated randomly by a machine like the one below:

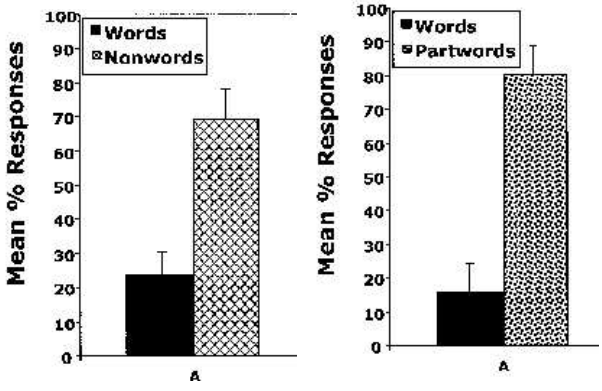




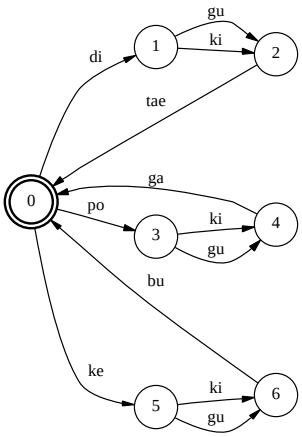
Words: tupiro, golabu, bidaku, padoti
Test words: tupiro, golabu
Test non-words: dapiku, tilado
Test part-words: tibida, kupado

(Hauser et al 2001)
20 mins; 1 min next day; test

The monkeys, like human infants, did notice the difference between test words and non-words, and between test words and test part-words (which appeared in the input but with pauses between):

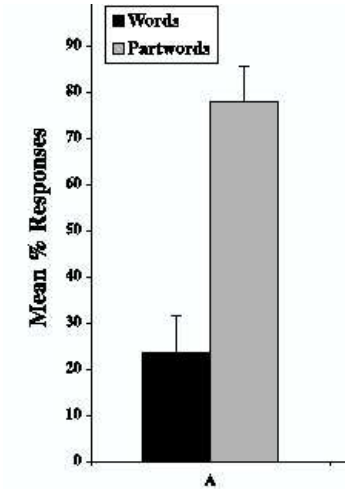


(17) This gets pushed another step in the following study… Studies of monkeys in the wild suggest that they may use 2-constituent signals, but Newport has these more interesting studies of tamarins learning word structure. [24, 18]



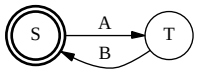
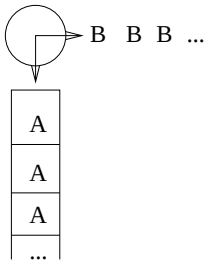
Words: di_tae, po_ga, ke_bu
filled with: ki, gu

(Newport et al 2004)
21 mins; 2 min next day; test



(Adult humans are not so good at this task, but can do the similar task with discontinuous segments)

(18) One more…

Reg: $(AB)^n$		$S \rightarrow A\ T$ $T \rightarrow B\ S$ $S \rightarrow \epsilon$ $A \rightarrow$ ba | di | yo | tu | la | mi | no | wu (♂) $B \rightarrow$ pa | li | mo | nu | ka | bi | do | gu (♀)
CF: A^nB^n		$S \rightarrow A\ S\ B$ $S \rightarrow \epsilon$ $A \rightarrow$ ba | di | yo | tu | la | mi | no | wu (♂) $B \rightarrow$ pa | li | mo | nu | ka | bi | do | gu (♀)
tested n=2,3		

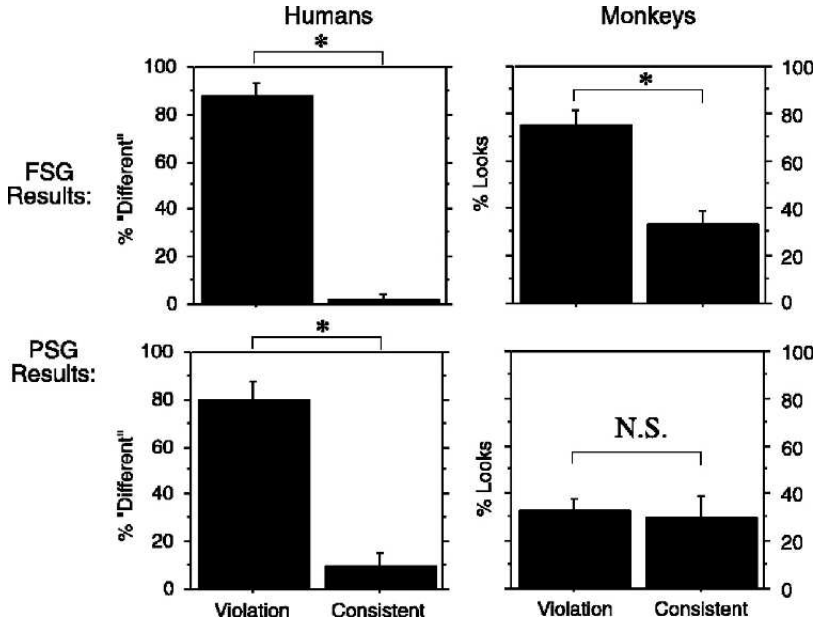
If

$A \rightarrow$ ba | di | yo | tu | la | mi | no | wu
$B \rightarrow$ pa | li | mo | nu | ka | bi | do | gu

Then

$\lvert(AB)^2\rvert = \lvert A^2B^2\rvert =$	4096	
$\lvert(AB)^3\rvert = \lvert A^3B^3\rvert =$	262144	
total # strings of lengths 4, 6 =	266240	
# strings used in training, 20mins =	60,	randomly generated
# strings used in test =	4,	randomly generated

But also: A → ♂, and B → ♀, …



Given the male/female voice issue (and even setting aside some other methodological issues that have been raised), it will be hard to know what to make of results about these stimuli

Exercise 3

(Optional-1) If you have never used the GNU tools described in §6.4, you should do a project that will involve them. There are many possibilities. See me.

(Optional-2) Apply the MDL idea in some domain you are interested in.

(Optional-3) Get Goldsmith's code and see if you can adapt it to the Mitton lists.

References for Lecture 6

[1] DE MARCKEN, C. The unsupervised acquisition of a lexicon from continuous speech. Massachusetts Institute of Technology, Technical Report, A.I. Memo 1558, 1995.

[2] DUNIN, E. Frequently-asked questions about Kryptos. *http://elonka.com/kryptos/faq.html* (2005).

[3] FITCH, W. T., AND HAUSER, M. D. Computational constraints on syntactic processing in a nonhuman primate. *Science 303*, 5656 (2004), 377–380.

[4] FROMKIN, V., Ed. *Linguistics: An Introduction to Linguistic Theory*. Basil Blackwell, Oxford, 2000.

[5] GEMAN, S., POTTER, D. F., AND CHI, Z. Composition systems. *Quarterly of Applied Mathematics 60* (1998), 707–736.

[6] GLASS, J. R. A probabilistic framework for segment-based speech recognition. *Computer Speech and Language 17* (2003), 137–152.

[7] GOLDSMITH, J. Unsupervised learning of the morphology of a natural language. *Computational Linguistics 27*, 2 (2001), 153–198.

[8] GOLDSMITH, J. An algorithm for the unsupervised learning of morphology. *Forthcoming* (2004).

[9] HAFER, M. A., AND WEISS, S. F. Word segmentation by letter successor varieties. *Information Storage and Retrieval 10* (1974), 371–385.

[10] HARRIS, Z. S. From phoneme to morpheme. *Language 31* (1955), 190–222.

[11] HAUSER, M. D., NEWPORT, E. L., AND ASLIN, R. N. Segmentation of the speech stream in a non-human primate: statistical learning in cotton-top tamarins. *Cognition 78*, 1 (2001), B53–B64.

[12] HAUSER, M. D., WEISS, D., AND MARCUS, G. Rule learning by cotton-top tamarins. *Cognition 86*, 1 (2002), B15–B22.

[13] JOHNSON, E. K., AND JUSCZYK, P. W. Word segmentation by 8 month-olds: when speech cues count more than statistics. *Journal of Memory and Language 44* (2001), 548–567.

[14] LI, M., AND VITÁNYI, P. Minimum description length induction, Bayesianism and Kolmogorov complexity. In *1998 IEEE International Symposium on Information Theory* (MIT, Cambridge, 1998).

[15] LUCE, P. A., AND LARGE, N. R. Phonotactics, neighborhood density, and entropy in spoken word recognition. *Language and Cognitive Processes 16* (2001), 565–581.

[16] MINTZ, T., NEWPORT, E., AND BEVER, T. Distributional regularities of grammatical categories in speech to infants. In *Proceedings of the North Eastern Linguistic Society, NELS 25* (1995).

[17] MITTON, R. Oxford Advanced Learner's Dictionary of Current English: expanded 'computer usable' version. 1992.

[18] NEWPORT, E. L., AND ASLIN, R. N. I. statistical learning of non-adjacent dependencies. *Cognitive Psychology 48* (2004), 127–162.

[19] OKANOYA, K. The Bengalese Finch: A window on the behavioral neurophysiology of birdsong syntax. *Annals of the New York Academy of Sciences 1016* (2004), 724–735.

[20] RISSANEN, J. *Stochastic Complexity in Statistical Inquiry*. World Scientific Publishing, Singapore, 1989.

[21] SAFFRAN, J. R., ASLIN, R. N., AND NEWPORT, E. L. Statistical learning by 8-month old infants. *Science 274* (1996), 1926 1928.

[22] TU, Z., CHEN, X., YUILLE, A., AND ZHU, S.-C. Image parsing: Unifying segmentation, detection, and object recognition. *International Journal of Computer Vision* (2005), 37–72.

[23] TU, Z., AND ZHU, S.-C. Parsing images into regions, curves, and curve groups. *Forthcoming* (2005).

[24] ZUBERBÜHLER, K. A syntactic rule in forest monkey communication. *Animal behavior 419* (2002), 920–922.

7 The simplest languages: loose ends

7.1 Summary

(1) We are considering the following trivial fragment,

grammar G: S ::= p_1 | p_2 | ...| p_n (each expression is just given, in this "lexicon")

semantics: $\mu : L(G) \rightarrow R$ (each expression just assigned some meaning)

inference: so far, arbitrary

(2) How can such a simple language be learned? There are many open questions about how this could work. The picture developed so far is something like this:

a. how to learn what the expressions are? Roughly: candidate expressions may stand out as relatively low entropy sequences; Harris suggests segmented at points where entropy increases; Goldsmith and other suggest that cutting at points of high entropy also reduced "description length"

b. how to learn the meanings of these expressions? Roughly: for some candidate expressions (identified in step 2a), it may be possible to determine (typically not a perfect definition but) some of the conditions in which the expression applies. Feldman proposes a metric according to which people learn simple functions most easily.

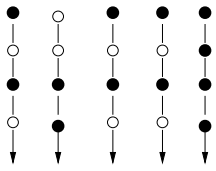
(3) Our simple language in (1) has no morphology or syntax, but there could still be a string of meaningful signals. So if English were like this, we could picture the learner's predicament with a "cartoon" like this,

observed signal: ðədagʧestðəkæt

inferred entropy of the signal: ↑ ↘ ↑ ↘ ↑ ↘ ↑ ↘ ↑ ↘

inferred segmentation: |ðə|dag|ʧest|ðə|kæt

observed salient properties of the situation:



inferred form-meaning hypotheses: … … … … TIGER→kæt

4-LEGGED←kæt

…

We show only one of the target form-meaning representations, but these would be developing for all words, and they will involve more than just entailment relations among concepts of the sort shown

here (there will often be remembered examples, etc). Also, this cartoon leaves out the complicating factor of 'negative examples', which may sometimes be available.

(4) **We do not think English is learned this way, since English is not like (1)!** The rest of the class will be about this point.

(5) Let's address two related issues before developing this picture for more interesting language fragments:

- Row 2 in our picture mentions 'entropy'. We should define and at least briefly explore that notion
- In the observations (Rows 1 and 4), we never considered 'noise' in the data.

Our conclusion will be that this talk of entropy commits us to very little, but is almost certainly on the right track, and that something like this picture can be maintained even when the data is noisy.

7.2 Entropy and codes

(6) Suppose we have a countable sample space Ω with events $A \subseteq \Omega$ and a probability measure $P : 2^\Omega \to [0,1]$, where 2^Ω is the set of all subsets of Ω,[1] and as usual:

i. $0 \leq P(A) \leq 1$ for all $A \subseteq \Omega$

ii. $P(\Omega) = 1$

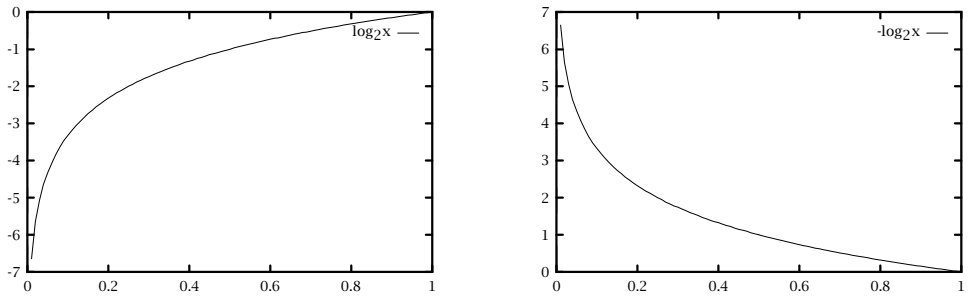
iii. $P(A \cup B) = P(A) + P(B)$ for any events A, B where $A \cap B = \emptyset$,

(7) Then the amount of information ("self-information," "surprisal") of an event A is

$$i(A) = \log \frac{1}{P(A)} = -\log P(A)$$

So if we have 10 possible events with equal probabilities of occurrence, so $P(A) = 0.1$, then

$$i(A) = \log \frac{1}{0.1} = -\log 0.1 \approx 3.32$$



surprisal as a function of $p(A)$

(8) The freely available programs **octave** and **gnuplot** make drawing these graphs a trivial matter. After starting **octave**, the graphs above are drawn with the commands:

```
>x=(0.005:0.005:0.99)';data = [x,log2(x)];gplot [0:1] data
>x=(0.005:0.005:0.99)';data = [x,-log2(x)];gplot [0:1] data
```

Of course it's easy to plot these curves on very expensive platforms like Mathematica or Matlab too.

[1] All our examples will involve countable spaces Ω, but it is important to note that when Ω is not countable, the collection of all its subsets is "too big" for our probability measure in a certain sense, and so then certain additional conditions need to be imposed on the set of events. My favorite gentle introduction to this very basic point is [7].

(9) Consider the case where probability is distributed uniformly across 8 possible events. Then exactly 3 bits of information are given by the occurrence of any particular event A:

$$i(A) = \log \frac{1}{0.125} = -\log 0.125 = 3$$

Of course, the information given by the occurrence of the "universal event" $\cup\Omega_X$, where $P(\cup\Omega_X) = 1$, is zero:

$$i(A) = \log \frac{1}{1} = -\log 1 = 0$$

And obviously, if events A, B are independent, that is, $P(AB) = P(A)P(B)$, then

$$\begin{aligned} i(AB) &= \log \tfrac{1}{P(AB)} \\ &= \log \tfrac{1}{P(A)P(B)} \\ &= \log \tfrac{1}{P(A)} + \log \tfrac{1}{P(B)} \\ &= i(A) + i(B) \end{aligned}$$

And in the case where there are two possible events A, B where $P(A) = 0.1$ and $P(B) = 0.9$,

$$i(A) = \log \frac{1}{0.1} = -\log 0.1 \approx 3.32$$

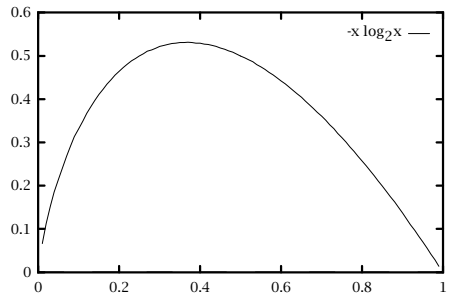
The information conveyed by the other event in this situation is:

$$i(B) = \log \frac{1}{0.9} \approx .15$$

(10) Often we are interested not in the average information of the possible events:
…from the point of view of engineering, a communication system must face the problem of handling any message that the source can produce. If it is not possible or practicable to design a system which can handle everything perfectly, then the system should handle well the jobs it is most likely to be asked to do, and should resign itself to be less efficient for the rare task. This sort of consideration leads at once to the necessity of characterizing the statistical nature of the whole ensemble of messages which a given kind of source can and will produce. And information*, as used in communication theory, does just this.* [17, p14]

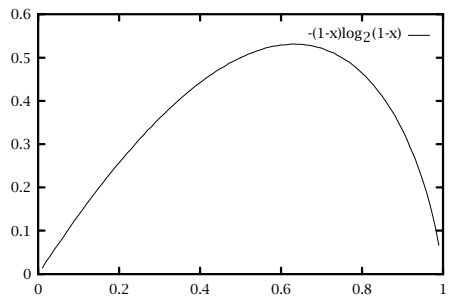
(11) The **entropy** or average information of an arbitrary event A is

$$H = \sum_{A \in \Omega_X} P(A) i(A) = -\sum_{A \in \Omega_X} P(A) \log P(A)$$



$-p(A) \log p(A)$ as a function of $p(A)$

```
>x=(0.01:0.01:0.99)';data = [x,(-x .* log2(x))];gplot [0:1] data
```

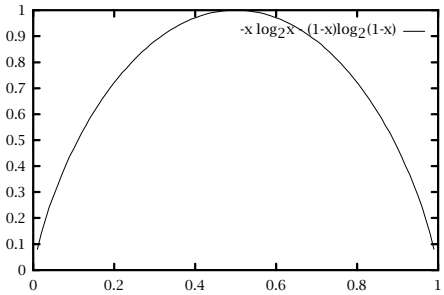


$-(1-p(A))(\log(1-p(A))$ as a function of $p(A)$

```
>x=(0.01:0.01:0.99)';data = [x,(-(1-x) .* log2(1-x))];gplot [0:1] data
```

(12) The **entropy** of a source X of independent events whose union has probability 1:

$$H(X) = \sum_{A \in \Omega_X} P(A)i(A) = -\sum_{A \in \Omega_X} P(A)\log P(A)$$



sum of previous two curves: $p(A)\log p(A) - (1-p(A))(\log(1-p(A))$

```
>x=(0.01:0.01:0.99)';data = [x,(-x .* log2(x))-(1-x).*log2(1-x)];gplot [0:1] data
```

(13) If we use some measure other than bits, a measure that allows r-ary decisions rather than just binary ones, then we can define $H_r(X)$ similarly except that we use $\log_r$ rather than $\log_2$.

[14, 11] (14) Shannon shows that this measure of information has the following intuitive properties (as discussed also in the Miller & Chomsky's famous discussion of this result):

a. Adding any number of impossible events to Ω_X does not change $H(X)$.

b. $H(X)$ is a maximum when all the events in Ω_X are equiprobable.
(see the last graph on page 70)

c. $H(X)$ is additive, in the sense that $H(X_i \cup X_j) = H(X_i) + H(X_j)$ when X_i and X_j are independent.

(15) Shannon considers the information in a discrete, noiseless message. Here, the space of possible events is given by an alphabet (or "vocabulary") Σ.

[14] The entropy of the source sets a lower bound on the size of the messages, using the definition of H_r in (13):

> **Shannon's thm:** Suppose that X is a first order source with outcomes (or outputs) Ω_X. Encoding the characters of Ω_X in a code with characters Γ where $|\Gamma| = r > 1$ requires an average of $H_r(X)$ characters of Γ per character of Ω_X.
>
> Furthermore, for any real number $\epsilon > 0$, there is a code that uses an average of $H_r(X) + \epsilon$ characters of Γ per character of Ω_X.

7.3 Noise

[1, 12, 16] (16) Some of the literature cited above shows how our learning problems could be regarded as an instance of the traditional problem in statistical modeling of identifying the "best" classifier. Notice that the 'classifiers' we considered – e.g. the Boolean functions – are more complex than usual (and probably still less complex than necessary!). But we have not considered how a learner could deal with 'noise', 'mistaken' data .

(17) **Some simpler, practice puzzles.**

a. Suppose I have a fair coin that either has heads and tails as usual, or else has 2 heads. You can sample the results of flips of the coin. How many flips will ensure that you can be 99% sure whether the coin has 2 heads? A single tails will definitively answer the question, so all we need to worry about is a sequence of heads that happens even when the tails is there. We calculate:

$$\begin{gathered}(\tfrac{1}{2})^n \leq \tfrac{1}{100}\\ (\tfrac{1}{2^n}) \leq \tfrac{1}{100}\\ 2^n \geq 100\\ \log_2 100 \leq n\\ 6.6439 \leq n\\ \text{so, 7 flips is always enough}\end{gathered}$$

b. Suppose that I want to tell you one thing: 0 or 1. But my message is noisy in the sense that the message is altered $0 < \eta < \frac{1}{2}$ of the time. How many messages do you need before you can be 99% sure you've got it right? This is not the same puzzle, since now, a single 0 or 1 never definitely indicates what the message is.

(18) **Define**

$$\textbf{expected value of } X: \quad E(X) = \mu = \sum_{w \in \Omega} X(w)P(w)$$

$$\textbf{variance of } X: \quad V(X) = E((X - E(X))^2)$$

$$\textbf{standard deviation of } X: \quad D(X) = \sigma = \sqrt{V(X)}$$

(19) **Chebyshev's inequality**: randomly drawn values probably do not differ too much from μ:

$$P(|x - \mu| > n\sigma) \leq \frac{1}{n^2}.$$

In a normal distribution, 68% of the values will be within 1 standard deviation of the mean. 95% within 2 standard deviations. 99.7% within 3.

(20) **Chernoff bounds:** [3, 13, 6]

(21) Angluin, Laird and Kearns show how to adjust the monomial learner (from lecture 4) so that it can handle simple classification error. [10, §5]

(22) Let $EX_\eta(c, P)$ be a source of elements of drawn randomly and independently according to P, and labeled according to whether the example is in c, with classification error η.

(23) Various "malicious adversary" models have been studied. only very small amounts of noise can be handled when the misinformation can be designed to be maximally damaging. [15, 9]

Angluin & Laird show how certain classes of CNF formulas can be learned with simple classification noise. In this setting, after each sample is drawn, there is a probability $\eta < \frac{1}{2}$ that the label of the sample will be incorrect. Kearns shows how a similar trick can be used to adjust many (but not all) successful PAC-learners to learning with classification noise. [2] [8, 10]

(24) We want the learner to estimate the noise level in such a way that a clause is included in the hypothesis when we accumulate enough evidence for it, given the following:

n	(the number of propositional variables)
ϵ	(the admissible margin of error)
η_b	(an upper bound on the amount of noise $< \frac{1}{2}$)

XXXXXXX

Theorem: Learner ϕ PAC-identifies monomials in the presence of η classification noise given a number of samples that is polynomial in $n^k, \frac{1}{\epsilon}, \log \frac{1}{\delta}$ and $\frac{1}{(1-2\eta_b)}$.

Exercise 4 These exercises are slightly challenging, but success on any of them would constitute a new, probably publishable achievement. A start on one of these could be a term project.

(Optional-0) Establish whether the computation of Feldman's series expansions from the (DNF) satisfying vectors is tractable.

(Optional-1) Design a Feldman learner that can handle noise.

(Optional-2) Apply an algorithm like Harris's or Goldsmith's to phonetic sequences. (maybe also compare de Marcken's work [4, 5])

(Optional-3) Modify the Feldman representation of Boolean functions so that the simplest components of the expansion can over- and under-generalize instead of just over-generalizing. Compare the fit with the data for a complexity measure based on this with the fit that Feldman achieves.

References for Lecture 7

[1] Alon, N., Ben-David, S., Cesa-Bianchi, N., and Haussler, D. Scale-sensitive dimensions, uniform convergence, and learnability. *Journal of the Association for Computing Machinery 44*, 4 (1997), 615–631.

[2] Angluin, D., and Laird, P. D. Learning from noisy examples. *Machine Learning 2* (1988), 343–370.

[3] Chernoff, H. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics 23*, 4 (1952), 493–507.

[4] de Marcken, C. Linguistic structure as composition and perturbation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics* (1996).

[5] de Marcken, C. *Unsupervised language acquisition*. PhD thesis, Massachusetts Institute of Technology, 1996.

[6] Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the Americal Statistical Association 58*, 1 (1963), 13–30.

[7] Jacod, J., and Protter, P. *Probability Essentials*. Springer, NY, 2000.

[8] Kearns, M. Efficient noise-tolerant learning from statistical queries. *Journal of the Association for Computing Machinery 45*, 6 (1998), 392–401. See also Proceedings of the 25th ACM Symposium on the Theory of Computing, pp. 392-401, 1993, ACM Press.

[9] Kearns, M., and Li, M. Learning in the presence of malicious errors. *SIAM Journal on Computing 22*, 4 (1993), 807–837. See also Proceedings of the 20th ACM Symposium on the Theory of Computing, pp. 267-280, 1988, ACM Press.

[10] Kearns, M. J., and Vazirani, U. V. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, Massachusetts, 1994.

[11] Miller, G. A., and Chomsky, N. Finitary models of language users. In *Handbook of Mathematical Psychology, Volume II*, R. D. Luce, R. R. Bush, and E. Galanter, Eds. Wiley, NY, 1963, pp. 419–492.

[12] Mukherjee, S., Niyogi, P., Poggio, T., and Rifkin, R. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of Empirical Risk Minimization. *Advances in Computational Mathematics* (2004). forthcoming.

[13] Okamoto, M. Some inequalities relation to the partial sum of binomial probabilities. *Annals of the Institute of Statistical Mathematics 10*, 1 (1958), 29–35.

[14] Shannon, C. E. The mathematical theory of communication. *Bell System Technical Journal 127* (1948), 379–423. Reprinted in Claude E. Shannon and Warren Weaver, editors, *The Mathematical Theory of Communication*, Chicago: University of Illinois Press.

[15] Valiant, L. Learning disjunctions of conjunctions. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence* (Los Angeles, CA, 1985), Morgan Kaufmann, pp. 560–566.

[16] Vapnik, V. N. *Statistical Learning Theory*. Wiley, NY, 1998.

[17] Weaver, W. Recent contributions to the mathematical theory of communication. In *The Mathematical Theory of Communication*, C. E. Shannon and W. Weaver, Eds. University of Illinois Press, Chicago, 1949.

8 Subject-Predicate languages with *every*

(1) Languages in which there are only utterances like “snake!” or “leopard” or “eagle” are syntactically, semantically and inferentially trivial, but the problem of learning such a language is still significant. We do not really have a good model of how kids (or monkeys) learn the meanings of single lexical items.

But there are generalizations that hold across languages, generalizations that

we should be able to explain. For example, (Snedeker & Gleitman 2004) say:

> *Even though children hear both verbs and nouns from earliest infancy, their earliest vocabulary is overwhelmingly nominal [4, 1], with only very few true verbs.*

(2) Kids make various sorts of errors in learning word meanings:

> *Children may under-extend a word by using a category label, for instance, for only a subset of the members of the (adult) category.…Children may also over-extend a word by applying it to members of the adult category and to members of other categories that are perceptually similar. For example, they may use ball for balls of all kinds, and also for round hanging lampshades, doorknobs and the moon. Or they may over-extend a term like door to corks, jar-lids, box-lids and gates when wishing to have the relevant object opened or closed…*(Clark 1993)

(3) We saw that some of these errors could be avoided by conservative learning strategies, but we apparently see such errors regularly in humans. How do these errors get straightened out? Some errors could be straightened out by explicit correction:

This is not a zorg!

Or by classification facts like

Every zorg is an animal

Obviously, utterances like this are not useful to the language learner unless they can be understood. So let’s extend our language first by adding the latter sort of instruction.

(4) Looking ahead to a puzzle we should consider: there is some reason to think that maybe kids have trouble understanding sentences with *all* or *every*, but the situation is not clear yet. Guerts (2003) claims that the evidence supposedly showing this should be attributed to a different kind of “error”:

> *Inhelder and Piaget (1959) presented children with displays of colored squares and circles…*
>
> *Scene:* *14 blue circles, 2 blue squares, 3 red squares*
> *Q:* *Are all the circles blue?*
> *A:* *No, there are two blue squares*
>
> *Children who give non-adult responses to quantified sentences construe the strong determiner as if it were weak…it is a parsing problem.*

8.1 Syntax

[7, 6]

(5) Inspired by recent work of Pratt-Hartmann and Moss, we now turn to languages with grammars of the following form (for some $i > 0$):

syntax G:

$$\begin{aligned} \text{N} ::= &\quad n_1 \mid n_2 \mid \ldots \mid n_i \\ \text{S} ::= &\quad \textbf{every N is a N} \end{aligned}$$

Note that the grammar uses the special terminal symbols **every, is,** and **a**.

(6) Since we now have two categories of expressions – nouns N and sentences S – it's convenient to think of the category as part of the expression.
So let's think of the first rule of G as defining the lexicon

$$Lex = \{(n_1, \text{N}), \ldots, (n_j, \text{N})\}$$

(7) The second rule of G provides a way of building complex expressions: for any $(n, \text{N}), (n', \text{N}) \in Lex$,

$$f((n, \text{N}), (n', \text{N})) \quad = (\textbf{every } n \textbf{ is a } n', \text{S})$$

The set of structure building rules $\mathcal{F} = \{f\}$ has just one element here.

(8) **The language L(G)** = *closure*$(Lex, \mathcal{F})$**,** the *closure of Lex with respect to the functions* $\mathcal{F}$.
This is just a short way of saying that the language is the lexical items plus everything you can build from them.

(9) The standard approach in formal language theory considers only strings of category S as part of the language, but we are linguists, and so we care about everything in the language. In particular, as will be clear below, the semantic interpretation extends to non-S elements (nouns), and the inference methods also highlight the role of the nouns.
When we want to pick out just the sentences or the expressions of any category C, we use

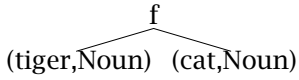
$expression(C, G) = \{(x, C) \mid (x, C) \in L(G)\}$	the expressions of category C
$string(C, G)) = \{x \mid (x, C) \in L(G)\}$	the strings of category C

(10) Obviously, L(G) is finite, since we're assuming that i is.
Notice the grammar as given above will have fewer symbols than the list of sentences. This turns out to be important: *powerful grammars can simplify descriptions of even simple sets.*

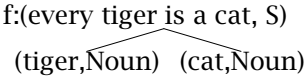
(11) **The derivations** Γ**(G)** are defined this way:

a. Every lexical item is a (trivial) derivation. Think of it as a 1-node tree.

b. If a pair of expressions (x, y) is in the domain of $f \in \mathcal{F}$, then the function expression $f(x, y)$ is a derivation, which we could depict with a 3-node tree like this:



Notice that the order of the branches matters. The **(syntactic) value** of a derivation tree like this is the sentence *every tiger is a cat.* Sometimes we include the value in the derivation tree for convenience, like this:



(12) **How are derivation trees are superior to the standard ones:**

a. In our trees, the rule being applied is made explicit. In standard trees, we have to figure this out from the label of the parent and daughters. For this simple fragment, figuring that out is a trivial matter, but in later fragments things will get more complicated.

b. In our trees, every arc means the same thing: "is a constituent of."
In standard trees, the 'pronounced strings', the 'terminals' are attached by arcs, but are not naturally regarded as constituents of the categories they attach to.

c. Our trees extend to functions that do something other than simply concatenating the strings of subconstitutents. For our simple fragment, we do not need this power, but later we will.

8.2 Semantics

(13) Over any universe E, each interpretation μ assigns meanings to elements of $\Gamma(G)$ as follows. For any lexical items $(n,\mathrm{N}),(n',\mathrm{N})$,

$$\mu(n,\mathrm{N}) \subseteq E$$
$$\mu(f((n,\mathrm{N}),(n',\mathrm{N}))) = \begin{cases} 1 & \text{if } \mu(n,\mathrm{N}) \subseteq \mu(n',\mathrm{N}) \\ 0 & \text{otherwise,} \end{cases}$$

where f is the structure building function defined by the 2nd grammar rule.

(14) For any sentence e, we say μ *verifies* or *satisfies* e iff $\mu(e) = 1$
For any set A of sentences, $\mu(A) = 1$ iff for every sentence $e \in S$, $\mu(e) = 1$
For any set A of sentences, we say μ *verifies* or *satisfies* A iff $\mu(A) = 1$

(15) For any set of sentences $A \subseteq S(G)$ and any sentence $e \in L(G)$,

$$A \models e \text{ iff every interpretation that verifies } A \text{ also verifies } e.$$

In this case, we say A *entails* or *semantically implies* e.

(16) **Thm.** For every interpretation μ and every noun A, μ(every A is an A,S) = 1

(17) **Puzzle.** Why would a language like this one, or like English or any other natural language, provide so many ways to say something with zero information content?

AL: We could avoid having all these trivial truths if we changed the semantics of the sentences so that they assert proper subset relations:

$$\mu(f((n,\mathrm{N}),(n',\mathrm{N}))) = \begin{cases} 1 & \text{if } \mu(n,\mathrm{N}) \subset \mu(n',\mathrm{N}) \\ 0 & \text{otherwise,} \end{cases}$$

BK: But then the sentences of the form "Every A is an A" would be necessarily false, and so still useless for conveying information.

8.3 Inference

(18) Traditionally, the inference system is given this way. For every noun A,B,C:

$$\frac{}{\text{every A is an A}} \qquad \frac{\text{every A is a B} \quad \text{every B is a C}}{\text{every A is a C}}$$

The rules of the form on the left have no premises, and so they are sometimes called the *Axioms*.

Chaining together applications of the rules on the right, we get tree structures that logicians draw with the root on the bottom. For example,

$$\dfrac{\text{every leopard is a cat} \quad \dfrac{\text{every cat is a danger} \quad \text{every danger is a danger}}{\text{every cat is a danger}}}{\text{every leopard is a danger}}$$

BK: Really, since expressions can be ambiguous, the elements involved in inferences should be derivations, not strings or expressions.

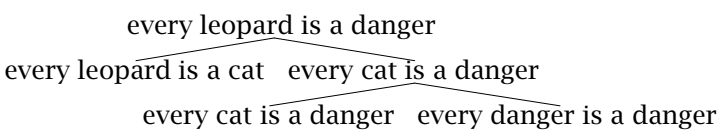
ES: Right. We will do that with later fragments which are ambiguous, but for the moment, we can follow traditional presentations. However, I prefer the linguists' presentation of trees, with the root up…pushing that analogy further we can say…

(19) A proof for the language of our fragment is a derivation tree from a grammar that has the rule:

$$\text{(every A is a B)} \rightarrow \text{(every A is a C) (every C is a B)}$$

for any nouns A,B,C.

Then the tree drawn logician's-style above can be recognized as a context-free derivation tree where the category labels are sentences of our language L(G):



(20) For any set of sentences $A \subseteq S(G)$ and any sentence $e \in L(G)$,

$$A \vdash e \text{ iff there is a derivation tree with root } A \text{ and leaves from } \Gamma \cup \mathit{Axioms}.$$

We call such a derivation a proof of e from Γ.

(21) **Thm.** Every sentence e has infinitely many proofs.

AL: Does allowing zero-information sentences make the reasoning easier?

References for Lecture 8

[1] BATES, E., DALE, P., AND THAW, D. Individual differences and their implications for theories of language development. In *The Handbook of Child Language*, P. Fletcher and B. MacWhinney, Eds. Blackwell, Oxford, 1995, pp. 96–151.

[2] CLARK, E. V. *The Lexicon in Acquisition*. Cambridge University Press, Cambridge, 1993.

[3] GEURTS, B. Quantifying kids. *Language Acquisition 11* (2003), 197–218.

[4] GOLDIN-MEADOW, S., SELIGMAN, M. E. P., AND GELMAN, R. Language in the two-year old. *Cognition 4* (1976), 189–202.

[5] INHELDER, B., AND PIAGET, J. *La Genèse des Structures Logiques Elémentaires: Classifications et Sériations*. Delachaux et Niestlé, Neuchâtel, 1959. English translation, *The Early Growth of Logic in the Child: Classification and Seriation*, London: Routledge and Kegan Paul, 1964.

[6] MOSS, L. Natural language, natural logic, natural deduction. *Forthcoming* (2004). Indiana University.

[7] PRATT-HARTMANN, I. Fragments of language. *Journal of Logic, Language and Information 13*, 2 (2004), 207–223.

[8] SNEDEKER, J., AND GLEITMAN, L. R. Why it is hard to label our concepts. In *Weaving a Lexicon*, D. Hall and S. Waxman, Eds. MIT Press, Cambridge, Massachusetts, 2004, pp. 257–293.

9 Subject-Predicate languages with *every*, part 2

> *…the basic criterion in terms of which our proposals, and those of any form of model theoretic semantics for natural language, are to be evaluated is given by:*
>
> *(1)* Criterion of Logical Adequacy. *Expressions* e *and* e' *of a natural language are judged by speakers to be semantically related in a certain way if and only if their formal representations are provably related in that way.*
>
> *The fundamental relation that we, and others, endeavor to represent is the* entailment *relation…*
>
> *(2)* *a. John is a linguist and Mary is a biologist*
>
> *b. John is a linguist*
>
> *…our system would be descriptively inadequate if we could not show that our representation for (2a) formally entailed our representation for (2b).* (Keenan & Faltz 1985, pp1-2)

9.1 Summary

(3) Inspired by a debate between Pratt-Hartmann and Moss, we are considering the following fragment, (for some $i > 0$) [8, 7]

syntax G:

$$\begin{aligned} Lex &= \{(n_1, \mathrm{N}), \ldots, (n_i, \mathrm{N})\} \\ \mathcal{F} &= \{f\} \text{ where for any } 1 \le j, k \le i \\ f((n_j, \mathrm{N}), (n_k, \mathrm{N})) &= (\textbf{every } n_j \textbf{ is a } n_k, \mathrm{S}) \end{aligned}$$

- This defines a finite language $L(G)$ and a finite set of syntactic derivations $\Gamma(G)$.

semantics: $\mu : \Gamma(G) \to (E, 2)$ defined as follows

$$\mu(n, \mathrm{N}) \subseteq E$$

$$\mu(f((n_j, \mathrm{N}), (n_k, \mathrm{N}))) = \begin{cases} 1 & \text{if } \mu(n_j, \mathrm{N}) \subseteq \mu(n_k, \mathrm{N}) \\ 0 & \text{otherwise.} \end{cases}$$

- here E is the "universe" of things we might talk about, and 2 is the set of truth values {0,1}.
- each noun is interpreted as a **"property of things,"** represented here by the set of those things
- *every* is interpreted as $\subseteq$

inference: (*Axioms*) every n_j is an n_j (*Rules*) every n_j is a n_l, every n_l is a n_k ⟹ every n_j is a n_k

- This defines an infinite set of logical derivations: trees built from the rules

(4) For any set of sentences $A \subseteq S(G)$ and any sentence $e \in L(G)$,

$A \models e$ iff every interpretation that verifies A also verifies e.

$A \vdash e$ iff there is a (logical) derivation tree with root A and leaves from $A \cup Axioms$.

(5) Some 'meta' properties noted last time – the main subject today. For every set A and expression e,

(Soundness)	If $A \vdash e$ then $A \models e$
(Completeness)	If $A \models e$ then $A \vdash e$
(Canonical model property)	There is a model μ that verifies exactly the sentences that A derives.

9.2 Reflections: why this fragment is so interesting!

(6) **What did we add?** A quantifier, *every*. What's a quantifier?

A quantifier is a property of, or relation among, properties of things or relations among things.

[3, 2, 6, 1, 4, 12]

So as we learned from Frege (and Church, Montague,...), quantifiers are "second order" concepts. They are concepts about concepts.

While it is natural to think that "This is a snake" attributes a property (being a snake) of "this" thing, Frege noted that it may seem odd at first to regard a quantified sentence as telling us not about any things, but about concepts, But $\subseteq$ is a relation among properties of things (which we represent as sets), and similar, elegant accounts can be provided for an infinite range of quantifiers we find in natural and artificial languages:

some, every, most, more than 2/3, at least six, at most five, exactly 12, finitely many,...

We'll treat some of these later, and it will become clear if it's not already: these are naturally regarded as concepts about concepts.

> *It is true that at first sight the proposition*
>
> *"All whales are mammals"*
>
> *seems to be not about concepts but about animals; but if we ask which animal then we are speaking of, we are unable to point to any one in particular...If it be replied that what we are speaking of is not, indeed, an individual definite object, but nevertheless an indefinite object, I suspect that "indefinite object" is only another term for concept...* (Frege 1884, §47)

Puzzle: if quantifiers are second-order concepts, why do we have "all" $\forall$ and "some" $\exists$ in first-order logic? A proper answer to this question is beyond the scope of this class (and fortunately not needed to address our main questions), but it is right to be puzzled about this. A clue: it is no accident that standard treatments of first order logic do not assign denotations to $\forall$ or $\exists$, whereas denotations for these are provided in completely clear, explicit form by Church [2], for example. The special status of "first order" quantifiers is discussed in [11, 13].

(7) **A new puzzle for the learner.** In the previous fragment we considered some first ideas about learning words like *leopard* or *snake* from examples. But this fragment contains a word (or, more precisely, a construction **"every ____ is a ____ "**) with a meaning that is very different. Even when we are told *every snake is a danger*, we are not typically in a situation where all snakes or all dangers and the inclusion relation would be perceivable or even imaginable! Subset relations are abstract in a way that snakes and leopards aren't, as Frege noticed, so how could we learn what *every* means?

(8) **Inference patterns and meaning: the "Gentzen-Hertz" idea.** One idea is that we learn about the meanings of these words (at least in part) by seeing how to reason with them. This is why the "soundness" and "completeness" of our fragment is so interesting. In a very simple sense that we can analyze, the simple inference rules we gave provide the whole story about reasoning with *every* in this fragment. If we knew these patterns, would there be anything else to learn about the meaning of this word?

Hertz and his student Gentzen proposed that the meaning of connectives like *and* might be given, in some sense, by the inference methods for the connective. We should consider a similar idea about *every*. (We'll see: it won't quite work!) [9]

BK: Isn't this fragment peculiar because it has no names? This question is worth thinking about. In English, proper names are syntactically different from common nouns: names do not allow a determiner, unless they are "used as" nouns, in some intuitive sense:

He's the Albert Einstein of the group, always thinking about what's really happening

But other languages differ in this respect, so what is a name, really?

The logician Willard van Orman Quine weighed in on this topic, noting that we can perfectly well define a predicate that applies to just one particular individual. For example, we could have a predicate *socratizes* that applies to just one thing, namely Socrates. And so we could reason about the socratizer like this: [10]

every socratizer is a mortal

every socratizer is a human — every human is a mortal

We usually think of names as denoting a single thing, but Quine notices that this is not even close to right: we have no trouble at all in using names to talk about the winged horse Pegasus or the great detective Sherlock Holmes, and in saying that really there is no Pegasus or Sherlock Holmes.

The full story about the meaning of names in human languages promises to be complicated, but for now, we notice that nothing blocks having a noun like *socratizer* that only applies to one thing, or a noun like *pegasizer* that does not apply to anything. In our fragment, there is no problem providing a natural interpretation for *every pegasizer is a nonexistent*.

We now go over soundness and completeness carefully: to see exactly how the reasoning characterizes the sound uses of *every*.

9.3 Metatheory, carefully

First, remember the definitions from earlier:

(9) For any set of sentences $A \subseteq S(G)$ and any sentence $e \in L(G)$,

$A \models e$ iff every interpretation that verifies A also verifies e.

$A \vdash e$ iff there is a deduction with root e and leaves from $A \cup Axioms$.

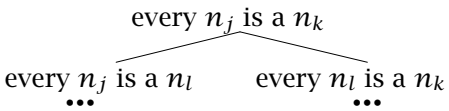
(10) (Soundness) If $A \vdash e$ then $A \models e$

Proof: We use an induction on the number of nodes n in the logical derivation tree.

($n = 1$) Then by the definition of $\vdash$, the sentence is either in A or an axiom, and hence in either case verified by any interpretation that verifies A.

(IH) Suppose that the result holds for all derivations with i or fewer nodes.

($n = i + 1$) Consider any logical derivation tree with $i + 1$ nodes. The tree will look like this:



where the dots ••• represent subtrees rooted by the daughters of the root e = (every n_j is a n_k,S). Since these subtrees have fewer than i nodes, we know the daughters are each entailed by A.

That means that any interpretation verifying A is also one in which

$$\mu(n_j, \mathrm{N}) \subseteq \mu(n_l, \mathrm{N}) \text{ and } \mu(n_l, \mathrm{N}) \subseteq \mu(n_k, \mathrm{N}).$$

So it follows that

$$\mu(n_j, \mathrm{N}) \subseteq \mu(n_k, \mathrm{N}),$$

and so $A \models$ (every n_j is a n_k,S). □

Completeness is not quite so easy. Here we present a short proof from [7]:

(11) (Completeness) If $A \models e$ then $A \vdash e$

Proof: i. Consider any A and any e = (every n_1 is a n_2,S) where $A \models e$.

ii. Define an interpretation μ for the language over universe E = {you} as follows:
For each $(n_j, \mathrm{N}) \in Lex$,

$$\mu(n_j, N) = \begin{cases} E & \text{if } A \vdash \text{(every } n_1 \text{ is a } n_j\text{,S)} \\ \varnothing & \text{otherwise.} \end{cases}$$

iii. Notice first, we are allowed to define the interpretations however we like, conforming to our earlier characterization of the fragment. There is nothing wrong with this definition.

iv. Second, if we can show that $\mu(n_2, \mathrm{N}) \neq \varnothing$, then the proof is complete.

v. If (every n_j is a n_k,S) $\in A$ then $\mu(n_j, \mathrm{N}) \subseteq \mu(n_k, \mathrm{N})$.

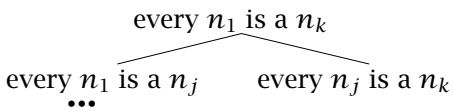
Sub-Proof of v: Assume (every n_j is a n_k,S) $\in A$ and consider these two cases:

($\mu(n_j, \mathrm{N}) = \varnothing$) In this case, v holds trivially.

($\mu(n_j, \mathrm{N}) = E$)

a. In this case, by the definition of μ, $A \vdash$ (every n_1 is a n_j,S).

b. So there is some derivation of the form:



where the dots ••• is a subtree whose leaves are Axioms and elements of A.

c. This tree shows that $A \vdash$ (every n_1 is a n_k,S).

d. So then by the definition of μ, $\mu(n_k, \mathrm{N}) = E$.

e. But then it follows that $\mu(n_j, \mathrm{N}) \subseteq \mu(n_k, \mathrm{N})$, and the *Sub-Proof is complete.*

vi. By v, μ verifies A. Since $A \models e$, it follows that μ verifies e. That is, $\mu(n_1, \mathrm{N}) \subseteq \mu(n_2, \mathrm{N})$.

vii. Since every set derives (every n_1 is a n_1,S), A does, and so by the definition of μ again, $\mu(n_1, \mathrm{N}) = E$, and so by vi, $\mu(n_2, \mathrm{N}) = E$ too. □

We'll postpone the discussion of the "canonical model property" until after looking at some computational issues – the computational issues will make the relevance of the canonical models especially clear.

Remember the recipe for mastering these non-trivial proofs! It takes 2 days.

References for Lecture 9

[1] BARWISE, J., AND COOPER, R. Generalized quantifiers and natural language. *Linguistics and Philosophy 4* (1981), 159–219.

[2] CHURCH, A. A formulation of the simple theory of types. *Journal of Symbolic Logic 5* (1940), 56–68.

[3] FREGE, G. *Die Grundlagen der Arithmetik*. Koebner, Breslau, 1884. J.L. Austin's translation available as *The Foundations of Arithmetic*, Evanston, Illinois: Northwestern University Press, 1980.

[4] KEENAN, E. L. The semantics of determiners. In *The Handbook of Contemporary Semantic Theory*, S. Lappin, Ed. Blackwell, Oxford, 1996.

[5] KEENAN, E. L., AND FALTZ, L. M. *Boolean Semantics for Natural Language*. Reidel, Dordrecht, 1985.

[6] MONTAGUE, R. The proper treatment of quantification in ordinary English. In *Approaches to Natural Language*, J. Hintikka, J. Moravcsik, and P. Suppes, Eds. Reidel, Dordrecht, 1973. Reprinted in R.H. Thomason, editor, *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press, §8.

[7] MOSS, L. Natural language, natural logic, natural deduction. *Forthcoming* (2004). Indiana University.

[8] PRATT-HARTMANN, I. Fragments of language. *Journal of Logic, Language and Information 13*, 2 (2004), 207–223.

[9] PRAWITZ, D. Ideas and results in proof theory. In *Proceedings of the Second Scandinavian Logic Symposium*, J. Fenstad, Ed. North-Holland, Amsterdam, 1971, pp. 235–307. Partially reprinted as "Gentzen's analysis of first order proofs," in R.I.G. Hughes, *A Philosophical Companion to First Order Logic*, Hackett: Indianapolis, 1993.

[10] QUINE, W. On what there is. *Review of Metaphysics 2* (1948), 21–38. Reprinted in *From a Logical Point of View*, NY: Harper, 1953.

[11] VAN BENTHEM, J. Questions about quantifiers. *Journal of Symbolic Logic 49* (1984), 443–466.

[12] VAN BENTHEM, J. *Essays in Logical Semantics*. Reidel, Dordrecht, 1986.

[13] WESTERSTÅHL, D. Quantifiers in formal and natural languages. In *Handbook of Philosophical Logic, Volume IV*, D. Gabbay and F. Guenthner, Eds. Reidel, Dordrecht, 1989, pp. 463–504.

10 Languages with *every* and relative clauses

> **28.** *The definition of the number two, "That is called 'two'" – pointing at two nuts – is perfectly exact. But how can two be defined like that?…*
>
> **30.** *So one might say: the ostensive definition explains the use – the meaning – of the word when the overall role of the word in the language is clear. Thus if I know that someone means to explain a color-word to me the ostensive definition "That is called 'sepia'" will help me to understand the word.* (Wittgenstein 1958)

> *…language design is constrained by how it is learned…Overall, children's comprehension ability looks the way it does because they are building a linguistic ladder, so to speak, as they are climbing it. (Trueswell and Gleitman 2004)*

10.1 Recap

(1) We looked at a logic with sentences of the form *every N is a N*, and saw that it was easy to provide a sound, complete inference system.

In a certain sense, this logic tells the whole story about the role of *every* in the language.

There may be other reasons to recognize a sentence as true besides reasoning, based on some understanding of how you are interpreting the nouns, but all the sound reasoning is captured by our simple Axioms and Rules.

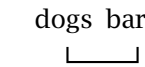
(2) We left many questions about the last fragment unexplored. We did not consider the computational issues (how to recognize sentences, how to recognize logical derivations, how to learn the language,…). We did not prove the canonical model property. And finally, we never carefully considered the the "Gentzen-Hertz" idea that meaning might be partly determined by use in reasoning, and the puzzle about why kids seem to misunderstand *all* and *every*. We'll make sure to hit these topics.

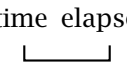
(3) Anyone who has looked at the Pratt-Hartmann and Third papers will have noticed that relative clauses make things harder, so I want to take a first look at them this week. To set the stage, it is useful to reformulate last week's fragment in a slightly less antique way, making the structure of that language more natural. [8, 9]

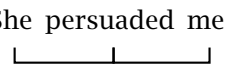
10.2 Another perspective on CFGs: Merge

(4) **Dependence and locality.** Words are often syntactically and semantically dependent on adjacent elements.

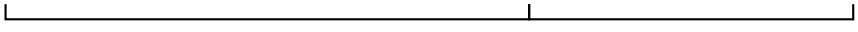
For example, the first and second word below are semantically related ("predicate-argument") and they are syntactically dependent in the sense that a change in the first word below may require a change in the second.







(5) Letting a given word act as a "head" of a phrase that may include modifiers, etc., other dependent heads are often in adjacent phrases.



In TB2, the verb *bark* only occurs once, in the idiom *barking up the wrong tree*.
The verb *elapse* only occurs once (shown below).
The verb *persuade* occurs 43 times (first 12 shown below).

```
19/wsj_1946:But the Beebes didn't come to that conclusion until {time limits} had {elapsed} for adding the adhesives maker as a defendant in the case, Ms. Adams said.

00/wsj_0060:At the same time, {Viacom} is trying to {persuade} {stations} to make commitments to "A Different World," a spin-off of "Cosby" whose reruns will become ava
01/wsj_0126:Several years ago {he} gave up trying to {persuade} {Miami} to improve its city-owned Orange Bowl, and instead built his own $100 million coliseum with priv
02/wsj_0242:For one thing, Pentagon officials, who asked not to be identified, worry that {the U.S.} will have a much tougher time {persuading} {Europeans} to keep some
03/wsj_0309:Last week, housing {lobbies} {persuaded} {Congress} to raise the ceiling to $124,875, making FHA loans more accessible to the well-to-do.
04/wsj_0405:Another hurdle concerns {Japan}'s attempts to {persuade} the {Soviet Union} to relinquish its post-World War II control of four islands north of Japan.
04/wsj_0439:And has {he} truly {persuaded} the {Communist Party} to accept economic change of a kind that will, sooner or later, lead to its demise?
04/wsj_0450:{They} {persuaded} {Mr. Trotter} to take it back and, with the help of the FBI, taped their conversation with him.
05/wsj_0589:And cultivating a luxury {image} strong enough to {persuade} {consumers} to pay more than $15 for lipstick or eye makeup requires a subtle touch that packag
06/wsj_0629:India's overregulated {businessmen} had to be {persuaded}, but they have started to think big.
\index{Japanese}08/wsj_0800:In 30 years of collecting impressionist and Japanese paintings, {he} has acquired 600 items, he says, enough to {persuade} {him} to start a
08/wsj_0810:Mr. de Cholet said {Mr. Fournier}'s biggest hope was to somehow {persuade} regulatory {authorities} to block the bid.
08/wsj_0835:Mr. Barnhardt said {Federal} was apparently successful in its effort to {persuade} former Tiger {pilots} to give the company a chance without a union.
```

[6]

(6) **Precedence.** Across languages, there are some preferred linear orders among words. For example, in a famous study of different language types, Greenberg notices that subjects usually precede objects, and that question words come first not only in English but also in: Berber, Finnish, Fulani, Greek, Guarani, Hebrew, Italian, Malay, Maori, Masai, Mayan, Norwegian, Serbian, Welsh, Yoruba, Zapotec.

Partly to account for facts like these, some grammatical frameworks assume that each head may first select a "complement" on the right, and then a "specifier" on the left, where these are identified by category (and possibly also by semantic value).

We will represent these kinds of requirements with lexical items like this:

x :: X	word **x** has category X
y :: Y	word **y** has category Y
z :: =X =Y Z	the word **z** selects a complement X, then a specifier Y, to form Z

[2, 13, 14]

(7) Inspired by basic ideas in Chomskian minimalism, let's define a kind of **merge grammar** which always uses just one rule *merge*, allowing only the lexicon to vary.

Each **lexical item** is a sequence of pronounced parts (possibly empty), followed by :: (indicating its lexical status), followed by a sequence of features.

The **features** are either category symbols (N,V,A,P,…) or features that "select" constituents of those same categories (=N,=V,=A,=P,…).

Derived expressions are strings followed by : and then by a feature sequence.

The generating function *merge* combines an expression that selects another with another expression of the right type, and concatenates it on the right if the selecting expression is lexical, and on the left otherwise.

$$merge(s \star =f\gamma,\ t \cdot f) = \begin{cases} st:\gamma & \text{if } \star \text{ is } :: \text{ (i.e. first arg is a lexical item)} \\ ts:\gamma & \text{if } \star \text{ is } : \text{ (i.e. first arg is a derived expression)} \end{cases}$$

for any strings s, t, where $\cdot$ is either : or ::, and γ is any (possibly empty) feature sequence

It is sometimes more readable to define a function like this in the logicians' style, with the arguments over the result, showing the two cases as separate rules:

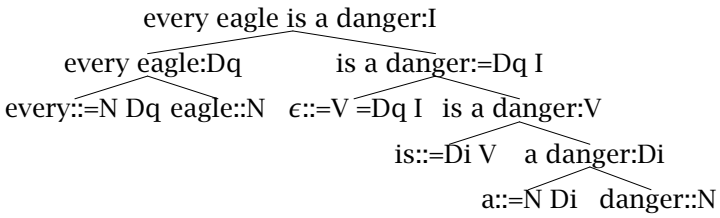
$$\frac{s :: =f\gamma \qquad t \cdot f}{st:\gamma}\ \text{merge1: selector is a lexical item}$$

$$\frac{s : =f\gamma \qquad t \cdot f}{ts:\gamma}\ \text{merge2: selector is a derived item}$$

Then the function *merge* is the union of merge1 and merge2.

(8) **A merge grammar for the fragment:** We use category Dq for "quantified determiners" and category Di for "indefinite determiners." We use I for "inflection," but leave it empty for the moment because we are ignoring tense and person distinctions. Then the whole grammar is given by the following lexicon:

snake::N	every::=N Dq
leopard::N	a::=N Di
eagle::N	is::=Di V
danger::N	ϵ::=V =Dq I



10.3 The recap with improvements

(9) **syntax G:** (we allow any $i > 0$ different nouns like these 4)

Lex={ snake::N, every::=N Dq,
leopard::N, a::=N Di,
eagle::N, is::=Di V,
danger::N, ϵ::=V =Dq I }.

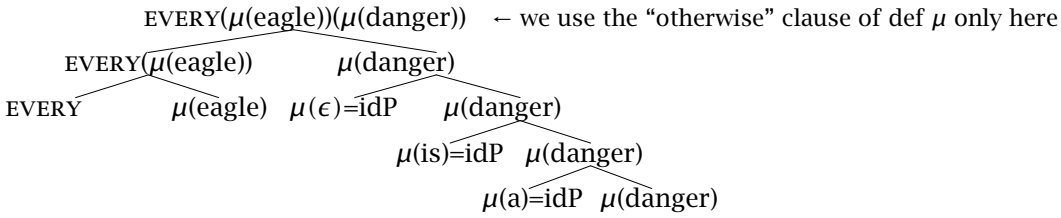
$\mathcal{F} = \{merge\}$, as defined in (7) above. $L(G) = closure(Lex, \mathcal{F})$, as before.

semantics: $\mu : \Gamma(G) \rightarrow (E, 2)$ defined as follows

$\mu(n :: \text{N}) \subseteq E$
$\mu(every :: =\text{N Dq})$ is the function EVERY that maps a property P to EVERY(P), where
EVERY(P) is the function that maps property Q to 1 iff $P \subseteq Q$
$\mu(a :: =\text{N Dq}) = \mu(is :: =\text{Di V}) = \mu(\epsilon :: =\text{V =Dq I})$
= the identity function on properties,*idP*

$$\mu(merge(A, B)) = \begin{cases} \mu(A)(\mu(B)) & \text{if defined} \\ \mu(B)(\mu(A)) & \text{otherwise} \end{cases}$$

- As before, each noun is interpreted as a **"property,"** represented here by a set
- *every* is interpreted as a function from properties to properties to truth values
- For example, each constituent of our example derivation is interpreted as follows:



inference: (*Axioms*) every n_j is an n_j (*Rules*) every n_j is a n_l every n_l is a n_k / every n_j is a n_k

- With the new analysis of the language, the inference methods apply as before.
- Since the fragment is sound and complete, it gives the whole story about *every*, in a sense.

10.4 Recognizing sentences: CKY parsing

Chomsky normal form grammars have rules of only the following forms,

A→BC A→w

These grammars allow an especially simple CKY-like tabular parsing method, based on proposals of Cocke, Kasami and Younger [1, 11, 12]. The soundness and completeness of the CKY method are shown together with a prolog implementation in [11]. For a sentence of length n, the maximum number of steps needed to compute the closure is proportional to n^3 or less [1, §4.2.1]. There are other, more complex and slightly more efficient 'tabular' parsing methods like Earley's [4, 3] or Graham, Harrison and Ruzzo's [5]. We will not discuss them unless someone wants to.

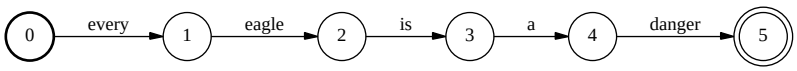
(10) Adapting the standard CKY method to merge grammars we get this:

Input: An arbitrary sequence of words $w_1 w_2 \ldots w_n$
Output: A decision (yes/no) about whether $w_1 w_2 \ldots w_n \in expression(C, G)$
$T ::= \{(i-1, i, w_i) \mid 1 \leq i \leq n\}$
while any of the following apply **do**
if $(i, j, w) \in T$ and $w :: \gamma$ **then** $T ::= T \cup \{(i, j :: \gamma)\}$ **endif**
if $\epsilon :: \gamma$ and $0 \leq i \leq n$ **then** $T ::= T \cup \{(i, i :: \gamma)\}$ **endif**
if $(i, j :: =f\gamma), (j, k \cdot f) \in T$ **then** $T ::= T \cup \{(i, k : \gamma)\}$ **endif**
if $(i, j : =f\gamma), (h, i \cdot f) \in T$ **then** $T ::= T \cup \{(h, j : \gamma)\}$ **endif**
end while
if $(0, n, C) \in T$ **then** yes **else** no **endif**

Algorithm CKY(G)

(11) **Continuing the same example.**

A string input can be regarded as a finite state machine:



The CKY method can apply to any finite state machine representation, and computing the CKY closure can be regarded as adding some new arcs to the finite state graph or as filling in the "upper triangle" of an $n \times n$ matrix, from the diagonal up,[1]

	0	1	2	3	4	5
0	(0,0::=V =Dq I)	(0,1::=N Dq) (0,1,every)	(0,2:Dq)			(0,5:I)
1		(1,1::=V =Dq I)	(1,2::N) (1,2,eagle)			
2			(2,2::=V =Dq I)	(2,3::=Di V) (2,3,is)		(2,5:=V I) (2,5:V)
3				(3,3::=V =Dq I)	(3,4::=N Di) (3,4,a)	(3,5:Di)
4					(4,4::=V =Dq I)	(4,5::N) (4,5,danger)
5						(5,5::=V =Dq I)

This table represents all derivations, all analyses of all substrings, and so it is not difficult to collect a derivation tree from any successful recognition.

10.5 Recognizing logical derivations

(12) We noticed that our logical derivations can be regarded as context free derivation trees. Let's see how this is literally true.

(13) Let the sentences of the fragment be the categories in a new grammar. Then for any nouns n_j, n_k, n_l we have the rule:

$$(\text{every } n_j \text{ is a } n_l) \rightarrow (\text{every } n_j \text{ is a } n_k)\ (\text{every } n_k \text{ is a } n_l).$$

We don't need any "pronounced," "terminal" symbols, so let's assume that we have empty rewrite rules like this for every category symbol:

$$(\text{every } n_j \text{ is a } n_k) \rightarrow \epsilon$$

Input: An set of sentences A and expression e
Output: A decision (yes/no) about whether $A \vdash e$
  $T ::= A \cup Axioms$ (the "empty productions" corresponding to $A \cup Axioms$)
  while any of the following apply **do**
    if $N1, N2 \in T$ and $N3 \rightarrow N1\ N2$ **then** $T ::= T \cup \{N3\}$ **endif**
  end while
  if $e \in T$ **then** yes **else** no **endif**

specializing CKY to decide if $A \vdash e$

[1] CKY tables and other similar structures of intermediate results are frequently constructed by matrix operations. This idea has been important in complexity analysis and in attempts to find the fastest possible parsing methods [16, 7]. Extensions of matrix methods to more expressive grammars are considered by [10] and others.

10.6 Metatheory: canonical models

In the previous section we saw that we could check $A \vdash e$ by, in effect, calculating the transitive, reflexive closure of the subset relations indicated by A. Since $\models$ is the same relation as $\vdash$ for this fragment, we can view what is happening in the previous section as the calculation of a certain, particular model.

(14) In this fragment, it is possible to construct a certain simple model for any set A of sentences, where that model verifies exactly the things that can be derived from A.

Given any set A, define the canonical model μ_A as follows. We let the universe E be the set of nouns. Then define a relation $\preceq$ on the (string part of the) nouns this way:

$$n_j \preceq n_k \text{ iff (every } n_j \text{ is a } n_k : I) \in A.$$

And let $\preceq^*$ be the reflexive, transitive closure of $\preceq$. Then we interpret each noun $(n_j : \mathrm{N})$ in the language as follows:

$$\mu_A(n_j : N) = \{n_k |\ n_k \preceq^* n_j\}.$$

We claim that μ_A is a canonical model of the kind required by the following claim, following Moss. We have immediately:

(15) $\mu_A(n_i, N) \subseteq \mu_A(n_j, N)$ iff $n_i \preceq^* n_j$

(16) μ_A is a model of A, that is, μ_A verifies A.

Proof: Suppose A contains sentence e = (every n_i is a $n_j : I$) $\in L(G)$. Then by the definition of μ_A, $\mu_A(n_j, N)$ will contain everything that $\mu_A(n_i, N)$ contains, and so μ_A verifies e. □

(17) **(Canonical model property)** There is a model μ that verifies exactly the sentences that A derives.

Proof: Consider any set A and any sentence e = (every n_i is a $n_j : I$) $\in L(G)$. We establish the theorem by showing that μ_A verifies e iff $A \vdash e$.

($\Rightarrow$) Assume μ_A verifies e = (every n_i is a $n_j : I$) $\in L(G)$. By (15), $n_i \preceq^* n_j$. So by the definition of μ_A, A has a transitive chain

$$(\text{every } n_1 \text{ is a } n_2 : I), (\text{every } n_1 \text{ is a } n_2 : I), \ldots, (\text{every } n_{k-1} \text{ is a } n_k : I)$$

with $n_1 = n_i$ and $n_k = n_j$, in which case there is also a derivation $A \vdash e$.

($\Leftarrow$) Assume $A \vdash e$. Then by soundness $A \models e$, that is, every model of A is a model of e. So it suffices to show that μ_A is a model of A, which we have from (16). □

References for Lecture 10

[1] AHO, A. V., AND ULLMAN, J. D. *The Theory of Parsing, Translation, and Compiling. Volume 1: Parsing*. Prentice-Hall, Englewood Cliffs, New Jersey, 1972.

[2] CHOMSKY, N. *The Minimalist Program*. MIT Press, Cambridge, Massachusetts, 1995.

[3] EARLEY, J. *An Efficient Context-Free Parsing Algorithm*. PhD thesis, Carnegie-Mellon University, 1968.

[4] EARLEY, J. An efficient context-free parsing algorithm. *Communications of the Association for Computing Machinery 13* (1970), 94–102.

[5] GRAHAM, S. L., HARRISON, M. A., AND RUZZO, W. L. An improved context free recognizer. *Association for Computing Machinery Transactions on Programming Languages and Systems 2* (1980), 415–462.

[6] GREENBERG, J. Some universals of grammar with particular reference to the order of meaningful elements. In *Universals of Human Language*, J. Greenberg, Ed. Stanford University Press, Stanford, California, 1978.

[7] LEE, L. Fast context-free parsing requires fast Boolean matrix multiplication. In *Proceedings of the 35th Annual Meeting, ACL'97* (1997), Association for Computational Linguistics.

[8] PRATT-HARTMANN, I. Fragments of language. *Journal of Logic, Language and Information 13*, 2 (2004), 207–223.

[9] PRATT-HARTMANN, I., AND THIRD, A. More fragments of language. *Notre Dame Journal of Formal Logic Forthcoming* (2004).

[10] SATTA, G. Tree adjoining grammar parsing and boolean matrix multiplication. *Computational Linguistics 20* (1994), 173–232.

[11] SHIEBER, S. M., SCHABES, Y., AND PEREIRA, F. C. N. Principles and implementation of deductive parsing. Tech. Rep. CRCT TR-11-94, Computer Science Department, Harvard University, Cambridge, Massachusetts, 1993.

[12] SIKKEL, K., AND NIJHOLT, A. Parsing of context free languages. In *Handbook of Formal Languages, Volume 2: Linear Modeling*, G. Rozenberg and A. Salomaa, Eds. Springer, NY, 1997, pp. 61–100.

[13] STABLER, E. P. Derivational minimalism. In *Logical Aspects of Computational Linguistics*, C. Retoré, Ed. Springer-Verlag (Lecture Notes in Computer Science 1328), NY, 1997, pp. 68–95.

[14] STABLER, E. P., AND KEENAN, E. L. Structural similarity. In *Algebraic Methods in Language Processing, AMiLP 2000* (University of Iowa, 2000), A. Nijholt, G. Scollo, T. Rus, and D. Heylen, Eds. Revised version forthcoming in *Theoretical Computer Science*.

[15] TRUESWELL, J., AND GLEITMAN, L. Children's eye movements during listening: Developmental evidence for a constraint-based theory of lexical processing. University of Pennsylvania, 2004.

[16] VALIANT, L. General context free recognition in less than cubic time. *Journal of Computer and System Sciences 10* (1975), 308–315.

[17] WITTGENSTEIN, L. *Philosophical Investigations*. MacMillan, NY, 1958. This edition published in 1970.

11 Languages with *every* and relative clauses: grammar

11.1 Summary

(1) The soundness and completeness of the fragment with *every* reveals a remarkable and easy reasoning power.

It is no wonder that we do not find anything like *every* in animal languages.

Frege's insight might help us see why: quantifiers require a significant abstraction step, so that instead of just classifying things we can talk about relations among classifications.

(2) Considering the *every* fragment though, there are still many simple things that cannot be expressed. For example, suppose we have the nouns *male, leopard*, and *danger*, and we realize that the male leopards are dangers but the females are not – then we might want to express the fact:

$$(\text{male} \cap \text{leopard}) \subseteq \text{danger}.$$

The language does not provide a way to do this, without introducing a new word of some kind!

11.2 Adding relative clauses

(3) Human languages have a large variety of relative clause constructions. In English:

i. the woman [[who] _____ talked to you] left
the woman [[that] _____ talked to you] left
* the woman [[] _____ talked to you] left

ii. the woman [[who] you like _____] left
the woman [[that] you like _____] left
the woman [[] you like _____] left

iii. the woman [[who] I was talking to _____] left early
the woman [[that] I was talking to _____] left early
the woman [[] I was talking to _____] left early

iv. the woman [[to whom] I was talking _____] left

v. the woman [[whose money] you took _____] left

vi. the box [[which] I brought _____] was heavy
the box [[that] I brought _____] was heavy
the box [[] I brought _____] was heavy

vii. the reason [[why] he left _____] is mysterious

viii. the day [[when] he left _____] was the beginning of the end

ix. the day [[] he left _____] was the beginning of the end

x. the city [[where] I live _____] is beautiful

xi. * John [[who] I was talking to _____] left early
* my mother [[who] I was talking to _____] left early

"nonrestrictive" relatives:

xii. John, who I told you about _____ before, left early
My cat, which I told you about _____ before, left early
* John, [] I told you about _____ before, left early

"stacking" relatives:

xiii. The woman [[who] _____ talked to you] [[who] _____ came from Japan] left.

xiv. The woman [[who] _____ talked to you] [[who] you liked _____] left.

xv. * The woman [[] you like _____] [[] you saw _____ in Japan] left

xvi. John, [[who] _____ talked to you] [[who] _____ came from Japan] left.

(4) Roughly speaking: relative clauses look like sentences in which a question phrase has been moved from its usual position to the front.

To define languages like this, we need to be able to handle the dependency between the initial wh-phrase and some position in the clause.

(5) Notice that this relative clause dependency will typically be missed by bigrams or trigrams:

common:	he likes the	as in: *he likes the music*
rare:	he likes understands	??
structural conditioning:	*the teacher he likes understands the question*	

(6) We will treat this dependency as a kind of "movement," extending the merge grammars with two new kinds of features: licensing features +X and licensee features -X.

A -X feature marks a phrase that needs to be licensed, and +X features triggers the movement of a -X phrase to its licensed position.

Let's work an example before presenting the formal definitions.

(7) In our fragment, which has only one form of one verb, *be*, let's first get just relative clauses of this form:

every N [[which] _____ is (also) a N] is a N

as in

every leopard which is also a male is a danger.

With a reasonable grammar for this construction, we set the stage for later developments.

Grammar for the fragment:

snake::N	every::=N Dq
leopard::N	a::=N Di
eagle::N	is::=Di V
danger::N	ϵ::=V =Dq I
male::N	
which::Dq -wh	ϵ::=I +wh C
	ϵ::=C =N N

Only add the last 3 lexical items introduce something really new. The word *which* is a quantificational determiner Dq which needs to wh-licensing. There is an empty element which combines with an I and then licenses a wh-phrase to form a C (a "complementizer phrase"). And finally there is an empty element which allows a complementizer phrase to modify a noun, forming a more complex noun phrase N.

To understand how these new pieces go together, remember that with the merge grammar we could derive the pieces

every leopard:Dq is a male:=Dq I

These two pieces could then be merged to produce

every leopard is a male: I.

In our extended grammar, though, we need something special to happen when we combine a wh-phrase, because wh-phrases play two roles in the construction of the sentence (sometimes in very different positions in the sentence, as we saw above!). So picking up at this point, what we propose is this:

step 1: merge. We merge

which:Dq -wh is a male:=Dq I

but the two phrases do not get concatenated. Instead, they remain separate parts in the resulting expression:

is a male:I, which:-wh

What this means is that we have a phrase whose head is inflection I, pronounced *is a male*, but also containing another part which still needs to be placed into a wh-licensed position, namely *which*. That part is a *mover*, a phrase that is moving to its final position.

step 2: merge. Now we can combine the I phrase with a lexical item:

ϵ::=I +wh C is a male:I, which:-wh

These are merged as before, except that the mover just tags along, yielding this single expression that contains the same moving part:

is a male:+wh C, which:-wh.

step 3: move. Now the interesting thing happens. Because the head can license a wh-phrase, we move the wh-mover and attach it, in a step we call *move*. From

is a male:+wh C, which:-wh.

we get

which is a male:C.

step 4: merge. The final steps involve only merges of the sort we saw before. We can merge the relative clause with a lexical item:

ϵ::=C =N N which is a male:C

to yield

which is a male:=N N

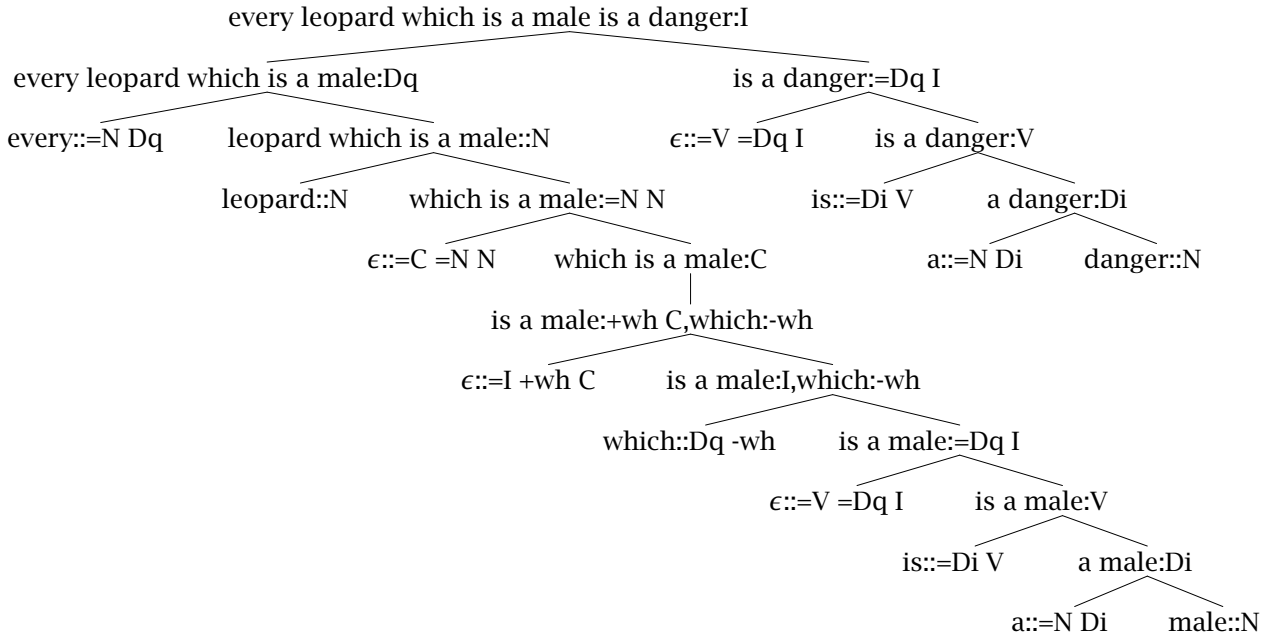
step 5: merge. Then we can merge with another lexical item

which is a male:=N N leopard::N

to yield

leopard which is a male:N.

Then the rest of the derivation can continue as in the merge grammar. So the whole derivation is this:



(8) **Miminalist grammars: formal definition**

A minimalist grammar $G = (\Sigma, F, Types, Lex, \mathcal{F})$, where

Alphabet $\Sigma \neq \emptyset$

Features $F = base$ (basic features, $\neq \emptyset$)

$\cup \{=f \mid f \in base\}$ (selection features)

$\cup \{+f \mid f \in base\}$ (licensor features)

$\cup \{-f \mid f \in base\}$ (licensee features)

Types $= \{::, :\}$ (lexical, derived)

For convenience: Chains $C = \Sigma^* \times Types \times F^*$

Expressions $E = C^+$ (nonempty sequences of chains)

Lexicon $Lex \subseteq C^+$ is a finite subset of $\Sigma^* \times \{::\} \times F^*$.

Generating functions $\mathcal{F} = \{merge, move\}$, partial functions from E* to E, defined below.

(9) The generating functions $merge$ and $move$ are partial functions from tuples of expressions to expressions. We present the generating functions in an inference-rule format for convenience, "deducing" the value from the arguments. We write st for the concatenation of s and t, for any strings s, t, and let ϵ be the empty string.

$merge : (E \times E) \to E$ is the union of the following 3 functions, for $s, t \in \Sigma^*$, for $\cdot \in \{:, ::\}$, for $f \in base$, $\gamma \in F^*$, $\delta \in F^+$, and for chains $\alpha_1, \ldots, \alpha_k, \iota_1, \ldots, \iota_l$ $(0 \leq k, l)$

$$\frac{s :: =f\,\gamma \qquad t \cdot f, \alpha_1, \ldots, \alpha_k}{st : \gamma, \alpha_1, \ldots, \alpha_k}$$ merge1: lexical item selects a non-mover

$$\frac{s : =f\,\gamma, \alpha_1, \ldots, \alpha_k \qquad t \cdot f, \iota_1, \ldots, \iota_l}{ts : \gamma, \alpha_1, \ldots, \alpha_k, \iota_1, \ldots, \iota_l}$$ merge2: derived item selects a non-mover

$$\frac{s \cdot =f\,\gamma, \alpha_1, \ldots, \alpha_k \qquad t \cdot f\delta, \iota_1, \ldots, \iota_l}{s : \gamma, \alpha_1, \ldots, \alpha_k, t : \delta, \iota_1, \ldots, \iota_l}$$ merge3: any item selects a mover

Notice that the domains of merge1, merge2, and merge3 are disjoint, so their union is a function.

$move : E \to E$ is the union of the following 2 functions, for $s, t \in \Sigma^*$, $f \in base$, $\gamma \in F^*$, $\delta \in F^+$, and for chains $\alpha_1, \ldots, \alpha_k, \iota_1, \ldots, \iota_l$ $(0 \leq k, l)$ satisfying the following condition,

(SMC) none of $\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k$ has $-f$ as its first feature,

$$\frac{s : +f\gamma, \alpha_1, \ldots, \alpha_{i-1}, t : -f, \alpha_{i+1}, \ldots, \alpha_k}{ts : \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k}$$ move1: final move of licensee phrase

$$\frac{s : +f\gamma, \alpha_1, \ldots, \alpha_{i-1}, t : -f\delta, \alpha_{i+1}, \ldots, \alpha_k}{s : \gamma, \alpha_1, \ldots, \alpha_{i-1}, t : \delta, \alpha_{i+1}, \ldots, \alpha_k}$$ move2: nonfinal move of licensee phrase

Notice that the domains of move1 and move2 are disjoint, so their union is a function.

(The (SMC) restriction on the domain of $move$ is a simple version of the "shortest move condition" [5].

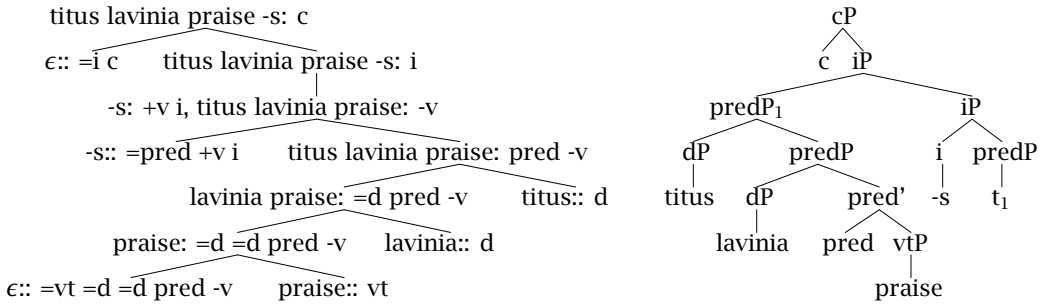
11.3 The power of minimalist grammars

For practice, it is useful to see how some of the basic, common word orders in human languages can be derived with these minimalist grammars, in a way that highlights their similarities. It is common to use the start category C or c (for "complementizer").

(10) **SOVI: Naive Tamil**. We first consider some very simple examples inspired by Mahajan [14]. The order Subject-Object-Verb-Inflection is defined by the following grammar:

```
lavinia::d                  titus::d
praise::vt                  criticize::vt
laugh::v                    cry::v
ϵ::=i c                     -s::=pred +v i
ϵ::=vt =d =d pred -v        ϵ::=v =d pred -v
```

Notice that the `-s` in the string component of an expression signals that this is an affix, while the `-v` in the feature sequence of an expression signals that this item must move to a `+v` licensing position. With this lexicon, we have the following derivation of the sentence *titus lavinia praise -s*, showing the derivation tree on the left and the more traditional X-bar tree on the right:



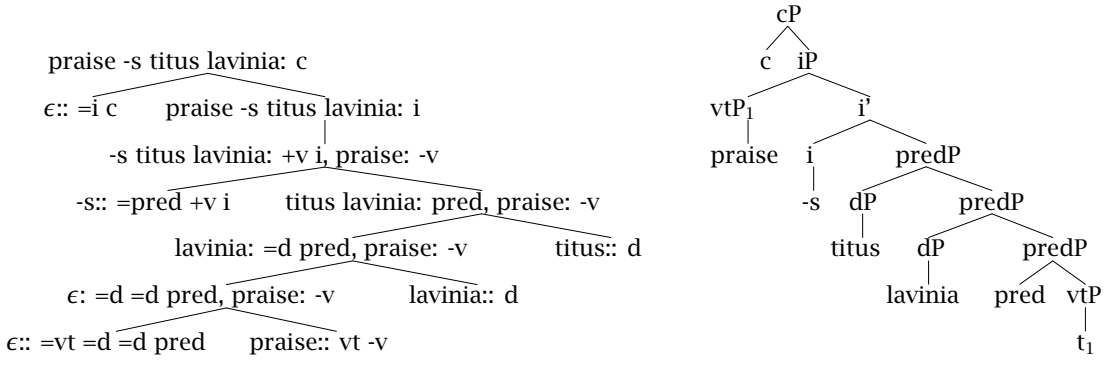
These conventional structures show some aspects of the history of the derivations, something which can be useful for linguists even though it is not necessary for the calculation of derived expressions.

(11) **VISO: Naive Zapotec**

An VSO language like Zapotec can be obtained by letting the verb select its object and then its subject, and then moving the just the lowest part of the SOV complex move to the "specifier" of I(nflection). The following 10 lexical items provide a naive grammar of this kind:

```
lavinia::d                  titus::d
praise::vt -v               laugh::v -v
ϵ::=i c                     -s::=pred +v i
ϵ::=vt =d =d pred           ϵ::=v =d pred
```

With this lexicon, we have the following derivation of the sentence *praise -s titus lavinia*



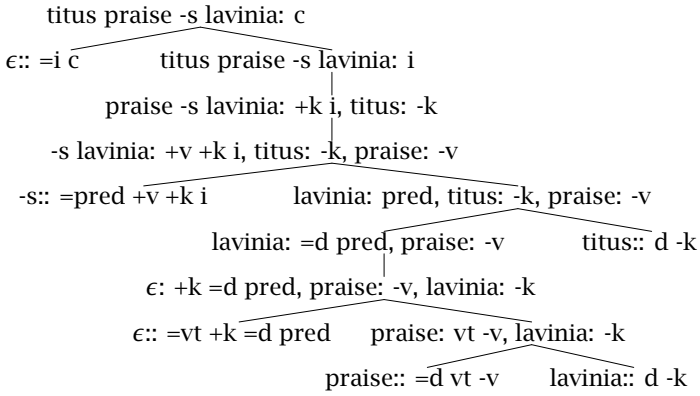
(12) **SVIO: naive English**

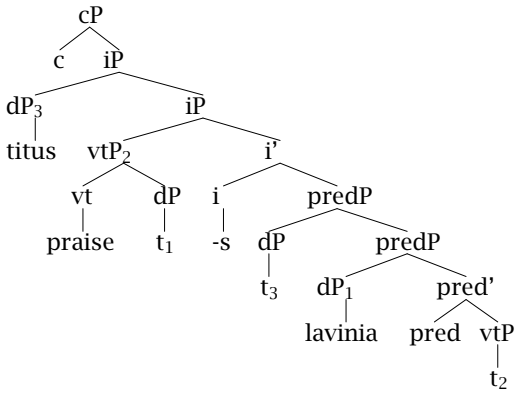
The following 16 lexical items provide a slightly more elaborate fragment of an English-like SVIO language:

lavinia:: d -k	titus:: d -k	who:: d -k -wh	
some:: =n d -k	every:: =n d -k	noble:: n	kinsman:: n
laugh:: =d v -v	cry:: =d v -v		
praise:: =d vt -v	criticize:: =d vt -v		
-s:: =pred +v +k i	ϵ:: =vt +k =d pred	ϵ:: =v pred	
ϵ:: =i c	ϵ:: =i +wh c		

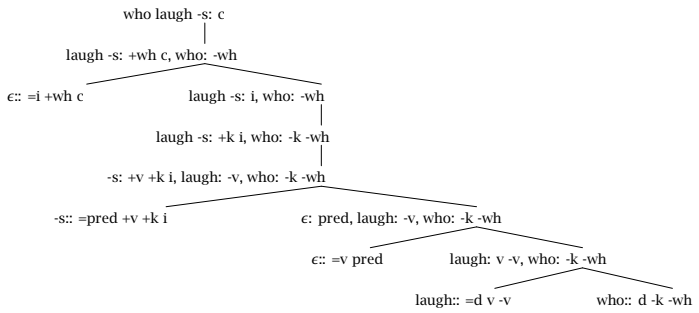
Notice that an SVIO language must break up the underlying SVO complex, so that the head of inflection can appear postverbally. This may make the SVIO order more complex to derive than the SOVI and VISO orders, as in our previous examples.

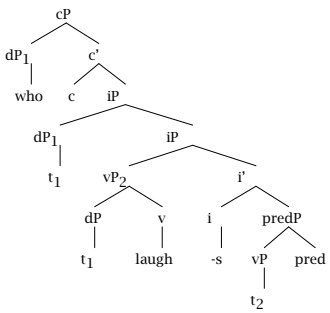
With this lexicon, we have the following derivation of the sentence *titus praise -s lavinia*



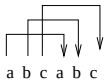


These lexical items allow wh-phrases to be fronted from their "underlying" positions, so we can derive *who laugh -s* and (since "do-support" is left out of the grammar for simplicity) *who titus praise -s*:





(13) **XX: reduplication** There are many cases of reduplication in human languages. In reduplication, each element of the reduplicated pattern must correspond to an element in the "reduplicant," and these correspondences are "crossing:"



Chomsky noticed this crossing dependency between verbs and the following affix in 1956, but it is bounded:

John will have -∅ be -en eat -ing pie

The extent of dependencies in English auxiliary constructions is bounded, but we can get unbounded crossing dependencies between subjects and verbs in Dutch constructions like the following [12, 2]:

… because I Cecilia Henk the hippo saw help feed
… omdat ik Cecilia Henk de nijlpaarden zag helpen voeren

And in Swiss-German, the relation between verbs and their objects is signaled not only by word order but also by case marking [21]:

… that we the children Hans the house let help paint
… das mer d'chind em Hans es huus lönd hälfe aastriiche

In Chinese, one way to form a yes/no-question involves phrasal reduplication [22]:

Zhangsan like play basketball not like play basketball
Zhangsan ai da lanqiu (*,) bu ai da lanqiu
'Does Zhangsan like to play basketball?'

Zhangsan like play basketball not like play volleyball
Zhangsan ai da lanqiu *(,) bu ai da paiqiu
'Zhangsan likes to play basketball, not to play volleyball'

We find contrastive focus reduplication in English [7]:

we're not LIVING TOGETHER living together

We find whole word reduplication in the African language Bambara [6],

whatever dog w ulu -o- wulu (Bambara)

We find various kinds of reduplication in the American indigenous language Pima [19]:

mountain lion -s g egevho (Pima)

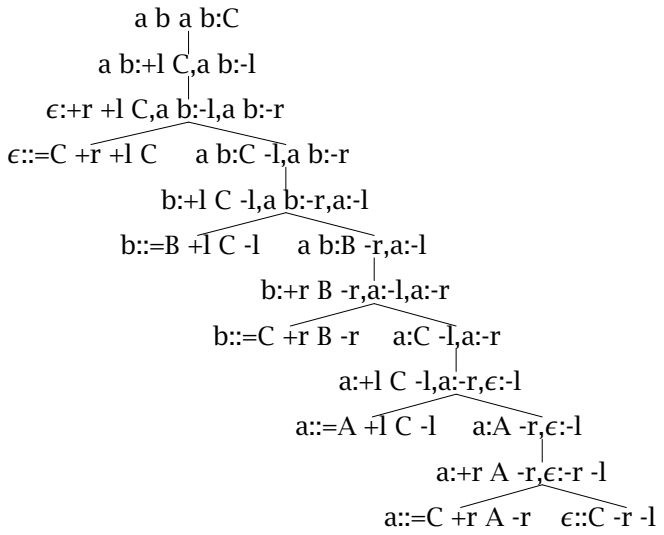
We find partial reduplication in the Australian language Mangarayi [13], the American language Yaqui [11], the Austronesian language Manam [3], and very many others.

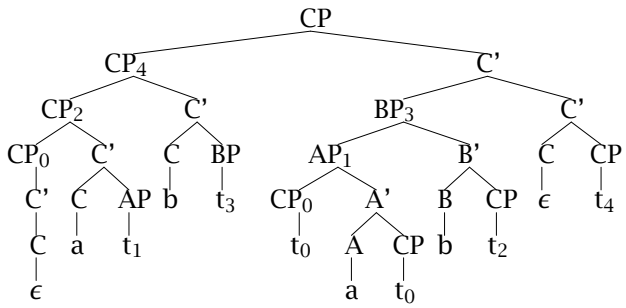
(14) Unbounded crossing correspondences are beyond the power of CFGs, but are easily captured in MGs. We can provide a grammar for the language $\{xx \mid x \in \{a,b\}^*\}$, using two features to trigger movement

-l(eft) and -r(ight).

The following grammar with 6 lexical items does the trick:

a::=A +l C -l	b::=B +l C -l	ϵ::=C +r +l C
a::=C +r A -r	b::=C +r B -r	ϵ::C -r -l





(15) Michaelis and Harkema [15, 17, 9, 10, 16] showed that the expressive power of MGs is exactly the same as "multiple context free grammars" [18, 20], "(set local) multiple component tree adjoining grammars" [24], "simple range concatenation grammars" [1], and many others [23].

11.4 Recognizing an MG language: CKY

We can adapt the CKY method simply by introducing a clause for each of the cases of each rule: merge1, merge2, merge3, move1, and move2.

Input: An arbitrary sequence of words $w_1 w_2 \ldots w_n$
Output: A decision (yes/no) about whether $w_1 w_2 \ldots w_n \in \mathit{expression}(C, G)$
$T ::= \{(i-1, i, w_i) \mid 1 \leq i \leq n\}$
while any of the following apply **do**
if $(i, j, w) \in T$ and $w :: \gamma$ **then** $T ::= T \cup \{(i, j :: \gamma)\}$ **endif** (overt lex features)
if $\epsilon :: \gamma$ and $0 \leq i \leq n$ **then** $T ::= T \cup \{(i, i :: \gamma)\}$ **endif** (empty lex features)
if $(i, j :: =f\gamma), (j, k \cdot f, \alpha_1, \ldots, \alpha_k) \in T$ **then**
$T ::= T \cup \{(i, k : \gamma), \alpha_1, \ldots, \alpha_k\}$ (merge1)
end if
if $(i, j : =f\gamma, \alpha_1, \ldots, \alpha_k), (h, i \cdot f, \iota_1, \ldots, \iota_l) \in T$ **then**
$T ::= T \cup \{(h, j : \gamma), \alpha_1, \ldots, \alpha_k, \iota_1, \ldots, \iota_l\}$ (merge2)
end if
if $(i, j : =f\gamma, \alpha_1, \ldots, \alpha_k), (d, e : f\delta, \iota_1, \ldots, \iota_l) \in T$ and $\delta \neq \epsilon$ **then**
$T ::= T \cup \{(i, j : \gamma), \alpha_1, \ldots, \alpha_k, (d, e : \delta), \iota_1, \ldots, \iota_l\}$ (merge3)
end if
if $(i, j : +f\gamma, \alpha_1, \ldots, \alpha_{k-1}, (h, i : -f), \alpha_{k+1} \ldots, \alpha_l) \in T$ and no other $-f$ is exposed **then**
$T ::= T \cup \{(h, j : \gamma), \alpha_1, \ldots, \alpha_l\}$ (move1)
end if
if $(i, j : +f\gamma, \alpha_1, \ldots, \alpha_{k-1}, (d, e : -f\delta), \alpha_{k+1} \ldots, \alpha_l) \in T$, $\delta \neq \epsilon$, no other $-f$ exposed **then**
$T ::= T \cup \{(i, j : \gamma), \alpha_1, \ldots, \alpha_{k-1}, (d, e : \delta), \alpha_{k+1} \ldots, \alpha_l\}$ (move2)
end if
end while
if $(0, n, C) \in T$ **then** yes **else** no **endif**

Algorithm CKY(MG)

This algorithm and many variants are studied in [8, 10].

Exercise 5 Due Thursday, Feb 17 (2 exercises were optional, so this is the 3rd req'd exercise)

(1) Write a merge grammar (no +f or -f features that would trigger movements) for the language

$$a^n b^n = \{\epsilon, ab, aabb, aaabbb, \ldots\}.$$

(2) Using the grammar you provided in (1), show the complete CKY table for the input `aabb`

(3) Using the "naive English" grammar on page 103, draw the complete derivation tree for the sentence *who titus praise -s*.
(I don't want the X-bar style tree that I showed on the right in some cases, but the derivation-style tree of the sort shown on the left, where every merge and move step is shown.)

References for Lecture 11

[1] Boullier, P. Proposal for a natural language processing syntactic backbone. Tech. Rep. 3242, Projet Atoll, INRIA, Rocquencourt, 1998.

[2] Bresnan, J., Kaplan, R. M., Peters, S., and Zaenen, A. Cross-serial dependencies in Dutch. *Linguistic Inquiry 13*, 4 (1982), 613–635.

[3] Buckley, E. Integrity and correspondence in Manam double reduplication. In *Proceedings of the North Eastern Linguistic Society, NELS 27* (1997).

[4] Chomsky, N. Three models for the description of language. *IRE Transactions on Information Theory IT-2* (1956), 113–124.

[5] Chomsky, N. *The Minimalist Program*. MIT Press, Cambridge, Massachusetts, 1995.

[6] Culy, C. The complexity of the vocabulary of Bambara. *Linguistics and Philosophy 8*, 3 (1985), 345–352.

[7] Ghomeshi, J., Jackendoff, R., Rosen, N., and Russell, K. Contrastive focus reduplication in English. *Natural Language and Linguistic Theory 22*, 2 (2004), 307–357.

[8] Harkema, H. A recognizer for minimalist grammars. In *Sixth International Workshop on Parsing Technologies, IWPT'2000* (2000).

[9] Harkema, H. A characterization of minimalist languages. In *Logical Aspects of Computational Linguistics* (NY, 2001), P. de Groote, G. Morrill, and C. Retoré, Eds., Lecture Notes in Artificial Intelligence, No. 2099, Springer, pp. 193–211.

[10] Harkema, H. *Parsing Minimalist Languages*. PhD thesis, University of California, Los Angeles, 2001.

[11] Harley, H., and Amarillas, M. Reduplication multiplication in Yaqui: Meaning x form. In *Studies in Uto-Aztecan*, L. M. Barragan and J. D. Haugen, Eds., MIT Working Papers on Endangered and Less Familiar Languages #5. MIT, Cambridge, Massachusetts, 2003.

[12] Huybregts, M. Overlapping dependencies in Dutch. Tech. rep., University of Utrecht, 1976. Utrecht Working Papers in Linguistics.

[13] Kurisu, K., and Sanders, N. Infixal nominal reduplication in Mangarayi. *Phonology at Santa Cruz 6* (1999), 47–56.

[14] Mahajan, A. Eliminating head movement. In *The 23rd Generative Linguistics in the Old World Colloquium, GLOW '2000, Newsletter #44* (2000), pp. 44–45.

[15] Michaelis, J. Derivational minimalism is mildly context-sensitive. In *Proceedings, Logical Aspects of Computational Linguistics, LACL'98* (NY, 1998), Springer.

[16] Michaelis, J. *On Formal Properties of Minimalist Grammars*. PhD thesis, Universität Potsdam, 2001. *Linguistics in Potsdam 13*, Universitätsbibliothek, Potsdam, Germany.

[17] Michaelis, J. Transforming linear context free rewriting systems into minimalist grammars. In *Logical Aspects of Computational Linguistics* (NY, 2001), P. de Groote, G. Morrill, and C. Retoré, Eds., Lecture Notes in Artificial Intelligence, No. 2099, Springer, pp. 228–244.

[18] Nakanishi, R., Takada, K., and Seki, H. An efficient recognition algorithm for multiple context free languages. In *Proceedings of the Fifth Meeting on Mathematics of Language, MOL5* (1997).

[19] Riggle, J. Infixing reduplication in Pima and its theoretical consequences. UCLA. Publication forthcoming, 2003.

[20] SEKI, H., MATSUMURA, T., FUJII, M., AND KASAMI, T. On multiple context-free grammars. *Theoretical Computer Science 88* (1991), 191–229.

[21] SHIEBER, S. M. Evidence against the context-freeness of natural language. *Linguistics and Philosophy 8*, 3 (1985), 333–344.

[22] STABLER, E. P. Varieties of crossing dependencies: Structure dependence and mild context sensitivity. *Cognitive Science 93*, 5 (2004), 699–720.

[23] VIJAY-SHANKER, K., AND WEIR, D. The equivalence of four extensions of context free grammar formalisms. *Mathematical Systems Theory 27* (1994), 511–545.

[24] WEIR, D. *Characterizing mildly context-sensitive grammar formalisms.* PhD thesis, University of Pennsylvania, Philadelphia, 1988.

12 Languages with *every* and relative clauses: conclusion

12.1 The language with *every* and relative clauses: recap

(1) **syntax G:** (we allow any finite number of different nouns like these 5)

snake::N	every::=N Dq
leopard::N	a::=N Di
eagle::N	is::=Di V
danger::N	ϵ::=V =Dq I
male::N	
which::Dq -wh	ϵ::=I +wh C
	ϵ::=C =N N

$\mathcal{F} = \{merge, move\}$

(2) Unlike the simple fragment and the every fragment, this language has infinitely many sentences. The sentences in this language, the strings of category I, have the form

every N {which is a N}* is a N {which is a N}*

Notice that this is an infinite regular language, which we have described with a mildly context sensitive grammar.

(3) Today we provide a semantics and inference system, and consider the metatheory.

12.2 Semantics of the fragment

For this restricted fragment, a simple extension of our semantics suffices:

semantics: $\mu : \Gamma(G) \rightarrow (E, 2)$ defined as follows

$\mu(n :: \text{N}) \subseteq E$
$\mu(every :: \text{=N Dq})$ is the function EVERY that maps a property P to EVERY(P), where EVERY(P) is the function that maps property Q to 1 iff $P \subseteq Q$
$\mu(a :: \text{=N Di}) = \mu(is :: \text{=Di V}) = \mu(\epsilon :: \text{=V =Dq I})$
= the identity function on properties,idP

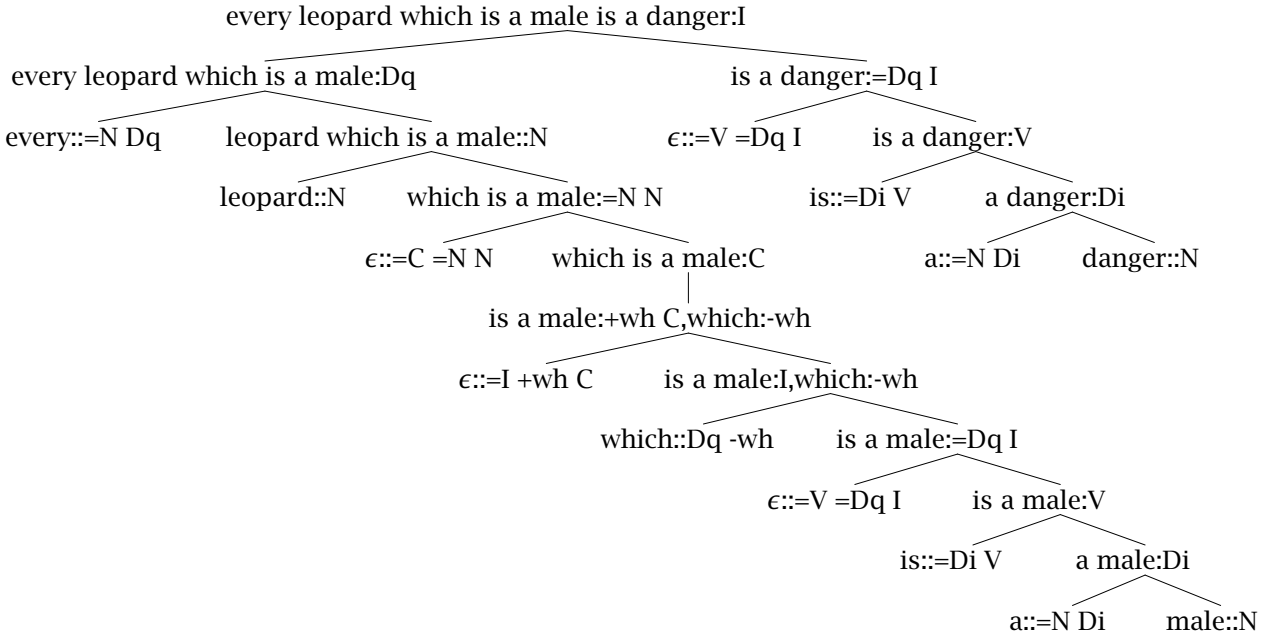
$$\mu(merge(A, B)) = \begin{cases} \mu(A)(\mu(B)) & \text{if defined} \\ \mu(B)(\mu(A)) & \text{otherwise} \end{cases}$$

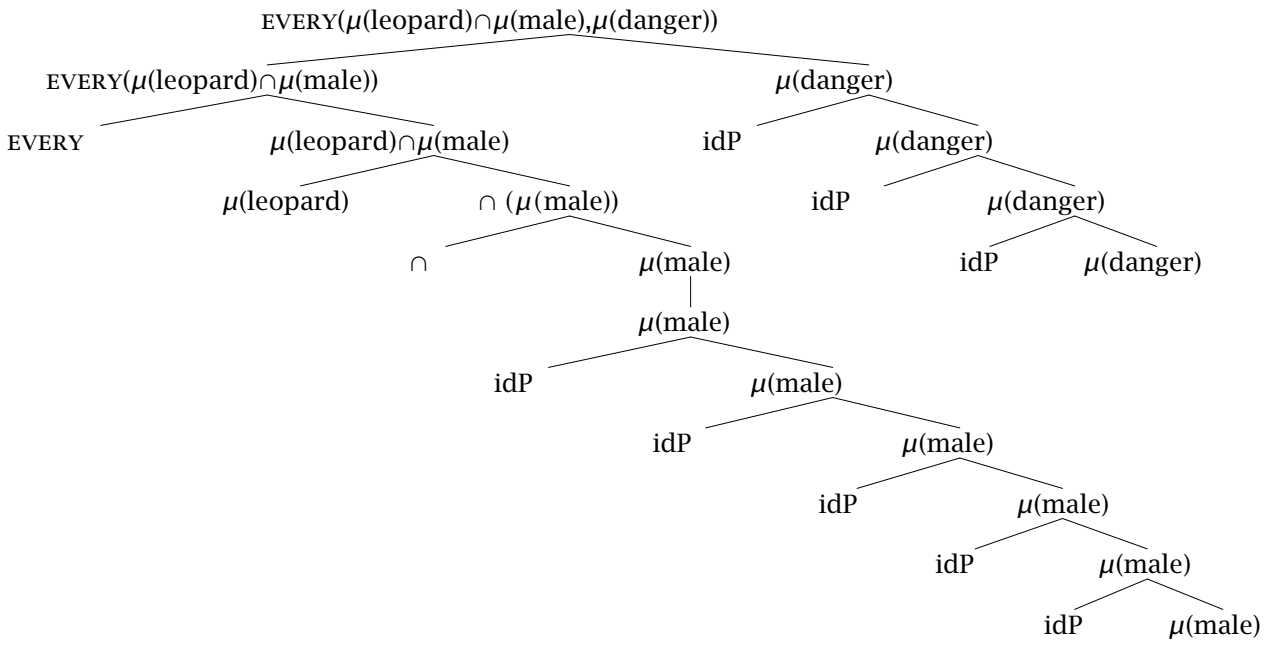
$\mu(which :: \text{=Dq -wh}) = idP$
$\mu(\epsilon :: \text{=I +wh C}) = idP$
$\mu(\epsilon :: \text{=C =N N})$ is the function $\cap$ that maps a property P to $\cap(P)$, where $\cap(P)$ is the function that maps property Q to $P \cap Q$
$\mu(move(A)) = \mu(A)$

(4) This semantics interprets every expression in the language, as we can see immediately from the fact that every lexical item is interpreted and every result of applying a structure building rule is interpreted.

(5) This semantic proposal for *move* suffices for the moment, but it is not an accurate idea about English or any other human language with movement (as anyone who has studied syntax knows very well already).
(And notice that it would make no sense for a language to have an operation like this without any semantic significance!)

Applying this semantics to our first example, we can interpret every expression that appears in our example syntactic derivation, as follows:





12.3 Inference for every A_0 A_1, B_0

(6) For any nouns A_0 A_1…A_i, B_0 B_1…which is a B_j, let

every A_0 A_1…A_i, B_0 B_1…B_j

refer to the sentence

every A_0 which is a A_1…which is a A_i is a B_0 which is a B_1…which is a B_j.

To simplify the inference problem, let's first restrict our attention to sentences of the form

every A_0 A_1, B_0

(7) *(Axioms)* For any nouns A_0, A_1, the following is an axiom (any $i \geq 0$):

every A_0 A_1, A_0

(8) *(Rules)* We present the inference steps in **linguists' style on left**, **logicians' style on right**. For any nouns A_0, A_1, B_0:

(Permutation) The order of the nouns in the subject does not matter:

every A_1 A_0,B_0
every A_0 A_1,B_0

$$\frac{\text{every } A_0\ A_1, B_0}{\text{every } A_1\ A_0, B_0}$$

(every rc trans) Our relative clauses are intersective, and every is transitive, so:

every A_0 A_1, D_0
every A_0 A_1, B_0 every A_0 A_1, C_0 every B_0 C_0, D_0

$$\frac{\text{every } A_0\ A_1, B_0 \qquad \text{every } A_0\ A_1, C_0 \qquad \text{every } B_0\ C_0, D_0}{\text{every } A_0\ A_1, D_0}$$

(9) Notice that although the derivation trees still look like context free derivation trees, and we still have finitely many axioms and rules.

12.4 Metatheory for every A_0 A_1, B_0

(10) (Soundness) If $A \vdash e$ then $A \models e$.

Proof: by induction on length of proof, as before.

(11) (Completeness) If $A \models e$ then $A \vdash e$.

Proof: i. Consider any A and any sentence e = (every A_0 A_1, B_0) where $A \models e$.

ii. Define an interpretation μ for the language over universe $E = \{\text{you}\}$ as follows:
For each noun N $\in Lex$,

$$\mu(\text{N}) = \begin{cases} E & \text{if } A \vdash (\text{every } A_0\ A_1, \text{N}) \\ \emptyset & \text{otherwise.} \end{cases}$$

iii. Notice that if we can show that $\mu(B_0) \neq \emptyset$ for, then the proof is complete.

iv. If (every C_0 C_1, D_0) $\in A$ then $\mu(C_0) \cap \mu(C_1) \subseteq \mu(D_0)$.

Sub-Proof of iv: Assume (every C_0 C_1, D_0) $\in A$ and consider these two cases:

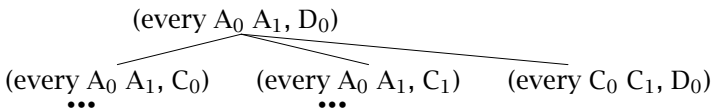
($\mu(C_0) \cap \mu(C_1)) = \emptyset$) In this case, iv holds trivially.

($\mu(C_0) \cap \mu(C_1)) =$ E) so then by def μ, $\mu(C_0) = \mu(C_1) = E$.

a. So in this case, again by the definition of μ,

$$A \vdash (\text{every } A_0\ A_1, C_0),$$
$$A \vdash (\text{every } A_0\ A_1, C_1).$$

b. So there is some derivation of the form:



where the dots ••• represent subtrees whose leaves are Axioms and elements of A.

c. This tree shows that $A \vdash$ (every A_0 A_1, D_0).

d. So then by the definition of μ, $\mu(D_0) = E$.

e. But then it follows that $\mu(C_0) \cap \mu(C_1) \subseteq \mu(D_0)$, and the *Sub-Proof is complete.*

v. By v, μ verifies A. Since $A \models e$, it follows that μ verifies e. That is, $\mu(A_0) \cap \mu(A_1) \subseteq \mu(B_0)$.

vi. Since every set derives (every A_0 A_1 is a A_0), A does, and so by the definition of μ again, $\mu(A_0) = E$ and similarly with one permutation step, $\mu(A_1) = E$. So by v, $\mu(B_0) = E$ too. □

12.5 Inference for the full fragment

(12) The finite subset considered in the previous section has a simple inference relation, but a proper appreciation of the role of relativization is concealed. Here we propose an inference relation for the whole fragment.

(13) *(Axioms: every reflexive)* For any nouns A_0 A_1…A_i, the following is an axiom (any $i \geq 0$):

$$\text{every } A_0\ A_1 \ldots A_i,\ A_0\ A_1 \ldots A_i$$

(14) *(Rules)* We present the inference steps in **linguists' style on left**, **logicians' style on right**.
For any nouns A_0, A_1,…,A_i, and B_0, B_1,…,B_j (any $i, j \geq 0$):

(Permutation) The order of the nouns in the the subject does not matter, and neither does the order of the nouns in the object. So for any permutation π_s of the numbers $\{0, \ldots, i\}$ and any permutation π_o of the numbers $\{0, \ldots, j\}$,

$$\begin{array}{c}\text{every } A_{\pi_s(0)}\ldots A_{\pi_s(i)},B_{\pi_o(0)}\ldots B_{\pi_o(j)}\\ |\\ \text{every } A_0\ldots A_i,B_0\ldots B_j\end{array} \qquad \frac{\text{every } A_0\ldots A_i,B_0\ldots B_j}{\text{every } A_{\pi_s(0)}\ldots A_{\pi_s(i)},B_{\pi_o(0)}\ldots B_{\pi_o(j)}}$$

(every1 add rc) You can add a relative clause on the left "for free":

$$\begin{array}{c}\text{every } A_0\ldots A_i,B_0\ldots B_j\\ |\\ \text{every } A_0\ldots A_{i-1},B_0\ldots B_j\end{array} \qquad \frac{\text{every } A_0\ldots A_{i-1},B_0\ldots B_j}{\text{every } A_0\ldots A_i,B_0\ldots B_j}$$

(every2 rem rc) You can remove a relative clause on the right "for free":

$$\begin{array}{c}\text{every } A_0\ldots A_i,B_0\ldots B_{j-1}\\ |\\ \text{every } A_0\ldots A_i,B_0\ldots B_j\end{array} \qquad \frac{\text{every } A_0\ldots A_i,B_0\ldots B_j}{\text{every } A_0\ldots A_i,B_0\ldots B_{j-1}}$$

(every1 rem rc) You can remove a relative clause from the left with an additional premise:

$$\begin{array}{ccc}&\text{every } A_0\ldots A_{i-1},\ B_0\ldots B_j&\\ \text{every } A_0\ldots A_i,\ B_0\ldots B_j & & \text{every } A_i,A_0\ldots A_{i-1}\end{array} \qquad \frac{\text{every } A_0\ldots A_i,\ B_0\ldots B_j \qquad \text{every } A_i,A_0\ldots A_{i-1}}{\text{every } A_0\ldots A_{i-1},\ B_0\ldots B_j}$$

(every2 add rc) You can add a relative clause to the right with an additional premise:

$$\begin{array}{ccc}&\text{every } A_0\ldots A_i,\ B_0\ldots B_k\ C_0\ldots C_j&\\ \text{every } A_0\ldots A_i,\ B_0\ldots B_k & & \text{every } A_0\ldots A_i,\ C_0\ldots C_j\end{array} \qquad \frac{\text{every } A_0\ldots A_i,\ B_0\ldots B_j \qquad \text{every } A_0\ldots A_i,C_0\ldots C_k}{\text{every } A_0\ldots A_i,\ B_0\ldots B_j\ C_0\ldots C_k}$$

(every) This quantifier is transitive:

$$\begin{array}{ccc}&\text{every } A_0\ldots A_i,\ B_0\ldots B_j&\\ \text{every } A_0\ldots A_i,\ C_0\ldots C_k & & \text{every } C_0\ldots C_k,\ B_0\ldots B_j\end{array} \qquad \frac{\text{every } A_0\ldots A_i,\ C_0\ldots C_k \qquad \text{every } C_0\ldots C_k,\ B_0\ldots B_j}{\text{every } A_0\ldots A_i,\ B_0\ldots B_j}$$

(15) Notice that although the derivation trees still look like context free derivation trees, we now have infinitely many axioms and infinitely many rules.

(16) We can now derive the rules of the previous section and I think that helps us to understand them better!

13 Languages with *some*

13.1 Summary

(1) We have discussed 3 fragments so far: the simple fragment, the *every* fragment, and the *every*+relatives fragment.
We saw that while the syntax and semantics of the *every*+relatives fragment is fairly easy to specify, the reasoning is already rather involved.

(2) Inspired by recent work of Pratt-Hartmann and Moss, we now add a second quantifier, which will complete the stage setting for approaching the problem of learning quantifiers. [1, 2, 3]

13.2 Syntax for a fragment with *some*

G: (we allow any finite number of different nouns like these 5)

snake::N	**some::=N Dq**
leopard::N	a::=N Di
eagle::N	is::=Di V
danger::N	ϵ::=V =Dq I
male::N	

$\mathcal{F} = \{merge, move\}$

13.3 Semantics for the fragment with *some*

$\mu : \Gamma(G) \to (E, 2)$ defined as follows

$\mu(n :: \text{N}) \subseteq E$
$\mu(some :: \text{=N Dq})$ is the function SOME that maps a property P to SOME(P), where
SOME(P) is the function that maps property Q to 1 iff $P \cap Q \neq \emptyset$
$\mu(a :: \text{=N Di}) = \mu(is :: \text{=Di V}) = \mu(\epsilon :: \text{=V =Dq I})$
= the identity function on properties,idP

$$\mu(merge(A, B)) = \begin{cases} \mu(A)(\mu(B)) & \text{if defined} \\ \mu(B)(\mu(A)) & \text{otherwise} \end{cases}$$

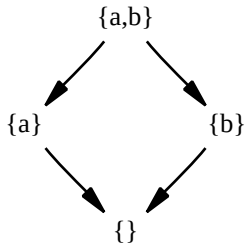
$\mu(move(A)) = \mu(A)$

(3) This semantics interprets every expression in the language, as we can see immediately from the fact that every lexical item is interpreted and every result of applying a structure building rule is interpreted.

13.4 Digression: the semantic perspective on *some* and other quantifiers

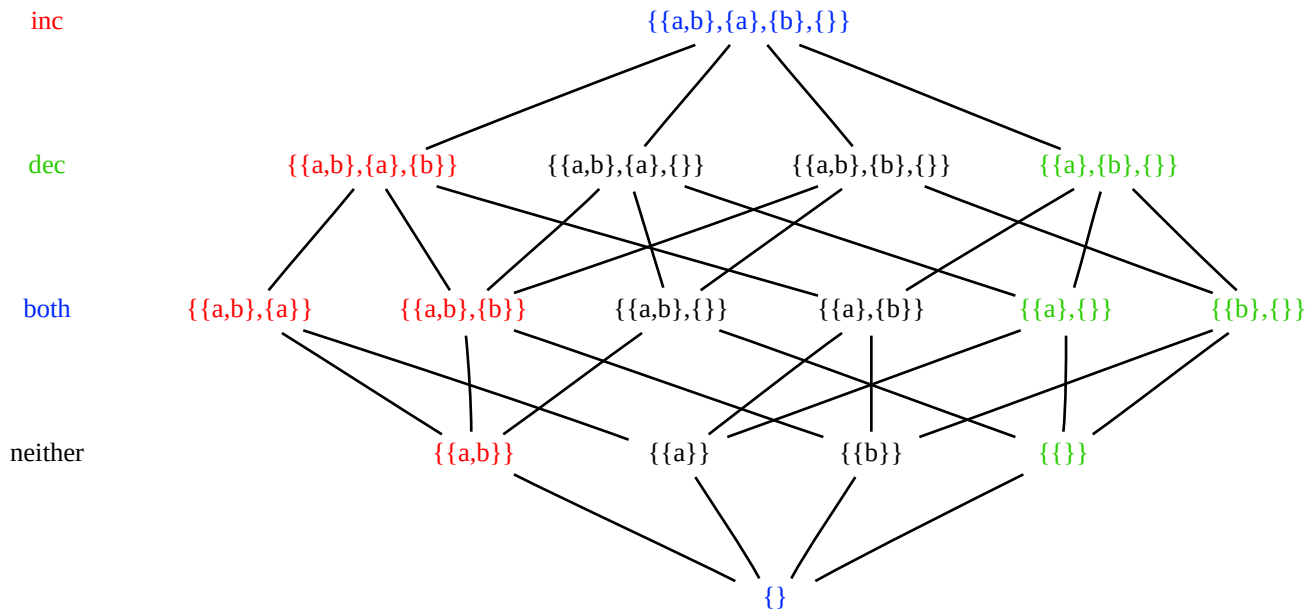
(4) Let's reflect again on the Fregean proposal about quantifiers that we have adopted here. That is, quantifiers are second-order in the sense of being properties or relations among properties or relations among things.

(5) Suppose the universe $E = \{a, b\}$ Then the **properties** are $\wp(E)$. For example, the set of *snakes* might be $\{a\}$, the set of *leopards* might be $\{b\}$, and so the the set of dangers might be E. There are 4 properties altogether, and when ordered by $\subseteq$ we have the following structure:



(6) The **quantifiers**, the **denotations of determiners** are relations between properties. That is, they are $\wp(\wp(E) \times \wp(E))$, so there are $2^{16} = 65,536$ of them. Too many to draw!

(7) The **denotations of determiner phrases** (DPs) like *every leopard* are properties of properties. That is, they are $\wp(\wp(E))$, and so there are $2^4 = 16$ of them, and we can draw them like this:



(8) For example, for any noun X, every(X) denotes the set of properties $\{P|\ X \subseteq P\}$.
Similarly, some(X)=$\{P|\ X \cap P \neq \emptyset\}$, no(X)=$\{P|\ X \cap P = \emptyset\}$, most(X)=$\{P|\ |X \cap P| > |X \cap \overline{P}|\}$, etc.

(9) Think about $E = \mathbb{R}$, the set of real numbers. With this universe, there are very many properties, quantifiers, and DP denotations! Nevertheless, it is easy to understand reasoning like this:

$$\frac{\text{some prime is a real number} \qquad \text{every real number has a reciprocal}}{\text{some prime has a reciprocal}}.$$

Clearly, *recognizing the validity of this reasoning does not require knowing or explicitly thinking about all the things we are talking about!*

Also note that language users do not 'compute' μ! The semantics does not play that kind of role.

13.5 Inference for the fragment with *some*

(10) *(Rules)* We present the inference steps in **linguists' style on left**, **logicians' style on right**. For any nouns A, B,

(some perm) The order of the nouns in the subject and object does not matter,

some B, A
|
some A, B

$$\frac{\text{some A, B}}{\text{some B, A}}$$

(some quasi-reflexive)

some A, A
|
some A, B

$$\frac{\text{some A, B}}{\text{some A, A}}$$

13.6 Metatheory for the fragment with *some*

(11) (Soundness) If $A \vdash e$ then $A \models e$.

Proof: by induction on length of proof, as before.

(12) (Completeness) If $A \models e$ then $A \vdash e$.

Proof: i. Consider any A and any sentence e = (some X, Y) where $A \models e$.

ii. Let the universe E be the collection of sets of sets of one or two nouns,

$$E = \wp(\{N_1, N_2\} \mid N_1, N_2 \in \mathit{expression}(N, G)\}).$$

Now let the meaning assignment be defined as follows: for each noun N_1,

$$\mu(N_1) = \{\{N_1, N_2\} \mid \text{either (some } N_1, N_2) \in A \text{ or (some } N_2, N_1) \in A\}$$

For example, suppose A = {(some leopard, danger)}. Then

$$\begin{aligned} \mu(\text{leopard}) &= \{\{\text{leopard}, \text{danger}\}\} \\ \mu(\text{danger}) &= \{\{\text{danger}, \text{leopard}\}\} \\ \mu(\text{snake}) &= \emptyset \end{aligned}$$

Notice then that

$$\mu(\text{leopard}) = \{\{\text{leopard}, \text{danger}\}\} = \{\{\text{danger}, \text{leopard}\}\} = \mu(\text{danger}).$$

So then μ((some leopard, danger)) = 1 because $\mu(\text{leopard}) \cap \mu(\text{danger}) \neq \emptyset$.

iii. With this definition, μ verifies A, so by hypothesis it verifies e too. That means

$$\mu(X) \cap \mu(Y) \neq \emptyset.$$

iv. a. Suppose X≠Y, that is, X and Y are different nouns. Then by the definition of μ,

$$\mu(X) \cap \mu(Y) = \{\{X, Y\}\}.$$

In this case we know that either (some X, Y)$\in A$ or (some Y, X)$\in A$, and so it follows immediately that $A \vdash e$.

b. Suppose X=Y, so

$$\mu(\text{X}) \cap \mu(\text{Y}) = \mu(\text{X})$$

By iii, we know that this set is non-empty, and so by the definition of μ we know that there is some Y such that either (some X, Y)$\in A$ or (some Y, X)$\in A$. In either case, $A \vdash e$. □

13.7 Syntax for a fragment with *some* and *every*

G: (we allow any finite number of different nouns like these 5)

snake::N	**some::=N Dq**
leopard::N	**every::=N Dq**
eagle::N	a::=N Di
danger::N	is::=Di V
male::N	ϵ::=V =Dq I

$\mathcal{F} = \{merge, move\}$

13.8 Semantics for the fragment with *some* and *every*

$\mu : \Gamma(G) \to (E, 2)$ defined as follows

$\mu(n :: \text{N}) \subseteq E$
$\mu(some :: \text{=N Dq})$ is the function SOME that maps a property P to SOME(P), where SOME(P) is the function that maps property Q to 1 iff $P \cap Q \neq \emptyset$
$\mu(every :: \text{=N Dq})$ is the function EVERY that maps a property P to EVERY(P), where EVERY(P) is the function that maps property Q to 1 iff $P \subseteq Q$
$\mu(a :: \text{=N Di}) = \mu(is :: \text{=Di V}) = \mu(\epsilon :: \text{=V =Dq I})$
= the identity function on properties,idP

$$\mu(merge(A, B)) = \begin{cases} \mu(A)(\mu(B)) & \text{if defined} \\ \mu(B)(\mu(A)) & \text{otherwise} \end{cases}$$

$\mu(move(A)) = \mu(A)$

(13) This semantics interprets every expression in the language, as we can see immediately from the fact that every lexical item is interpreted and every result of applying a structure building rule is interpreted.

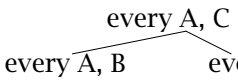
13.9 Inference for the fragment with *some* and *every*

(14) *(Axioms)*

(every reflexive) every A, A

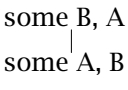
(15) *(Rules)* We present the inference steps in **linguists' style on left**, **logicians' style on right**.

(every trans) every is transitive.



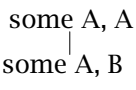
$$\frac{\text{every A, B} \qquad \text{every B, C}}{\text{every A, C}}$$

(some perm) The order of the nouns in the subject and object does not matter,



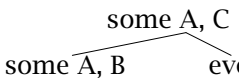
$$\frac{\text{some A, B}}{\text{some B, A}}$$

(some quasi-reflexive)



$$\frac{\text{some A, B}}{\text{some A, A}}$$

(some2 increasing)



$$\frac{\text{some A, B} \qquad \text{every B, C}}{\text{some A, C}}$$

(16) The last rule says that the second argument of *some* is increasing. Notice that we can also prove that the first argument is increasing. (Make sure you see this!)
As discussed in class, the transitivity of *every* shows that it is decreasing in its first argument and increasing in its second argument.

(17) (Soundness) If $A \vdash e$ then $A \models e$.

Proof: by induction on length of proof, as before.

We break the completeness result into two parts. One part for *some* sentences we handle in a Lemma, and then the other part for *every* sentences.

(18) (Lemma) For any set of sentences A, there is a model μ that verifies A, such that for <u>any</u> e = (some X,Y), if μ verifies e then $A \vdash e$.

Proof: i. List the sentences occurring in A containing *some* (e.g. in alphabetical order) like this:

$$(\text{some } V_1, W_1), (\text{some } V_2, W_2), \ldots, (\text{some } V_n, W_n).$$

(Since the set of nouns is finite, we know this list will be finite too.)

ii. Let the universe $E = \{1, \ldots, n\}$ and for each noun Z, define

$$\mu(Z) = \{i | \text{ either } V_i \preceq^* Z \text{ or } W_i \preceq Z\},$$

where as in Lecture 10 $\preceq^*$ is the reflexive transitive closure of the relation

$$n_j \preceq n_k \text{ iff (every } n_j, n_k) \in A.$$

(Sometimes this closure is also written $\preceq$ but I will write $\preceq^*$ to be completely clear.)

Suppose A = {(some leopard,cat),(some snake,danger),(every cat,danger)} and that the language contains only these 4 nouns, *cat, danger, leopard, snake*. Then we have only one non-reflexive instance of $\preceq^*$:

cat $\preceq^*$ cat, <u>cat $\preceq^*$ danger</u>, danger $\preceq^*$ danger, leopard $\preceq^*$ leopard, snake $\preceq^*$ snake

Now suppose we list the *some* sentences from A this way:

some leopard is a cat, some snake is a danger
V_1 W_1 V_2 W_2

Then by the definition of μ given above,

μ(cat) = {1} because cat $\preceq^*$ cat and cat = W_1
μ(leopard) = {1} because leopard $\preceq^*$ leopard and leopard = V_1
μ(danger) = {1, 2} because cat $\preceq^*$ danger, danger $\preceq^*$ danger, cat = W_1, and danger = W_2
μ(snake) = {2} because snake $\preceq^*$ snake and snake = V_2

Notice then that

μ(some leopard, cat) = 1 because μ(leopard) $\cap$ μ(danger) = {1} $\neq \emptyset$.
μ(some snake, danger) = 1 because μ(snake) $\cap$ μ(danger) = {2} $\neq \emptyset$.
μ(every cat, danger) = 1 because μ(cat) $\subseteq$ μ(danger)

iii. For any (every P, Q) $\in A$, we have $P \preceq^* Q$ and so $\mu(P) \subseteq \mu(Q)$, as we saw in (15) of Lecture 10.

iv. For every sentence (some V_i, W_i) in the list above, $i \in \mu(V_i)$ and $i \in \mu(W_i)$, so $\mu(V_i) \cap \mu(W_i) \neq \emptyset$.

v. By iii and iv, μ verifies A. This was the first thing we had to show.

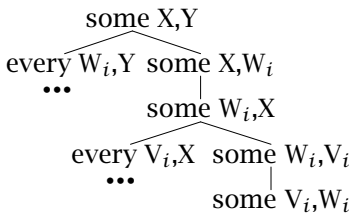
vi. Now suppose e = (some X,Y) and μ verifies e, so $\mu(X) \cap \mu(Y) \neq \emptyset$.

vii. Suppose $i \in \mu(X) \cap \mu(Y)$. Then at least one of the following four cases must hold:

($V_i \preceq^*$ X) and ($W_i \preceq^*$ Y)
($V_i \preceq^*$ Y) and ($W_i \preceq^*$ Y)
($V_i \preceq^*$ X) and ($W_i \preceq^*$ X)
($V_i \preceq^*$ Y) and ($W_i \preceq^*$ X)

Consider each case in turn.

viii. (($\mathbf{V_i \preceq X}$) **and** ($\mathbf{W_i \preceq Y}$)) In this case, since A includes (some V_i, W_i), we have a derivation:



ix. **(the other 3 cases)** are similar. □

(19) (Completeness) If $A \models e$ then $A \vdash e$.

Proof: i. Either e = (some X, Y) or e = (every X, Y). In the first case, the result follows from the (Lemma) just above. So suppose $A \models e$ and e = (every X, Y).

ii. Let A_{every} be all the *every* sentences in A, and consider any model μ_0 that verifies A_{every}.

iii. Now for each noun N, define

$$\mu(\text{N}) = \mu_0(\text{N}) \cup \{\text{you}\}.$$

iv. This model μ verifies all the *some* sentences in A. (In fact it verifies every *some* sentence in the whole language.)

v. So every model μ_0 of A_{every} can be extended to a model μ of A.

vi. Since by assumption $A \models e$, μ is also a model of e.
So, every model μ_0 of A_{every} can be extended to a model μ of e.

vii. Since e is an *every* sentence, the extension of μ_0 to μ is not necessary: μ will verify e.
That is, every model μ_0 of A_{every} is a model of e. $A_{every} \models e$.

viii. By the completeness of the every fragment, since $A_{every} \models e$, we know since $A \vdash_{every} e$.

ix. Since our $\vdash$ for the language of *every* and *some* properly includes all the rules and axioms of the *every* fragment, $A \vdash e$. □

(20) (Lacks canonical model property) There are sets of sentences A such that no model verifies exactly the sentences e such that $A \vdash e$.

Proof: Let $A = \{(\text{every X, Y})\}$ and consider any model μ of this set. If $\mu(Y) = \emptyset$ then μ verifies (every Y, X) which does not follow from (and is not derivable from) A. But if $\mu(Y) \neq \emptyset$ then μ verifies (some Y, Y) which does not follow from (and is not derivable from) A. □

Exercise 5

(1) Consider **at most 1** as in the sentence, *at most 1 leopard is a danger.*

a. How would you define **at most 1**(X)? (analogous to the defs in (8) on page 2)

b. When E={a,b}, which quantifier in the lattice on page 2 is this?

c. Is **at most 1**(X) increasing or decreasing or both or neither?

(2) a. How would you define **exactly 1**(X)?

b. When E={a,b}, which quantifier in the lattice on page 2 is this?

c. Is **exactly 1**(X) increasing or decreasing or both or neither?

(3) Using the inference rules of the *some, every* fragment, show a complete derivation of e = (some reptile,danger) from

$$A = \{(\text{some snake,danger}), (\text{every snake,reptile})\}.$$

(4-optional but recommended!) On page 2 we looked at all 4 properties and all 16 possible DP denotations in a universe containing just two things. Let's say that DP denotation I is a (Boolean) **homomorphism** iff for all properties p, q, (out of the 4 possible)

$$p \cap q \in I \text{ iff } p \in I \wedge q \in I$$
$$p \cup q \in I \text{ iff } p \in I \vee q \in I$$
$$E - p \in I \text{ iff } \neg(p \in I).$$

a. Which of the 16 DP denotations on page 2 are homomorphisms in this sense?
(You can solve this by checking each one, or by thinking more generally…)

b. Is there any intuitive property that all the homomorphisms all have in common?

References for Lecture 13

[1] MOSS, L. Natural language, natural logic, natural deduction. *Forthcoming* (2004). Indiana University.

[2] PRATT-HARTMANN, I. Fragments of language. *Journal of Logic, Language and Information 13*, 2 (2004), 207–223.

[3] PRATT-HARTMANN, I., AND THIRD, A. More fragments of language. *Notre Dame Journal of Formal Logic Forthcoming* (2004).

14 Learning languages with *some,every*

All animals learn. But only human beings create scientific theories, mathematics, literature, moral systems, and complex technology. And only humans have the capacity to acquire such culturally constructed knowledge in the normal course of immersion in the adult world. There are many reasons for the differences between the minds of humans and other animals…bigger brains …language …capacity for causal analysis …Each of these factors doubtless contributes to our prodigious ability to learn. But in my view another factor is even more important: our uniquely human ability to 'bootstrap.' Many psychologists, historians, and philosophers of science have appealed to the metaphor of bootstrapping in order to explain learning of a particularly difficult sort – those cases in which the endpoint of the process transcends in some qualitative way the starting point. The choice of metaphor may seem puzzling – it is self-evidently impossible to pull oneself up by one's own bootstrap.…I keep the term because of its historical credentials and because it seeks to explain cases of learning that many have argued are impossible. Sometimes learning requires the creation of new representational resources that are more powerful than those present at the outset.
Some researchers …debate the existence, even the possibility, of qualitative changes to the child's initial representations. One argument for the impossibility of such radical changes in the course of development is the putative lack of learning mechanisms that could explain them. This is the gap that my appeal to bootstrapping is meant to fill. (Carey 2004)

14.1 Summary

(1) We have discussed 5 fragments so far: the simple fragment, the *every* fragment, and the *every*+relatives fragment, the *some* fragment, and the *some+every* fragment.

Now let's consider the question of how a language with quantifiers like the *some+every* fragment could be learned.

(2) We have already mentioned a puzzle about the human acquisition of *every* or *all*.

Inhelder and Piaget (1959) presented children with displays of colored squares and circles…

Scene: *14 blue circles, 2 blue squares, 3 red squares*
Q: *Are all the circles blue?*
A: *No, there are two blue squares*

Children who give non-adult responses to quantified sentences construe the strong determiner as if it were weak…it is a parsing problem. (Geurts 2003)

How could that be a parsing problem?

(3) We also mentioned a puzzle about the human acquisition of relative clauses: they sometimes interpret relative clauses in sentences like (55) as modifying the subject.

(55) The pig bumps into [the horse that ____ jumps over the giraffe].

Typically, in experiments that evoked nonadult stimuli for (55), repeated here, children were given (besides a pig and a giraffe) only one horse, a situation that makes the use of the relative clause superfluous…Hamburger and Crain made a minor change to their experiments: they gave the children more than one object of the kind referred to by the head of the relative…The outcome was that 5-year-olds gave 95% correct responses, and even children younger than the ones tested in previous experiments (3-year-olds) performed well (69% correct responses)…Therefore, Hamburger and Crain conclude, the errors that children made in previous experiments do not reflect their lack of knowledge of relative clause structure; rather, they reveal their attempt to overcome infelicitous conditions for the use of relative clauses…

Under this view children have the same grammar for relative clauses as adults have. The deviations from the target are a matter of lexical learning… (Guasti 2002, pp227,240)

Today, we will consider the problem of learning a language with quantifiers, leaving the puzzles about relative clauses to later.

14.2 Psychological perspectives (briefly!)

[13, 16, 19] (4) **Misinterpretations.**

4 HORSES, 3 BOYS ON 3 OF THEM

Experimenter: Is every boy riding a horse?

Child: No, not that horse.

- Piaget & Inhelder propose that in children aged 3 or so, quantifiers often scope over both subject and predicate, in some sense. So then

  "Is every boy riding a horse?"

  is like

  "In every case, is a boy riding a horse?"

- Philip and Takahashi propose similarly that *every* is being intepreted adverbially by children around age 3

[15] - There are many studies of quantifier scope. Lidz & Musolino provide evidence that children (4 years old) interpret *every horse didn't jump over the fence* with *every* scoping outside the negation: for every horse, it didn't jump over the fence.

[6, 5, 11] (5) **Hidden competence.** Crain, Thornton, Guasti, et al *take it to be the null hypothesis that children have full linguistic competence* (Crain et al, p.147), and they argue that previous studies are flawed because children do not think both possible answers are 'felicitous', because they think the extra horse should matter.

Crain et al conduct more elaborate studies trying to control for this. Their results suggest that when the situation does not make any particular case stand out as exceptional, 3 year old children perform about as well as adults. They conclude: *"Young children have full grammatical competence with universal quantification."*

On this view, what the previous show is that children accomodate pragmatic infelicities differently from adults.

(6) **Most, etc.** Unlike *some, every, at most 3, less than 100*, the quantifier *most*(A)(B) introduces a more complex kind of relation between A and B, and it is complex in the more technical sense too that it cannot be defined in first order logic. So it is not a surprise that it seems to be acquired later than some, all, one, two, and every. [18]

Studying problems with *most* and other quantifiers in the Childes database (and elsewhere) is difficult because it it difficult to figure out what meaning children intend. Usually takes careful experimental study is required to get hints about the nature of misinterpretations. But we can see in Childes some strange uses of *most* in 4 and 5 year olds:

(Mark 4;5) the most one of the ones upstairs
(Sarah 5;1) How come that's most and that's not?

Stickney also reports this discussion with a 5 year old:

Mother: If you go to bed right now we can read most of *James and the Giant Peach*
Child: And what else are we going to read?
Mother: I didn't say anything about reading anything else. I said, if you go to bed right now we can read most of *James and the Giant Peach*
Child: Yes, you said we could read most of *James and the Giant Peach*, so what else are we going to read?

(7) **A parsing problem?** Geurts notes that the sides in the Misinterpretation vs. Hidden Competence dispute do not explain why children have much more trouble with *every* than with *some*. [9]

Furthermore, we want to explain all the types of error we find. In interpreting *every X is a Y*, children make all three of the following types of errors:

type A error: false when $\mu(X) = \{1,2,3\}$ and $\mu(Y) = \{1,2,3,4\}$

type B error: true when $\mu(X) = \{1,2,3,4\}$ and $\mu(Y) = \{1,2,3\}$

type C error: false when $\mu(X) = \{1,2,3\}$ and $\mu(Y) = \{1,2,3\}$ and $\mu(Y) = \{4\}$

4 HORSES, 3 BOYS ON 3 OF THEM

Experimenter: *Is every boy riding a horse?*

Child type A: *No, not that horse.*

5 CARS, 4 GARAGES WITH 1 CAR EACH

Experimenter: *Are all the cars in garages?* *(Donaldson & Lloyd 1974)*

Child type B: *Yes*

3 CATS EACH HOLDING A BALLOON, 1 MOUSE HOLDING AN UMBRELLA

Experimenter: *Is every cat holding a balloon?* *(Philip & Verrips 1994)*

Child type C: *No. (pointing to the mouse)*

A quantifier is **intersective** if the truth of Q(P)(Q) depends only on $P \cap Q$.[1]

…my proposal is as follows. Children who give non-adult responses to quantified sentences construe the quantifier as if it were [intersective]: the problem lies in the mapping between syntactic form and semantic representation; it is a parsing problem. More accurately, it starts out as a parsing problem, which is patched by means of pragmatic reasoning…

- A child interprets the sentence *some boy is happy* as

[1] Geurts uses the term "weak" instead of "intersective," but I avoid this use of "weak" and "strong" here because it conflicts with a more common use introduced by Barwise & Cooper.

some(<things in this context>,[x: Boy(x) and Happy(x)]

The weak interpretation of *every* is something like this:

every(<things in this context>,[x,y: Boy(x) and Elephant(y) and rides(x,y)])

This is a parsing problem in a sense, since really Boy should be in the first argument, not the second. This problem is "patched" by figuring out which set of cases should go into that position.

If <things in this context> is taken to be the boys ⇒ type-A error

If <things in this context> is taken to be the elephants ⇒ type-B error

If <things in this context> is taken to be the mouse ⇒ type-C error

14.3 Linguistic perspectives (briefly!)

(8) **How many quantifiers, part 1.** In order to bring our particular problem into focus, it is important to remember some of the most basic properties of our *some, every* fragment, and how these compare to human languages.

- We are treating quantifiers like *every* and *some* as binary relations on properties.

  We saw last time that when there are n things in the universe of discourse, there are 2^n different properties that we can represent as distinct sets[2]

  So when there are n thing in the universe of discourse, there are $2^n \times 2^n = 2^{2n}$ pairs of properties, and since a binary relation is an arbitrary subset of these pairs,

  there are $2^{2^{2n}}$ different (binary) quantifiers altogether.

EF: Woah, $2^{2^{2n}}$ grows really fast!

Looking at this with **octave**:

```
>> for i=1:10 vals(1,i)=i;vals(2,i)=2**i;vals(3,i)=2**(2*i);vals(4,i)=2**(2**(2*i)); endfor; vals
vals =

       1       2          3          4        5        6
       2       4          8         16       32       64
       4      16         64        256     1024     4096
      16   65536  1.8447e+19  1.1579e+77      Inf      Inf
```

We can get the actual values in this last row; **Mathematica** provides them by default:

```
In[17]:=
Table[2^i, {i, 6}]

Out[17]=
{2,4,8,16,32,64}

In[16]:=
Table[2^(2*i), {i, 6}]

Out[16]=
{4,16,64,256,1024,4096}

In[15]:=
Table[2^(2^(2*i)), {i, 6}]

Out[15]=
{16,
65536,
18446744073709551616,
115792089237316195423570985008687907853269984665640564039457584007913129639936,
179769313486231590772930519078902473361797697894230657273430081157732675805500\
963132708477322407536021120113879871393357658789768814416622492847430639474120\
437776789342486548527630221960124609411945308295208500576883815068234246288147\
391311054082723716335051068458629823994724593847971630483535632962422413721 6,
104438888141315250669175271071662438257996424904738378038423348328395390797155\
745684882681193499755834089010671443926283798757343818579360726323608785136527\
794595697654370999834036159013438371831442807001185594622637631883939771274567\
233468434458661749680790870580370407128404874011860911446797778359802900668693\
897688178778594690563019026094059957945343282346930302669644305902501597239986\
771421554169383555988529148631823791443449673408781187263949647510018904134900\
```

[2] Obviously, we eventually want to handle the fact that different properties can have the same set represention, like the concepts *featherless biped* and *human*. But for now, the set representation is a convenient approximation.

841706167509366833385055103297208826955076998361636941193301521379682583718809\
183365675122131849284636812555022599830041234478486259567449219461702380650591\
324561082573183538008760862210283427019769820231316901767800667519548507992163\
641937028537512478401490715913545998279051339961155179427110683113409058427288\
427979155484978295432353451706522326906139490598769300212296339568778287894844\
061600741294567491982305057164237715481632138063104590291613692670834285644073\
044789997190178146576347322385026725305989979599609079946920177462481771844986\
745565925017832907047311943316555080756822184657174637329688491281952031745700\
244092666169108741483850784119298045229818573389776481031260859030013024134671 8\
972667321649151113160292078173803343609024380470834040315419033 6}

- **Are all $2^{2^{2n}}$ (binary) quantifiers possible D denotations?** No!

(9) **Monotonicity.** We mentioned already the proposal from Barwise & Cooper which has very few exceptions. Using the term *monotone* to mean increasing or decreasing, and using the term *simple* to mean something like composed of one morpheme, they say: [2]

The denotations of simple [determiner phrases] in natural languages express monotone quantifiers or conjunctions of monotone quantifiers.

(10) **Conservativity: the subject matters.** Let's say that a quantifier like *a suprisingly large number of* is **intensional** in the sense that a set representation cannot suffice. For example, it can be true that [14]

a suprisingly large number of lawyers attended the meeting

and false that

a suprisingly large number of doctors attended the meeting

even when the number of doctors and lawyers attending is the same. This would not be surprising when it's a medical meeting for example.

A quantifier that can be treated as a set in a range of contexts sometimes called *extensional* Keenan & Stavi 1981, 1986 propose: *Extensional determiners in all languages are always interpreted by conservative functions,* where a quantifier Q is **conservative** iff $Q(P)(Q)=Q(P)(P\cap Q)$

So for example:

every student sings ≡ every student is a student who sings
some student sings ≡ some student is a student who sings
…

It is easy to design a quantifier that is not conservative. For example, consider the quantifier q defined by:

$$q(A)(B) \text{ iff } |A\times B|>4.$$

Obviously then, $q(A)(B)\not\equiv q(A)(A\cap B)$, so q is not conservative.

In English, lexical determiner denotations seem to be conservative, as long as we recognize that *only* is not a determiner. Consider the differences between the distribution of *only* and determiners like *some, every, all, one, the, a, most, few, several,…*:

some students sing	*every some students sing	only some students sing
	*some students every sing	some students only sing
not every student sings	*not only student sings	*?not only students sing
some of the students sing	*only of the students sing	*some of only students sing

(11) **Permutation Invariance: particular individuals don't matter.** For all $A, B \subseteq E$ and all permutations π of E,

$$Q_E(A)(B) \text{ iff } Q_E(\pi(A))(\pi(B)).$$

This property, mentioned by van Benthem, is not satisfied by expressions like *John's* or by *everyone except Stabler*, but it is satisified by the things we ordinarily regard as lexical determiners. [20, 21]

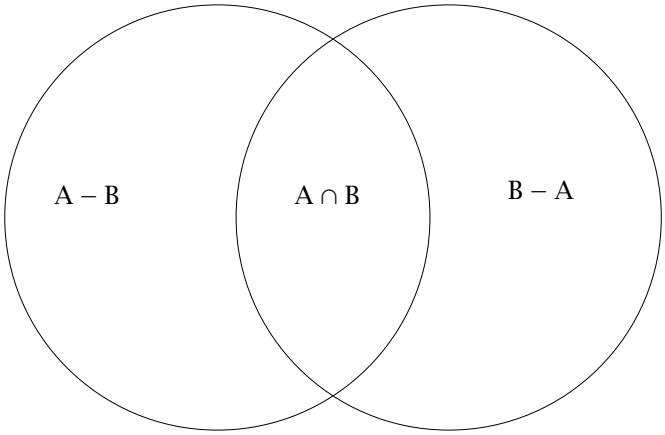
(12) **Extension invariance: size of the universe doesn't matter.** For all $A, B \subseteq E \subseteq E'$,

$$Q_E(A)(B) \text{ iff } Q'_E(A)(B).$$

(13) **Isomophism invariance.** For all but for all E, E' all bijections $\pi : E \to E'$, and all $A, B \subseteq E$,

$$Q_E(A)(B) \text{ iff } Q'_E(\pi(A))(\pi(B)).$$

This captures the irrelevance of individuals and of the size of the universe together, and it holds for most lexical determiners, though here again we need to recognize the difficulties posed by 'pragmatically loaded' quantifiers like few, many, and *a suprisingly large number of*.



(14) **Thm.** (van Benthem) Q is conservative and isomorphism invariant iff the truth of Q(A)(B) depends only on $|A - B|$ and $|A \cap B|$.

Proof: ($\Rightarrow$) Suppose $A, B \subseteq E$, and $A', B' \subseteq E'$, and Q is conservative and isomorphism invariant. It suffices to show that if $|A - B| = |A' - B'|$ and $|A \cap B| = |A' \cap B'|$, then $Q(A)(B) = Q(A')(B')$.
Since $|A - B| = |A' - B'|$ there is a bijection g from $|A - B|$ to $|A' - B'|$, and since $|A \cap B| = |A' \cap B'|$, we can extend this to a bijection that also maps $|A \cap B|$ to $|A' \cap B'|$. The bijection g then maps A to A'.
By conservativity, $Q(A)(B)$ iff $Q(A)(A \cap B)$. Then by isomorphism, $Q(A)(B)$ iff $Q(g[A])(g(A \cap B))$. Then again by conservativity, $Q(A)(B)$ iff $Q(A')(B')$.

($\Leftarrow$) Suppose the truth of $Q(A)(B)$ depends only on $|A - B|$ and $|A \cap B|$. That is, suppose that whenever $|A - B| = |A' - B'|$ and $|A \cap B| = |A' \cap B'|$, we have $Q_E(A)(B) = Q_{E'}(A')(B')$. Then it follows immediately from the definitions that Q is conservative and Q is preserved under isomorphisms. □

Functions which are conservative and isomorphism invariant are of special interest because their truth conditions can be given by simple comparison of set sizes::[3]

[3] Ben Shalom [3] extends the interest of quantifiers with these same properties with the following result:

Theorem 1 If Q is conservative and preserved by isomorphisms, then the modal logic in which □ has the semantics of Q is invariant under isomorphisms and generated submodels.

This result allows us to think of the quantifiers of human languages as modal operators of a standard kind. Cf. [8, 22, 1].

some(A)(B)	$\|A \cap B\| > 0$
no(A)(B)	$\|A \cap B\| = 0$
at least N(A)(B)	$\|A \cap B\| \geq N$
more than N(A)(B)	$\|A \cap B\| > N$
exactly N(A)(B)	$\|A \cap B\| = N$
all but N(A)(B)	$\|A - B\| = N$
between N and M(A)(B)	$N \leq \|A \cap B\| \leq M$
finitely many(A)(B)	$\|A \cap B\| < \aleph_0$

every(A)(B)	$\|A - B\| = 0$
at most N(A)(B)	$\|A \cap B\| \leq N$
most(A)(B)	$\|A - B\| > \|A \cap B\|$
fewer than N(A)(B)	$\|A \cap B\| > N$
the N(A)(B)	$\|A - B\| = 0 \& \|A \cap B\| = N$
the N out of M(A)(B)	$\|A - B\| = \|A \cap B\| \frac{M-N}{M}$
every third(A)(B)	$\|A - B\| = \|A \cap B\| \frac{3-1}{3}$

A technical point: the ones in **bold** cannot be expressed in first order logic, but the others can.

References for Lecture 14

[1] ALECHINA, N. Generalized quantifiers as modal operators. 1993.

[2] BARWISE, J., AND COOPER, R. Generalized quantifiers and natural language. *Linguistics and Philosophy 4* (1981), 159–219.

[3] BEN-SHALOM, D. Natural language, generalized quantifiers and modal logic. In *Proceedings, Amsterdam Colloquium* (1994).

[4] CAREY, S. Bootstrapping and the origins of concepts. *Daedalus Winter* (2004), 59–68.

[5] CRAIN, S., AND THORNTON, R. *Investigations in Universal Grammar: A Guide to Experiments on the Acquisition of Syntax and Semantics*. MIT Press, Cambridge, Massachusetts, 1998.

[6] CRAIN, S., THORNTON, R., BOSTER, C., CONWAY, L., LILLO-MARTIN, D., AND WOODAMS, E. Quantification without qualification. *Language Acquisition 5* (1996), 83–153.

[7] DONALDSON, M., AND LLOYD, P. Sentences and situations: Children's judgments of match and mismatch. In *Problèmes Actuels en Psycholinguistique*, F. Bresson, Ed. Presses Universitaires de France, Paris, 2000.

[8] FINE, K. In so many possible worlds. *Notre Dame Journal of Formal Logic 13* (1974), 516–520.

[9] GEURTS, B. Explaining grammaticalization (the standard way). *Linguistics 38* (2000), 781–788.

[10] GEURTS, B. Quantifying kids. *Language Acquisition 11* (2003), 197–218.

[11] GUASTI, M. T. *Language Acquisition: The Growth of Grammar*. MIT Press, Cambridge, Massachusetts, 2002.

[12] HAMBURGER, H., AND CRAIN, S. Relative acquisition. In *Language Development: Syntax and Semantics*, S. Kuczaj, Ed. Lawrence-Erlbaum, Hillsdale, New Jersey, 1982.

[13] INHELDER, B., AND PIAGET, J. *La Genèse des Structures Logiques Elémentaires: Classifications et Sériations*. Delachaux et Niestlé, Neuchâtel, 1959. English translation, *The Early Growth of Logic in the Child: Classification and Seriation*, London: Routledge and Kegan Paul, 1964.

[14] KEENAN, E. L., AND STAVI, J. A semantic characterization of natural language determiners. *Linguistics and Philosophy 9* (1986), 253–326.

[15] LIDZ, J., AND MUSOLINO, J. Children's command of quantification. *Cognition 84* (2002), 113–154.

[16] PHILIP, W. *Event Quantification in the Acquisition of Universal Quantification*. PhD thesis, University of Massachusetts, 1995.

[17] PHILIP, W., AND VERRIPS, M. Dutch preschoolers *elke*. In *Boston University Conference on Language Development, BUCLD'94* (1994).

[18] STICKNEY, H. Children's acquisition of "most". *Workshop on Quantifier Acquisition, University of Massachusetts* (2003).

[19] TAKAHASHI, M. Children's interpretation of sentences containing *every*. In *Papers on the Acquisition of WH*, T. Maxfield and B. Plunkett, Eds. University of Massachusetts Occasional Papers in Linguistics, 1991.

[20] VAN BENTHEM, J. Questions about quantifiers. *Journal of Symbolic Logic 49* (1984), 443–466.

[21] VAN BENTHEM, J. *Essays in Logical Semantics*. Reidel, Dordrecht, 1986.

[22] VAN DER HOEK, W., AND DE RIJKE, M. Generalized quantifiers and modal logic. In *Generalized Quantifier Theory and Applications*, J. van der Does and J. van Eijck, Eds. Dutch Network for Language, Logic and Information, Amsterdam, 1991.

15 Learning languages with *some,every*, part 2

> *There is a structural definition of "automaton" as anything that can be made by joining together simple elements. Nets remember by circulating information in loops, and so the essential criterion of simplicity has to do with loop structure. The simplest natural subvariety is the net with no loops, which turns out to be trivial. Slightly up in complexity is to allow only one very simple loop: the output of a unit is allowed to go back into its own input like a "hold" circuit on a relay. [these are "locally testable"]*
> *A combinational switch, the reader may recall, is a device whose output at any time is a Boolean function of the input conditions at that time; in other words, a device that has no memory at all. Now the machine as a whole for a locally testable event must have a memory because it must determine [whether a local event occurs at the beginning, end or anywhere in the middle of a sequence].* (McNaughton and Papert 1971, pp.xvii, 16)

15.1 Summary

(1) the syntax of the *some+every* fragment can be parsed by any of the standard MG parsing methods.

- the question of whether some assumptions A entail a given sentence is easily computable.
- We are now considering how a *some+every* fragment could be learned. This problem is particularly interesting since it seems to require some success on an earlier stage of learning: the identification of at least some noun meanings. This could begin a "bootstrapping" process…

(2) **Psychological perspectives.** Children have much more trouble with *every* than with *some*.

- **misinterpretation.** Piaget et al: some of these errors come from misinterpreting *every*
- **full competence.** Crain, Guasti, et al: children are innately equipped to understand "universal quantification," and the errors we see come pragmatic or discourse factors.
- **"parsing" errors.** Geurts: the problems come from a failure to parse apart the arguments of the quantifier, instead letting the first argument of a quantifier be always given by context. This predicts the relative lack of errors with *some*, and allows different pragmatic assumptions about the "context" to give the different kinds of errors with *every*

(3) **Linguistic perspectives.**

Syntactically, we saw that *only* is not a determiner, but involves some kind of focusing, and so it can appear in many positions where determiners are impossible.

We should have also noted that although *every* and *some* have the exactly the same distribution in the language of the fragment, they do **not** have the same distribution in English and most other human languages:

a. Every/*some student with any sense went to the concert.

b. Some/*every of the students didn't go.

c. There is some/*every student in the hall

d. You didn't see some student (cannot mean "It is not the case that there is a student X s.t. you saw X"

e. You didn't see every student (can mean "It is not the case that for every student X, you saw X"

These facts are interesting because they provide a possible way to support a **full competence** theory: if universal quantifiers have a distinctive part of speech, then a learner could learn what *every* means just by figuring out its syntax. (We'll return to this idea again later)

Semantically, We saw that with few exceptions, determiner denotations are **conservative**

that is: Q(A)(B)≡Q(A)(A∩B)

and **invariant under isomorphisms** $\pi : E \to E'$

that is: $Q_E(A)(B) \equiv Q'_E(\pi A)(\pi B)$.

Van Benthem shows that this has the important consequence

Thm. Q is conservative and isomorphism invariant iff the truth of $Q_A B$ depends only on $|A - B|$ and $|A \cap B|$.

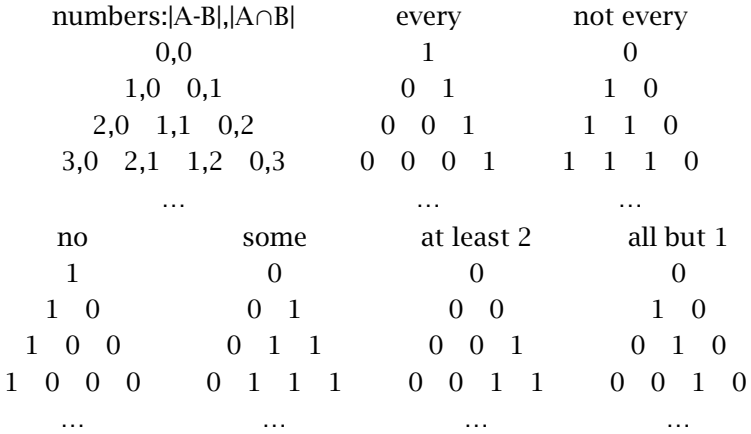
some(A)(B)	$\|A \cap B\| > 0$	every(A)(B)	$\|A - B\| = 0$
no(A)(B)	$\|A \cap B\| = 0$	at most N(A)(B)	$\|A \cap B\| \leq N$
at least N(A)(B)	$\|A \cap B\| \geq N$	**most**(A)(B)	$\|A - B\| > \|A \cap B\|$
more than N(A)(B)	$\|A \cap B\| > N$	fewer than N(A)(B)	$\|A \cap B\| > N$
exactly N(A)(B)	$\|A \cap B\| = N$	the N(A)(B)	$\|A - B\| = 0 \& \|A \cap B\| = N$
all but N(A)(B)	$\|A - B\| = N$	the N out of M(A)(B)	$\|A - B\| = \|A \cap B\| \frac{M-N}{M}$
between N and M(A)(B)	$N \leq \|A \cap B\| \leq M$	**every third**(A)(B)	$\|A - B\| = \|A \cap B\| \frac{3-1}{3}$
finitely many(A)(B)	$\|A \cap B\| < \aleph_0$		

A technical point: the ones in **bold** cannot be expressed in first order logic, but the others can.

Now the plot thickens...

15.2 Linguistic perspectives (cont'd)

(4) van Benthem notes that we can get a geometric picture of each quantifier with a **tree of numbers** that specifies for each pair (|A-B|,|A∩B|) whether it is in the quantifier or not:



most	**all but an even number**
0	1
0 0	0 1
0 0 1	1 0 1
0 0 1 1	0 1 0 1
0 0 0 1 1	1 0 1 0 1
…	…

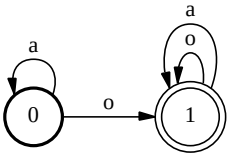
(5) Van Benthem also notes that the arithmetic relations defined by the first order quantifiers Q have an automata-theoretic characterization.

Letting any element of A-B be a and any element of A∩B be o (for "overlap"), a finite model where Q(A)(B) holds can be regarded as a (finite) set of sentences that list (A-B) and (A∩B) in any order. Then the set of all such sentences, for any situations where Q(A)(B) holds is a finite state language:

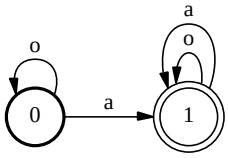
every=o^*:

no=a^*:

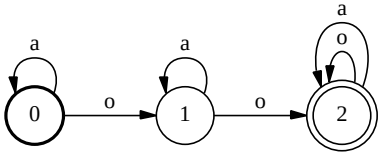
some=$a^*o\{a,o\}^*$:



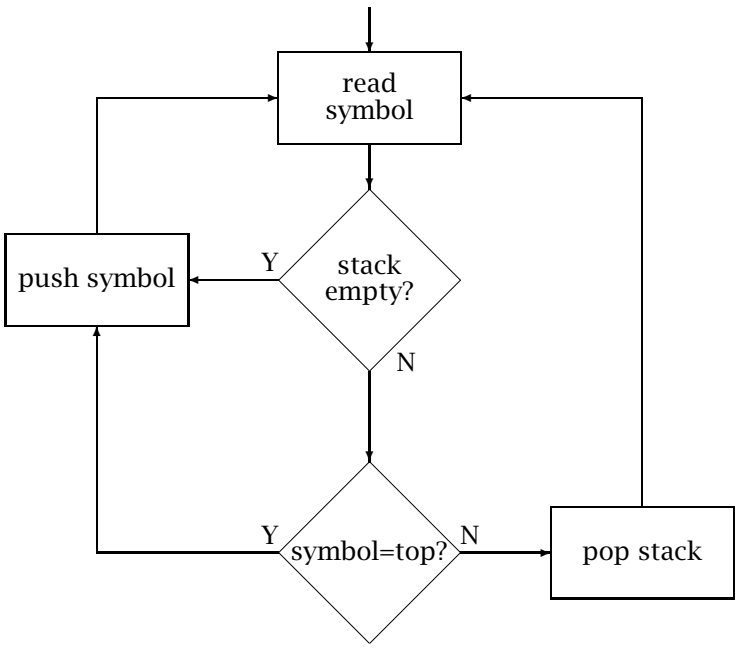
not every=$o^*a\{a,o\}^*$:



at least 2=$a^*oa^*o\{a,o\}^*$:



Note that we are showing here the minimum sized deterministic, "canonical" acceptors for these languages. Non-first order quantifiers require more complex acceptors. The *most* a-o language is accepted by this pushdown automaton:



We focus on first order quantifiers first.

(6) Following McNaughton & Papert [9], say a regular language L is **testable** iff the canonical acceptor for L contains no loops with more than one state. (see the inspirational quote on p.1)

All the a-o languages for first order quantifiers shown above are testable in this sense. They have another important property too:

(7) We can extend any permutation π of an alphabet Σ to a permutation of the set of strings Σ^* in the standard way, namely:

$$\pi(\epsilon) = \epsilon$$
$$\pi(as) = \pi(a)\pi(s) \qquad \text{for any } a \in \Sigma, s \in \Sigma^*$$

(8) Let's say a set of strings $L \subseteq \Sigma^*$ is **permutation invariant** iff for every permutation π of the alphabet Σ and every string s, $s \in L$ iff $\pi(s) \in L$

[12, 13, 14] (9) **Thm:** (van Benthem) The a-o language of every first order definable quantifier is accepted by a testable, permutation-invariant finite state automaton.

[10, 1, 2, 3, 8] (10) **Most etc.** Mostowski notices that the quantifier *at most finitely many* cannot be expressed in first order logic, and that any logic with this quantifier cannot have a complete inference method. Barwise & Cooper sketch a proof that the meaning of *most* cannot be expressed in first order logic. And Boolos notices that some complex constructions like

for every drop of rain, a flower grows
for every A, there is a B

when interpreted literally, mean that there is a kind of 1-1 pairing of the elements of A with the elements of B, and this is also not expressible in first order logic.

[12, 13, 14, 6] (11) **Thm:** (van Benthem;Ginsburg&Spanier) The a-o languages that can be computed by pushdown automata are the semilinear sets.

[As already noted, we will not worry about non-first-order quantifiers for now: our current fragment contains only first order quantifiers, and we know that children acquire the simple first order quantifiers first…]

15.3 Learning the syntax of the fragment

G: (we allow any finite number of different nouns)

snake::N	some::=N Dq
leopard::N	every::=N Dq
eagle::N	a::=N Di
danger::N	is::=Di V
male::N	ϵ::=V =Dq I

$\mathcal{F} = \{merge, move\}$

Since the language is finite, it can be learned in the technical sense by a device that simply remembers every example ever heard. We can do better, but let's postpone careful consideration of this problem for the moment…

15.4 Learning the semantics of the fragment

(12) We can think of this problem as identifying certain subsets of elements in the 'triangle of numbers', or equivalently as the problem of identifying a testable, permutation invariant finite state language. [4, 11, 5]

(13) Notice that if we stretch our English to allow arbitrary boolean combinations of first order determiners. Then we can denote the single point (0,0) in the tree of numbers with:

(every and no)(A)(B).

We can denote (0,1) with

(not every and all but 1)(A)(B).

We can denote (1,0) with

(every and exactly 1)(A)(B).

Furthermore, we can denote any finite combination of these points using disjunctions. For example, to denote the points {(0,0),(0,1)}:

(either (every and no) or (not every and all but 1))(A)(B).

So we can denote every finite subset of the tree of numbers, and we can also denote some infinite sets as we saw above. As Tiede notices: this is bad news for the learner…

(14) **Thm.** (Gold) Exact identification in the limit is impossible for a class containing all finite sets and one or more infinite sets. [7]

(15) **Possible responses:**

a. We can agree that we cannot learn all first order quantifiers, if we can find a subset of them that we can learn, in terms of which the others could be defined!

b. We have not made any use of inferential relations, which might also provide evidence about the meaning of the quantifiers.

c. In our fragment, the 2 quantifiers have the same syntax. But in human languages, the different quantifiers (esp. analogs of *every* and *some*) often have different syntactic roles, so maybe there could even be syntactic evidence about the meanings of quantifiers.

More on these next week.

References for Lecture 15

[1] BARWISE, J., AND COOPER, R. Generalized quantifiers and natural language. *Linguistics and Philosophy 4* (1981), 159–219.

[2] BOOLOS, G. For every A there is a B. *Linguistic Inquiry 12* (1981), 465–467.

[3] BOOLOS, G. Nonfirstorderizablility again. *Linguistic Inquiry 15* (1984), 343.

[4] CLARK, R. Learning first order quantifier denotations: An essay in semantic learnability. Tech. rep., Institute for Research in Cognitive Science, University of Pennsylvania, 1996.

[5] COSTA FLORÊNCIO, C. Learning generalized quantifiers. In *Proceedings of the Seventh ESSLLI Student Session* (2002).

[6] GINSBURG, S., AND SPANIER, E. H. Semigroups, Presburger formulas, and languages. *Pacific Journal of Mathematics 16* (1966), 285–296.

[7] GOLD, E. M. Language identification in the limit. *Information and Control 10* (1967), 447–474.

[8] HINTIKKA, J. Quantifiers vs. quantification theory. *Linguistic Inquiry 5* (1984), 153–178.

[9] MCNAUGHTON, R., AND PAPERT, S. *Counter-Free Automata.* MIT Press, Cambridge, Massachusetts, 1971.

[10] MOSTOWSKI, A. On a generalization of quantifiers. *Fundamenta Mathematicae 44* (1957), 12–36.

[11] TIEDE, H.-J. Identifiability in the limit of context-free generalized quantifiers. *Journal of Language and Computation 1* (1999).

[12] VAN BENTHEM, J. *Essays in Logical Semantics.* Reidel, Dordrecht, 1986.

[13] VAN BENTHEM, J. Semantic automata. In *Studies in Discourse Representation Theory and the Theory of Generalized Quantifiers*, J. Groenendijk, D. de Jongh, and M. Stokhof, Eds. Foris, Dordrecht, 1987.

[14] VAN BENTHEM, J. Towards a computational semantics. In *Generalized Quantifiers: Linguistic and Logical Approaches*, P. Gärdenfors, Ed. Reidel, Boston, 1987.

16 Learning quantifiers

16.1 the big picture.

(1) Following Montague and many others: Human languages have the 3 parts of a logic:

- a language (a set of expressions, sequences of gestures)
- a semantics μ assigns meanings to expressions, defining semantic entailment $\models$
- a inference relation on expressions, defining syntactic entailment $\vdash$

(2) (parts of) the language and inferences are recognized (computed!) by the language user.
μ is not 'computed' but semantic complexity is reflected in the inference relation, the 'reasoning'.

(3) Inspired by Pratt-Hartmann and Moss and others we consider very simple logics first: [20, 26, 27]

1	Simple (snake! leopard!)	identification and segmentation (from examples, by entropy peaks) identification of semantic values (from examples and informants) ⇒ we learn simple meaning components first
2	N+every	syntax: "every N is a N" semantics: quantifiers as second order concepts inference: sound and complete: $A \vdash e$ iff $A \models e$ entailments decidable in linear time allows "canonical models"
3	N+every+RC	syntax: "minimalist grammar" semantics: intersective interpretation of RC modifiers inference: sound and complete, efficiently decidable
4	N+some	syntax: "some N is a N" inference: sound and complete, efficiently decidable
5	N+some+every	syntax: "{some,every} N is a N" inference: sound and complete, efficiently decidable no "canonical models" identification of semantic values: quantifiers
6	N+some+every+no+not$_v$+names	inference: sound and complete, efficiently decidable
6a	+V$_{trans}$+V$_{ditrans}$	"
7	+RC	inference: sound and complete, but not efficiently decidable
8	+at least as many	no completeness result
9	+RC+anaphora	undecidable

- syntax and semantics are simple compared to inference
  ⇒ constructions learned (and used) piecemeal, from familiar, well-behaved parts of the language
- learning problems are challenging, even in fragment 1!
  ⇒ simplified when syntax, semantics and inference are not independent

When they (my elders) named some object, and accordingly moved towards something, I saw this and I grasped that the thing was called by the sound they uttered when they meant to point it out. Their intention was shown by their bodily movements, as it were the natural language of all peoples: the expression of the face, the play of the eyes, the movement of the other parts of the body, and the tone of the voice which expresses our state of mind in seeking, having, rejecting or avoiding something. Thus, as I heard words repeatedly used in their proper places in various sentences, I gradually learned to understand what objects they signified; and after I had trained my mouth to form these signs, I used them to express my own desires. (Augustine 398)

…language learning is possible only because infants selectively attune to certain properties of the language input and because they use a coalition of cues available in the input to help them "crack" the syntactic code. (Hirsh-Pasek & Golinkoff 1996)

…from a very early stage, the learner's comprehension machinery can best be characterized as a perceptual guessing game in which multiple probabilistic cues are used to converge on the grammatical operations that gave rise to the sentence… (Trueswell & Gleitman 2004)

16.2 Summary: N+some+every.

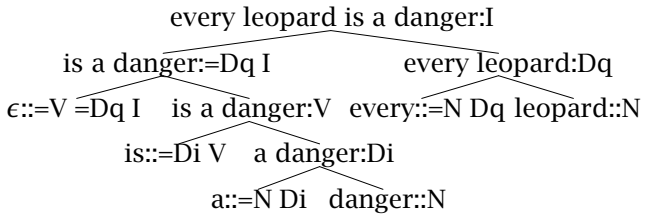
syntax G: a lexicon Lex of 10 elements (but we'll allow any finite number of different nouns)

snake::N	some::=N Dq
leopard::N	every::=N Dq
eagle::N	a::=N Di
danger::N	is::=Di V
male::N	ϵ::=V =Dq I

structure building rules $\mathcal{F} = \{merge, move\}$, fixed for all grammars.

$L(G) = closure(Lex, \mathcal{F})$ (everything you can build from the lexicon with the rules)

$\Gamma(G)$ the set of all the derivations of every element of $L(G)$



semantics: any $\mu : \Gamma(G) \to (E, 2)$ defined as follows (N denotations can vary, all else fixed):

$\mu(n :: \text{N}) \subseteq E$

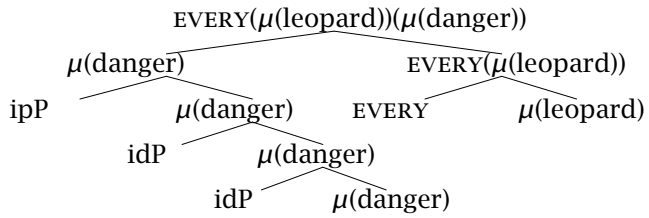
$\mu(some :: \text{=N Dq})$ is the function SOME that maps a property P to SOME(P), where SOME(P) is the function that maps property Q to 1 iff $P \cap Q \neq \emptyset$

$\mu(every :: \text{=N Dq})$ is the function EVERY that maps a property P to EVERY(P), where EVERY(P) is the function that maps property Q to 1 iff $P \subseteq Q$

$\mu(a :: \text{=N Di}) = \mu(is :: \text{=Di V}) = \mu(\epsilon :: \text{=V =Dq I})$
= the identity function on properties,idP

$$\mu(merge(A, B)) = \begin{cases} \mu(A)(\mu(B)) & \text{if defined} \\ \mu(B)(\mu(A)) & \text{otherwise} \end{cases}$$

$\mu(move(A)) = \mu(A)$



inference: (rules and proofs displayed linguists' style, root up)

(every reflexive) every(A)(A)

(every trans) every is transitive.

every(A)(C)
every(A)(B) every(B)(C)

(some perm) The order of the nouns in the subject and object does not matter,

some(B)(A)
some(A)(B)

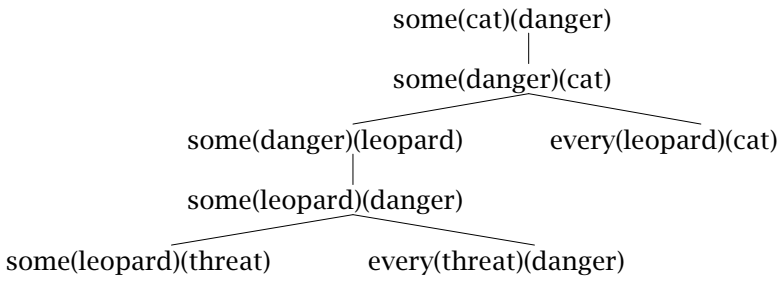
(some quasi-reflexive)

some(A)(A)
some(A)(B)

(some2 increasing)

some(A)(C)
some(A)(B) every(B)(C)

(4) These inference rules get the increasingness of both arguments of *some* via the permutation rule, as we see in proofs like this:



(5) First results:

- This fragment is sound and complete: $A \vdash e$ iff $A \models e$,
- **the** *every+some* **languages are efficiently parsable**, and
- **entailments can be efficiently decided**.
- Now: **learning** (with attention to methods that could generalize to more quantifiers)
  This problem is particularly interesting since it seems to require some success on an earlier stage of learning: the identification of at least some noun meanings. A "bootstrapping" process begins…

(6) **Thm.** Q is conservative and isomorphism invariant iff the truth of $Q(A)(B)$ depends only on $|A-B|$ and $|A \cap B|$.

some(A)(B)	$\|A \cap B\| > 0$	every(A)(B)	$\|A - B\| = 0$
no(A)(B)	$\|A \cap B\| = 0$	at most N(A)(B)	$\|A \cap B\| \leq N$
at least N(A)(B)	$\|A \cap B\| \geq N$	**most**(A)(B)	$\|A - B\| > \|A \cap B\|$
more than N(A)(B)	$\|A \cap B\| > N$	fewer than N(A)(B)	$\|A \cap B\| > N$
exactly N(A)(B)	$\|A \cap B\| = N$	the N(A)(B)	$\|A - B\| = 0 \& \|A \cap B\| = N$
all but N(A)(B)	$\|A - B\| = N$	the N out of M(A)(B)	$\|A - B\| = \|A \cap B\| \frac{M-N}{M}$
between N and M(A)(B)	$N \leq \|A \cap B\| \leq M$	**every third**(A)(B)	$\|A - B\| = \|A \cap B\| \frac{3-1}{3}$
finitely many(A)(B)	$\|A \cap B\| < \aleph_0$		

(The ones in **bold** cannot be expressed in first order logic, and are typically acquired later.)

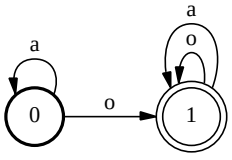
numbers:|A-B|,|A∩B|	every	some	not every
0,0	1	0	0
1,0 0,1	0 1	0 1	1 0
2,0 1,1 0,2	0 0 1	0 1 1	1 1 0
3,0 2,1 1,2 0,3	0 0 0 1	0 1 1 1	1 1 1 0
…	…	…	…
no	**at least 2**	**all but 1**	**exactly 1**
1	0	0	0
1 0	0 0	1 0	0 1
1 0 0	0 0 1	0 1 0	0 1 0
1 0 0 0	0 0 1 1	0 0 1 0	0 1 0 0
…	…	…	…

(7) Letting any element of A-B be a and any element of A∩B be o (for "overlap"), a finite model where Q(A)(B) holds can be regarded as a (finite) set of sentences that list (A-B) and (A∩B) in any order. Then the set of all such sentences, for any situations where Q(A)(B) holds is a finite state language:

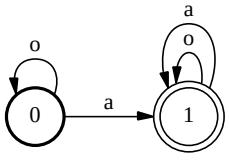
every=o^*:

no=a^*:

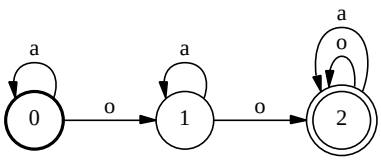
some=$a^*o\{a,o\}^*$:



not every=$o^*a\{a,o\}^*$:



at least 2=$a^*oa^*o\{a,o\}^*$:



(8) **Thm:** (van Benthem) The a-o language of every first order definable quantifier is accepted by a testable, permutation-invariant finite state automaton. [33, 34, 35]

(9) Notice that if we stretch our English to allow arbitrary boolean combinations of first order determiners, then we can denote the language $\{o\}$ with

(not every and all but 1)(A)(B).

We can denote the language $\{a\}$ with

(every and exactly 1)(A)(B).

Furthermore, we can denote any finite combination of these points using disjunctions. For example, to denote the points $\{a, o\}$:

(either (not every and all but 1) or (every and exactly 1))(A)(B)

So there are first order quantifiers corresponding to every finite language and also some infinite ones. As Tiede notices: this is bad news for the learner…

(10) **Thm.** (Gold) Exact identification in the limit is impossible for a class containing all finite sets and one or more infinite sets. [12]

So if the possible quantifier denotations includes the whole class of "first order" quantifiers, that class is **not learnable** from examples.

(Angluin and Pitt show that this problem remains for "measure 1" probabilistic identification, and it is easy to show that this class is not probably approximately (PAC) learnable either.[1]) [25, 22] [3, 17]

(11) **Possible responses:**

a. **we use syntactic evidence about the meanings of quantifiers.** In our fragment, the 2 quantifiers have the same syntax. But in human languages, *every* and *some* and their closest translation equivalents often have different syntactic roles. Furthermore, determiners have characteristic positions across languages: [13]

G20. When any or all of the items (demonstrative, numeral, and descriptive adjective) precede the noun, they are always found in that order. If they follow, the order is either the same or its exact opposite.

We find an extreme version of this kind of view in Crain, Guasti, et al, and moderate versions in proposals from Gleitman, Pinker, Brent, and others. [9, 14]
(But AFAIK no account of learning lexical quantifier meanings from this tradition yet)

b. **language learners get 'informant' data as well as examples**

Clark considers an extreme version of this view, with 'equivalence queries': [7]

…a minimally adequate teacher answers two types of queries on the part of the learner. The first type is a membership query *where the teacher responds* yes *or* no *depending on whether a particular string is a member of the target set or not…A second type of query consists of a* conjecture*…The answer is* yes *if the description is [correct, and otherwise a counterexample to the conjecture is presented].*
It is not obvious that there is a real world correlate to a conjecture …Brown & Hanlon (1970)…and Brown (1973)…report that adults tend to pay little attention to syntactic ill-formedness of children's utterances; they react instead to the truth value of the proposition which they suppose that the child intended to assert. This is somewhat like a conjecture…[but] the child does not produce a full-blown description of his or her hypothesis set and present it to the parents for their approval.

c. **we learn just a proper subset of the first order quantifiers from examples,** in terms of which the others could be defined.

[31, 8] Tiede and Costa seem to take this view…

d. **inferential relations provide evidence about the meaning of the quantifiers**: the "Gentzen-Hertz" idea.

[10, 28] [16] Logicians Gentzen, Hertz, Geach and some other philosophers seem to suggest this… Also 'coalition' acquisition theories of the sort found in Hirsh-Pasek, Golinkoff et al. (We'll explore this kind of proposal for the next fragment.)

16.3 Response c: learnable subsets of quantifiers

[2] (12) This is probably the most natural response to the learning problem, and maybe Barwise & Cooper set the stage:

> *The denotations of simple [determiner phrases] in natural languages express monotone quantifiers or conjunctions of monotone quantifiers.*

But Tiede notes that this does not immediately work…

(13) Notice that many a-o languages of increasing quantifiers are subsets of one another, for example:

$$\ldots\text{at least } 3 \subset \text{at least } 2 \subset \text{at least } 1 \subset \text{some} \subset \ldots$$

(14) **Thm.** (Tiede) The first order quantifiers that are increasing in their second argument are not identifiable from examples.

Proof: Suppose a class of languages $\mathcal{L}$ has a language L such that for every finite $X \subseteq L$ there is another $L' \in \mathcal{L}$ such that $X \subseteq L' \subset L$. Angluin shows that in this case $\mathcal{L}$ cannot be identified from examples.

Tiede shows that the language of the quantifier *some*, interpreted as the set of pairs $(|A-B|, |A \cap B|)$, has this property, as follows. Take any finite subset S of the *some* language, and define

$$i(S) = \{(a-k, b+k) \mid (a,b) \in S,\ 0 \leq k \leq a\}.$$

The set $i(S)$ is finite, increasing, and first order definable (since every finite set is). □

(15) A collection of languages has **finite thickness** iff for every sentence s, the subcollection of languages containing s is finite

(16) **Thm.** (Angluin) A collection of languages with finite thickness is learnable from examples.

Proof idea: Consider the learner that given any set of examples, guesses a smallest language consistent with those examples. (If there is more than one, an arbitrary choice will suffice.) Finite thickness guarantees that this learner will converge after finitely many steps. □

> *…a Venn diagram made of plywood. If over each point there is only a finite stack of ovals (languages), then the family has finite thickness…if the hypothesis space has finite thickness then there are only a finite number of possible conjectures to explain any finite string.* (Wright 1989)

(17) Obviously, every finite set has finite thickness, and so in particular, the collection of 2 languages {every,some} does. Tiede points out that we can construct arbitrarily large sets with finite thickness.

[1] This follows from the infinite "VC dimension" of the class.

16.4 Assembling a first learner

(18) We have already considered learners for a language of nouns or simple propositions.

The learner looks for the conditions in which each expression applies, roughly as Augustine suggested

(19) So we could consider a first learner roughly like this:

1. Learn the nouns: from vectors of literals that hold in each situation, determine conditions for application of each noun

   NB: applied to the non-nouns in the fragment → $\varnothing$ (no literals are true in all applications)

2. Learn the determiners from a class with finite thickness by associating with each non-noun a smallest determiner language consistent with the data.

   NB: applied to the non-determiners in the fragment → $\{a, o\}^*$ (all points (A-B,A∩B))

(20) Assume we have an enumeration of the a-o finite state languages of a class with finite-thickness, such that if L_i precedes L_j, then $L_j \not\subset L_i$. Let's use the notation $\phi[Q \rightarrow L]$ for the function which is like ϕ except that $\phi(Q) = L$. Then if we know which word is the determiner, step 2 could be:

Input: a length j text t of (sentence-situation) pairs where applicability of nouns assessible
Output: A 'smallest' determiner meanings consistent with the data
ϕ_0 := the empty function, everywhere undefined
for $i = 1$ to j **do**
Select $(S_i, M_i) \in t$ where S_i=Q A is a B

$$\phi_i := \begin{cases} \phi_{i-1} & \text{if in M, (A-B,A}\cap\text{B)} \in \phi_{i-1}(Q) \\ \phi_{i-1}[Q \rightarrow \text{next language containing (A-B,A}\cap\text{B)}] & \text{otherwise} \end{cases}$$

end for

Algorithm Finite Thickness DET

(21) **Problems with this approach:**

a. The learners considered early do not know when they have reached the right hypothesis

- calculate 'confidences' and specify a threshhold
- (Siskind) calculate 'upper' and 'lower' bounds, and assume correct when they meet

  subtract from upper bound = (possibly relevant) concepts that are salient in any situations where noun produced

  add to lower bound = concepts that must be involved in definition of meaning

b. The learning strategy only works in finite situations, where the $a - o$ sentences are finite!

(22) **Open:** what finite subset would provide a reasonable psychological model?

(23) **Alternatives to step 2:**

- use syntactic constraints to identify potential determiners instead of considering all non-nouns
- pay attention to inference patterns in discourse, which is possible even when sets ∞.
  (More on this Gentzen-Hertz idea for the next fragment!)

References for Lecture 16

[1] Angluin, D. Inductive inference of formal languages from positive data. *Information and Control 45* (1980), 117–135.

[2] Barwise, J., and Cooper, R. Generalized quantifiers and natural language. *Linguistics and Philosophy 4* (1981), 159–219.

[3] Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery 36* (1989), 929–965.

[4] Brent, M. R. Surface cues and robust inference as a basis for the early acquisition of subcategorization frames. In *The Acquisition of the Lexicon*, L. Gleitman and B. Landau, Eds. MIT Press, Cambridge, Massachusetts, 1994, pp. 433–470.

[5] Brown, R. *A First Language*. Harvard University Press, Cambridge, Massachusetts, 1973.

[6] Brown, R., and Hanlon, C. Derivational complexity and order of acquisition in child speech. In *Cognition and the Development of Language*, J. Hayes, Ed. Wiley, NY, 1970.

[7] Clark, R. Learning first order quantifier denotations: An essay in semantic learnability. Tech. rep., Institute for Research in Cognitive Science, University of Pennsylvania, 1996.

[8] Costa Florêncio, C. Learning generalized quantifiers. In *Proceedings of the Seventh ESSLLI Student Session* (2002).

[9] Crain, S., Thornton, R., Boster, C., Conway, L., Lillo-Martin, D., and Woodams, E. Quantification without qualification. *Language Acquisition 5* (1996), 83–153.

[10] Geach, P. *Reason and Argument*. University of California Press, Los Angeles, 1976.

[11] Gleitman, L. The structural sources of verb meaning. *Language Acquisition 1* (1990), 3–55.

[12] Gold, E. M. Language identification in the limit. *Information and Control 10* (1967), 447–474.

[13] Greenberg, J. Some universals of grammar with particular reference to the order of meaningful elements. In *Universals of Human Language*, J. Greenberg, Ed. Stanford University Press, Stanford, California, 1978.

[14] Guasti, M. T. *Language Acquisition: The Growth of Grammar*. MIT Press, Cambridge, Massachusetts, 2002.

[15] Harkema, H. *Parsing Minimalist Languages*. PhD thesis, University of California, Los Angeles, 2001.

[16] Hirsh-Pasek, K., and Golinkoff, R. M. *The Origins of Grammar: Evidence from Early Language Comprehension*. MIT Press, Cambridge, Massachusetts, 1996.

[17] Kearns, M., and Valiant, L. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the Association for Computing Machinery 41*, 1 (1994), 67–95.

[18] Lidz, J., Gleitman, H., and Gleitman, L. R. Kidz in the 'hood: Syntactic bootstrapping and the mental lexicon. In *Weaving a Lexicon*, D. Hall and S. Waxman, Eds. MIT Press, Cambridge, Massachusetts, 2004, pp. 603–636.

[19] Michaelis, J. *On Formal Properties of Minimalist Grammars*. PhD thesis, Universität Potsdam, 2001. *Linguistics in Potsdam 13*, Universitätsbibliothek, Potsdam, Germany.

[20] Moss, L. Natural language, natural logic, natural deduction. *Forthcoming* (2004). Indiana University.

[21] Motoki, T., Shinohara, T., and Wright, K. The correct definition of finite elasticity; corrigendum to identification of finite unions. In *The Fourth Annual Workshop on Computational Learning Theory* (San Mateo, California, 1989), Morgan Kaufmann, p. 375.

[22] NIYOGI, P. *The Computational Nature of Language Learning and Evolution.* MIT Press, Cambridge, Massachusetts, 2003. Forthcoming.

[23] PINKER, S. *Language Learnability and Language Development.* Harvard University Press, Cambridge, Massachusetts, 1984.

[24] PINKER, S. The bootstrapping problem in language acquisition. In *Mechanisms of Language Acquisition*, B. MacWhinney, Ed. Lawrence Erlbaum, Hillsdale, New Jersey, 1987.

[25] PITT, L. *Probabilistic inductive inference.* PhD thesis, University of Illinois, 1989.

[26] PRATT-HARTMANN, I. Fragments of language. *Journal of Logic, Language and Information 13*, 2 (2004), 207–223.

[27] PRATT-HARTMANN, I., AND THIRD, A. More fragments of language. *Notre Dame Journal of Formal Logic Forthcoming* (2004).

[28] PRAWITZ, D. Ideas and results in proof theory. In *Proceedings of the Second Scandinavian Logic Symposium*, J. Fenstad, Ed. North-Holland, Amsterdam, 1971, pp. 235–307. Partially reprinted as "Gentzen's analysis of first order proofs," in R.I.G. Hughes, *A Philosophical Companion to First Order Logic*, Hackett: Indianapolis, 1993.

[29] SISKIND, J. M. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition 61* (1996), 39–91.

[30] STABLER, E. P. Derivational minimalism. In *Logical Aspects of Computational Linguistics*, C. Retoré, Ed. Springer-Verlag (Lecture Notes in Computer Science 1328), NY, 1997, pp. 68–95.

[31] TIEDE, H.-J. Identifiability in the limit of context-free generalized quantifiers. *Journal of Language and Computation 1* (1999).

[32] TRUESWELL, J., AND GLEITMAN, L. Children's eye movements during listening: Developmental evidence for a constraint-based theory of lexical processing. University of Pennsylvania, 2004.

[33] VAN BENTHEM, J. *Essays in Logical Semantics.* Reidel, Dordrecht, 1986.

[34] VAN BENTHEM, J. Semantic automata. In *Studies in Discourse Representation Theory and the Theory of Generalized Quantifiers*, J. Groenendijk, D. de Jongh, and M. Stokhof, Eds. Foris, Dordrecht, 1987.

[35] VAN BENTHEM, J. Towards a computational semantics. In *Generalized Quantifiers: Linguistic and Logical Approaches*, P. Gärdenfors, Ed. Reidel, Boston, 1987.

[36] WRIGHT, K. Identification of unions of languages drawn from an identifiable class. In *The Second Annual Workshop on Computational Learning Theory* (San Mateo, California, 1989), Morgan Kaufmann, pp. 328–333.

17 The syllogistic fragment

> The not-relation is one of the simplest and most fundamental relations known to the human mind. For the study of logic, no more important and fruitful relation is known. (Royce 1917)

> All human systems of communication contain a representation of negation. No animal communication system includes negative utterances... (Horn 2001)

17.1 Syntax: N+some+every+no+names+not$_v$

(1) The syntax of main verbs, auxiliary verbs and negation in English is rather complex. Some simplistic grammars do not even get these basic patterns that were noticed in (Chomsky 1957):

the cat is a danger	the cat ate the monkey	the cat will have been eating the monkey
the cat is not a danger	the cat did not eat the monkey	the cat will not have been eating the monkey
is the cat not a danger?	did the cat not eat the monkey?	will the cat not have been eating the monkey?
isn't the cat a danger?	didn't the cat eat the monkey?	won't the cat have been eating the monkey?

(2) We assume that auxiliaries like *be* require "verbal licensing" so that they "move around" *not*.

(3) The subject argument of the verb is not adjacent to the VP, but on the other side of Neg and I(nfl). This is achieved by letting DP's be -k, "minus case," requiring licensing.

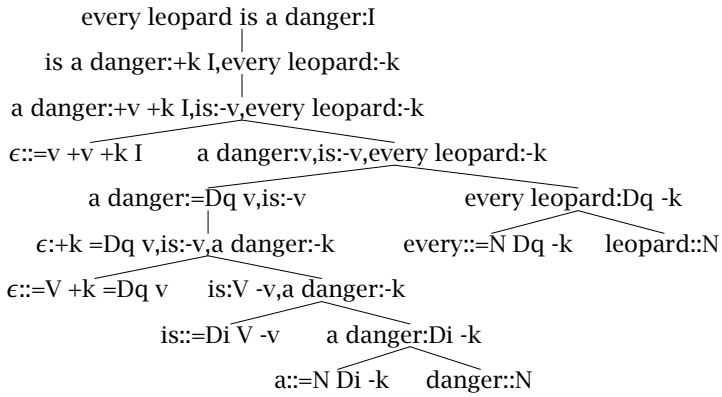
(4) **Burzio's generalization:** if a verb assigns objective case, it selects a subject.[1]
This is achieved by dividing the verb phrase into a higher part vp and a lower part VP, and letting v both assign case and select the subject.

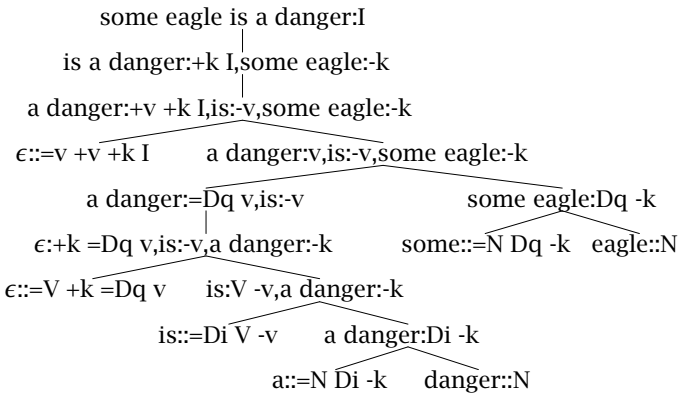
G: a lexicon Lex of 15 items (but we allow any finite number of nouns and names)

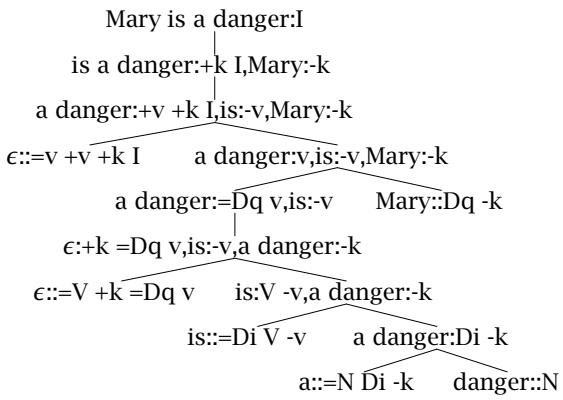
snake::N	leopard::N	(nouns as before)
eagle::N	danger::N	
every::=N Dq -k	a::=N Di -k	(determiners now require case)
some::=N Dq -k	no::=N Dq -k	
Mary::Dq -k	John::Dq -k	(names, requiring case too)
is::=Di V -v	ϵ::=V +k =Dq v	(verb phrase split into V and v)
not::=v Neg		(negation selects v)
ϵ::=v +v +k I	ϵ::=Neg +v +k I	(inflection selects+licenses v,assigns subj case)

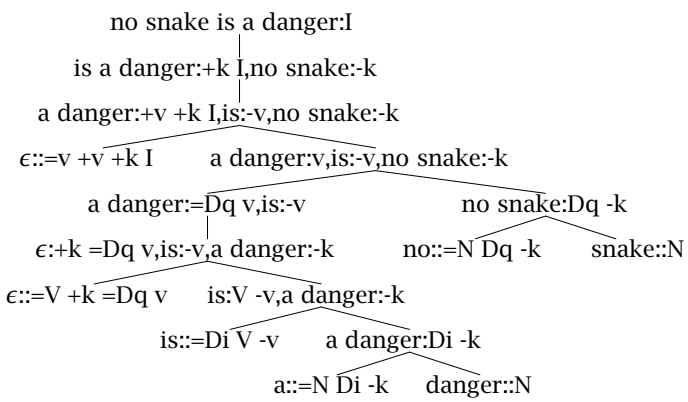
structure building rules $\mathcal{F} = \{merge, move\}$, fixed for all grammars.
$L(G) = closure(Lex, \mathcal{F})$ (everything you can build from the lexicon with the rules)
$\Gamma(G)$ the set of all the derivations of every element of $L(G)$

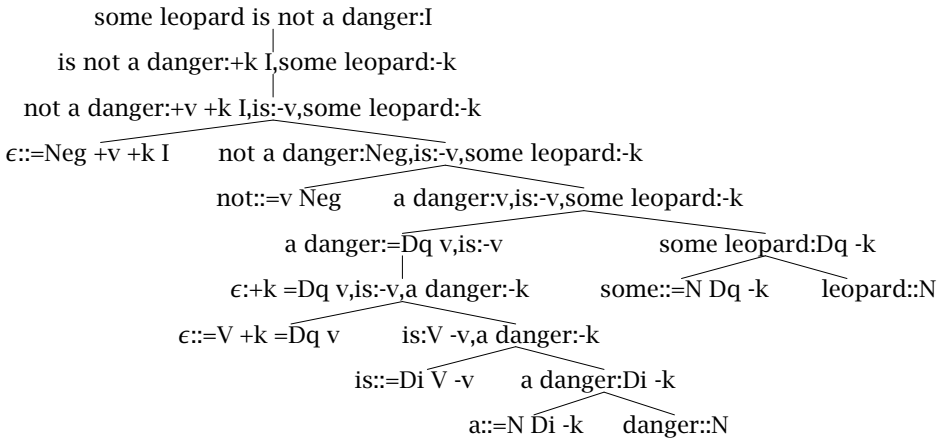
[1] There are now many variants of Burzio's original idea [1]. For a recent review see for example [20].


every leopard is a danger:I
is a danger:+k I,every leopard:-k
a danger:+v +k I,is:-v,every leopard:-k
ϵ::=v +v +k I
a danger:v,is:-v,every leopard:-k
a danger:=Dq v,is:-v
every leopard:Dq -k
ϵ:+k =Dq v,is:-v,a danger:-k
every::=N Dq -k
leopard::N
ϵ::=V +k =Dq v
is:V -v,a danger:-k
is::=Di V -v
a danger:Di -k
a::=N Di -k
danger::N


some eagle is a danger:I
is a danger:+k I,some eagle:-k
a danger:+v +k I,is:-v,some eagle:-k
ϵ::=v +v +k I
a danger:v,is:-v,some eagle:-k
a danger:=Dq v,is:-v
some eagle:Dq -k
ϵ:+k =Dq v,is:-v,a danger:-k
some::=N Dq -k
eagle::N
ϵ::=V +k =Dq v
is:V -v,a danger:-k
is::=Di V -v
a danger:Di -k
a::=N Di -k
danger::N


Mary is a danger:I
is a danger:+k I,Mary:-k
a danger:+v +k I,is:-v,Mary:-k
ϵ::=v +v +k I
a danger:v,is:-v,Mary:-k
a danger:=Dq v,is:-v
Mary::Dq -k
ϵ:+k =Dq v,is:-v,a danger:-k
ϵ::=V +k =Dq v
is:V -v,a danger:-k
is::=Di V -v
a danger:Di -k
a::=N Di -k
danger::N


no snake is a danger:I
is a danger:+k I,no snake:-k
a danger:+v +k I,is:-v,no snake:-k
ϵ::=v +v +k I
a danger:v,is:-v,no snake:-k
a danger:=Dq v,is:-v
no snake:Dq -k
ϵ:+k =Dq v,is:-v,a danger:-k
no::=N Dq -k
snake::N
ϵ::=V +k =Dq v
is:V -v,a danger:-k
is::=Di V -v
a danger:Di -k
a::=N Di -k
danger::N



17.2 Parsing: efficient, as before

17.3 Semantics

(5) Adults tend to interpret phrases in their pronounced positions, where linear positions corresponds to scope. And some results suggest that this tendency is even stronger in children, though what is really going on in children is controversial [10]. Scoping options are linguistically restricted, and among available options, pragmatic considerations can sometimes exclude or favor one or another reading: [19, 4]

Every representative of a major company arrived Every woman had a baby

(6) In the derivations of negative sentences shown just above, Neg applies to the whole VP with Subj and Obj, but we often want the subject outside the scope of Neg.

For example: *some leopard is not a danger* ≡ some(leopard,not(danger))
≢ not(some(leopard,danger))

(7) **A simple idea for this fragment:**

Scopal relations are determined by surface position.

We know this is **not true** for human languages, but it is a good first approximation, and quite different from our earlier semantics.

semantics: any $\mu : \Gamma(G) \to (E, 2)$ defined as follows (where x and y are any distinct feature occurrences):

$\mu(n :: \text{N}) \subseteq E$
$\mu(n :: \text{Dq -k}) \in E$
$\mu(some :: \text{=N Dq -k})$ is the function SOME that maps a property P to $\text{SOME}(P)$, where
  $\text{SOME}(P)$ is the function that maps property Q to 1 iff $|P \cap Q| > 0$
$\mu(every :: \text{=N Dq -k})$ is the function EVERY that maps a property P to $\text{EVERY}(P)$, where
  $\text{EVERY}(P)$ is the function that maps property Q to 1 iff $|P - Q| = 0$
$\mu(no :: \text{=N Dq -k})$ is the function NO that maps a property P to $\text{NO}(P)$, where
  $\text{NO}(P)$ is the function that maps property Q to 1 iff $|P \cap Q| = 0$
$\mu(not :: \text{=v Neg}) =$ the function $\neg$ that maps $(A_0, A_1, \ldots, A_n)$ to $(\neg A_0, A_1, \ldots, A_n)$
$\mu(\text{all other lexical elements}) = \text{id}$, the identity function

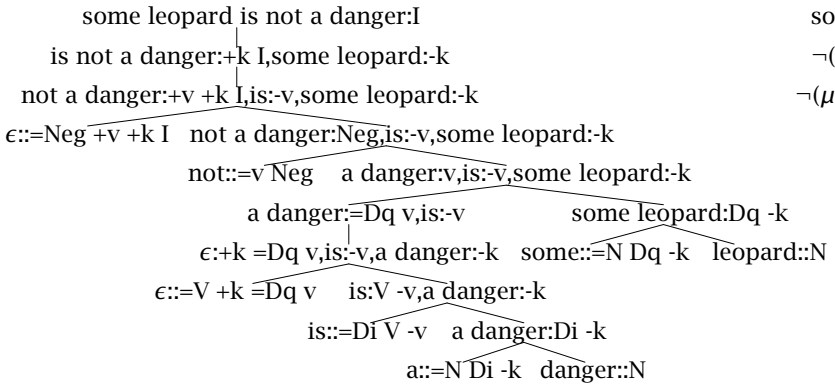
when B_0 has no -x feature: $\mu(merge((A_0, \ldots, A_i), (B_0, \ldots, B_j))) = (A, A_1, \ldots, A_i, B_1, \ldots, B_j)$
where: $A = \begin{cases} \mu(A_0)(\mu(B_0)) & \text{if defined} \\ \mu(B_0)(\mu(A_0)) & \text{otherwise} \end{cases}$

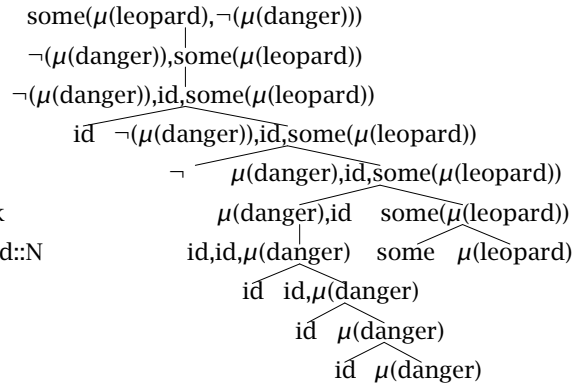
when B_0 has a -x feature: $\mu(merge((A_0, \ldots, A_i), (B_0, \ldots, B_j))) = (\mu(A_0), \ldots, \mu(A_i), \mu(B_0), \ldots, \mu(B_j))$

when A_i has no -y feature: $\mu(move(A_0[\text{+x}], \ldots, A_i[\text{-x}], \ldots, A_n)) = (A, \mu(A_1), \ldots, \mu(A_{i-1}), \mu(A_{i+1}), \ldots, \mu(A_n$
where: $A = \begin{cases} \mu(A_0)(\mu(A_i)) & \text{if defined} \\ \mu(A_i)(\mu(A_0)) & \text{otherwise} \end{cases}$

when A_i has a -y feature: $\mu(move(A_0[\text{+x}], \ldots, A_i[\text{-x}], \ldots, A_n)) = (\mu(A_0), \ldots, \mu(A_n))$

Notice that N and lexical Dq denotations can vary; all other aspects of the interpretation are fixed.





17.4 Digression: Dq and individuals

(8) One inelegance in our semantics is that quantified determiner phrases like *John* denote entities, while quantified determiner phrases like *every leopard* denote functions from properties to truth values!

(9) Another inelegance of our semantics is that while there is a Boolean meet for the domain of truth values (the $\wedge$ function), for the domain of properties, for the domain of quantifiers, and for the domain of DqP denotations, but there is no meet in the domain E of individuals.

So we immediately can define interpretations for

(John is a leopard) and (John is a danger)
John is (a leopard and a danger)
(some but not every) leopard is a danger
(some leopard and every tiger) is a danger

but we will have to do some extra work to interpret something like

(John and Mary) sing.

(10) Montague, Keenan and some other linguists advocate interpreting names not as denoting entities, but as denoting "individual quantifiers," quantifiers that map a property to true iff some particular individual is in it.

Then, all the DqP's could have the same kind of denotation: they are all functions from properties to truth values. This domain has a Boolean meet to interpret *and*.

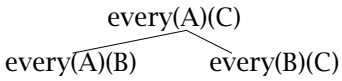
On this view, your name does not denote you, but rather denotes all the properties you have.

Furthermore, the functions from properties to truth values are "individual quantifiers" iff they are Boolean homomorphisms in the sense defined in the exercise in lecture 13.

17.5 Inference: shown linguists' style, root up; trees at nodes named with logicians' abbreviations

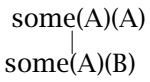
i. *(every reflexive)* every(A)(A)

ii. *(every trans)* every is transitive.

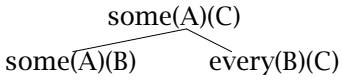


iii. *(some perm)* The order of the nouns in the subject and object does not matter,

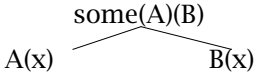
iv. *(some quasi-reflexive)*



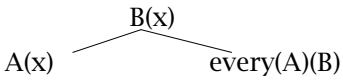
v. *(some2 increasing)*



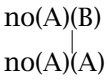
vi. *(name some)*



vii. *(name every)*

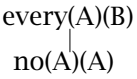


viii. *(no)*

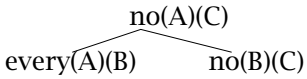


ix. *(no perm)* The order of the nouns in the subject and object does not matter,

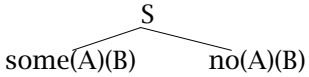
x. *(no to every)*



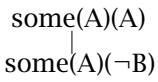
xi. *(every to no)*



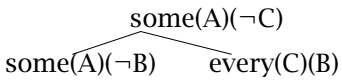
xii. *(contradiction)* For any sentence S:



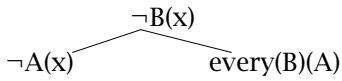
xiii. *(some quasi-reflexive II)*



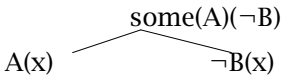
xiv. *(some-not2 decreasing)*



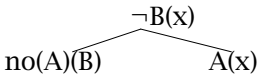
xv. *(name-not2 decreasing)*



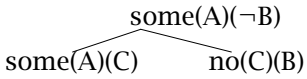
xvi. *(name not to some not)*



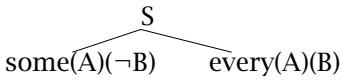
xvii. *(no to name not)*



xviii. *(some no to some not)*



xix. *(contradiction II)* For any sentence S:



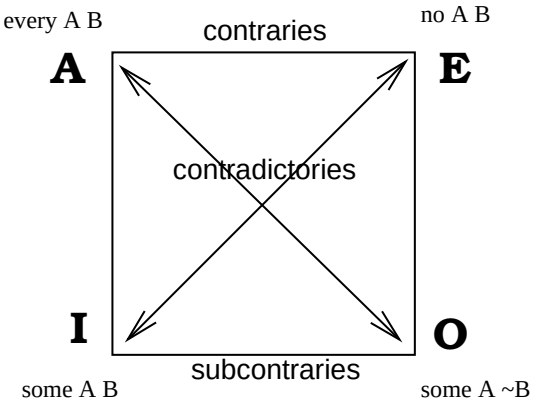
17.6 'Classical' syllogisms: a different interpretation of *every*

As we discussed, many people assume that *every leopard is a danger* implies that there is a leopard. So maybe we should usually interpret *every* as the Boolean conjunction of the standard logical EVERY with the standard SOME, like this:

μ(*every* :: =N Dq − k) is the function SOME ∧ EVERY that maps a property P to (SOME ∧ EVERY)(P), where (SOME ∧ EVERY)(P) is the function that maps property Q to 1 iff $|P \cap Q| > 0$ and $|P - Q| = 0$

Some of the well-known Aristotelian syllogisms depend on this interpretation.

Aristotle noticed many patterns of valid argument in his *Prior Analytics* (350BC). Some of them are surprising, and they fall into such simple patterns that they have turned more than one clever scholar into a mystic:



The letters A, E, I O label four types of quantified statements, which are obtained by varying two binary parameters: universal/particular and affirmative/negative:

A	universal	affirmative	Every A is a B
E	universal	negative	No A is a B
I	particular	affirmative	Some A is a B
O	particular	negative	Some A is not a B

The other notions indicated in the square are defined as follows:

- Statements p, q are **contradictory** iff they cannot both be true, and they cannot both be false.
- Statements p, q are **contrary** iff they cannot both be true
- Statements p, q are **subcontraries** iff they cannot both be false

Aristotle used regular variations on some simple patterns or "figures," to get the syllogisms. From the first and most basic "figure," Aristotle derives the following "perfect" syllogisms:

	premise 1	premise 2	conclusion	traditional name
1	every A B	every B C	every A C	Barbara
2	no A B	every C A	no C B	Celarent
3	every A B	some C A	some C B	Darii
4	no A B	some A C	some A ¬B	Ferio

The vowels in the traditional names correspond to the types of the premises, and other details of the names have significance too – see for example [17]. From the "second figure:"

	premise 1	premise 2	conclusion	traditional name
5	every A B	no C B	no C A	Camestres
6	no A B	every C B	no C A	Cesare
7	no A B	some C B	some C ¬A	Festimo
8	every A B	some C ¬B	some C ¬A	Baroco

From the "third figure:"

	premise 1	premise 2	conclusion	traditional name
9	every A B	every A C	some C B	Darapti
10	no A B	every A C	some C ¬B	Felapton
11	some A B	every A C	some C B	Disamis
12	every A B	some A C	some C B	Datisi
13	some A ¬B	every A C	some C ¬B	Bocardo
14	no A B	some A C	some C ¬B	Ferison

From a "fourth figure:" that was introduced after Aristotle:

	premise 1	premise 2	conclusion	traditional name
15	every A B	every B C	some C A	Bramantip
16	every A B	no B C	no C A	Camenes
17	some A B	every B C	some C A	Dimaris
18	no A B	every B C	some C ¬A	Fesapo
19	no A B	some B C	some C ¬A	Fresison

We have to go beyond Aristotelian syllogisms to get the inference,

Socrates is a man; every man is a mortal; so Socrates is a mortal.

Since our fragment includes names and rules for reasoning with them, we have captured this too.

> *I found no basis prepared; no models to copy…Mine is the first step and therefore a small one, though worked out with much thought and hard labor. You, my readers or hearers of my lectures, if you think I have done as much as can fairly be expected of an initial start…will acknowledge what I have achieved and will pardon what I have left for others to accomplish.* (Aristotle)

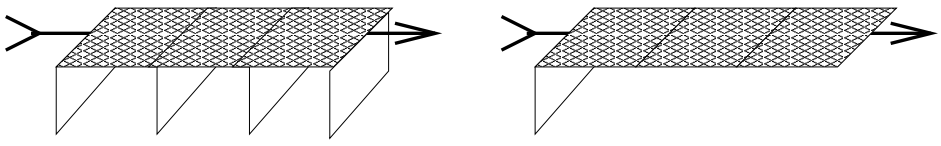
17.7 Digression: inference and shallowness

(11) It is often suggested that commonsense reasoning is "robust" and tends to be shallow and in need of support from multiple sources, while scientific and logical inference is "delicate" and relies on long chains of reasoning with very few points of support.

[13, pp193,189]

(12) Minsky puts the matter this way:

> *That theory is worthless. It isn't even wrong!* – WOLFGANG PAULI
>
> As scientists we like to make our theories as delicate and fragile as possible. We like to arrange things so that if the slightest thing goes wrong, everything will collapse at once!…
>
> Here's one way to contrast logical reasoning and ordinary thinking. Both build chainlike connections between ideas…



Commonsense Reasoning

Mathematical Logic

> Logic demands just one support for every link, a single, flawless deduction. Common sense asks, at every step, if all of what we've found so far is in accord with everyday experience. No sensible person ever trusts a long, thin chain of reasoning. In real life, when we listen to an argument, we do not merely check each separate step; we look to see if what has been described so far seems plausible. We look for other evidence beyond the reasons in that argument.

Of course, shallow thinking can often be wrong too! In fact, it seems that in language understanding, there are many cases where we seem to make superficial assumptions even when we know they are false. For example, Kamp tells the following story which I think (hope) is now out of date.

[7, p39]

> We are much assisted in our making of such guesses [about the referents of pronouns] by the spectrum of our social prejudices. Sometimes, however, these may lead us astray, and embarrassingly so, as in the following riddle which advocates of Women's Lib have on occasion used to expose members of the chauvinistic rearguard: In a head-on collision both father and son are critically wounded. They are rushed into a hospital where the chief surgeon performs an emergency operation on the son. But it is too late and the boy dies on the operating table. When an assistant asks the surgeon, 'Could you have a look at the other victim?', the surgeon replies 'I could not bear it. I have already lost my son.' Someone who has the built-in conception that the chief surgeons are men will find it substantially more difficult to make sense of this story than those who hold no such view.

What is interesting in the present context is that this story was puzzling in 1984 even for people who knew perfectly well that many surgeons were women, because the stereotypical surgeon was still a man. That is, superficial reasoning can relies on stereotypes that are false, and to be clear to your audience it is important to state things in a way that anticipates and avoids confusions that may be caused by them. The role of superficial assumptions has been explored in studies of conceptual "prototypes" (mentioned earlier) in language processing. [11, 3, 16, 14]

(13) This raises issues about the possibility of dividing linguistic knowledge from other 'background' knowledge. For example, it is historical and literary knowledge that Shakespeare was a great poet, but the knowledge of the many common Shakespearean word sequences is linguistic and perfectly familiar to most speakers. If we start thinking of familiar phrasing as a linguistic matter, this could actually take us quite far into what would have been regarded as world knowledge.

(This kind of linguistic knowledge is often tapped by clues for crossword puzzles. Although solving crossword puzzles from clues involves many domains of human knowledge, it draws particularly on *how that knowledge is conventionally represented in language*, and so theories about crossword solving overlap with language modeling methods to a rather surprising degree!) [8]

(14) **Depth-first reasoning**, pursuing one line of reasoning to the end (i.e. to success, to failure in which case we backtrack, or to nontermination) is not a reasonable model of the kind of superficial reasoning that goes on in commonsense understanding of entailment relations among sentences. Really, it is not clear what kind of model could even come close to explaining human-like performance, but we can do better than depth-first.

A better idea that has been used in theorem-proving and game-playing applications is **iterative deepening**. This strategy searches for a shallow proof first (e.g. a proof with depth = 0), and then if one is not found at that depth, increases the depth bound and tries again. Cutting this search off at a reasonably shallow level will have the consequence that the difficult theorems will not be found, but all the easy ones will be. [9, 18]

(15) In most natural language applications, the set of premises (background knowledge) inference rules and meaning postulates may be very large, so we will typically be wanting to see whether some particular proposition can be proven, the most natural strategy is "backward chaining:" we match the statement we want to prove with a conclusion of some inference scheme, and see if there is a way of stepping back to premises that are accepted, where the number of steps taken in this way is bounded by the iterative deepening method.

(16) The psychologist Johnson-Laird has argued in a series of works that the notion of a "mental logic" of reasoning is ludicrous on both conceptual and empirical grounds, but Macnamara and other have pointed out that with an appropriate recognition of the complexity of the relation between the principles of reasoning and the performance of reasoning, most or all of Johnson-Laird's objections become much less compelling. [6] [12, pp45-48]

Exercise 6 Due Thursday, Mar 10 (this is the 5th required exercise)

(1) Sorry to do this to you, but we should have a tiny bit more practice with doing derivations that involve movement. Using the grammar for the syllogistic fragment, show the whole derivation tree for the sentence

Mary is not a danger.

If you do this one carefully (and trying not to look too much at the other trees in the notes) it should be like doing a crossword puzzle, but (hopefully) easier.

(2) Suppose we interpret *every* as EVERY $\wedge$ SOME, as defined above.

Then when Aristotle uses a premise of type A, we can regard it as 2 premises expressed with our logician's *every* and *some*, and when he has a conclusion of type A, we can regard it as 2 conclusions expressed with our logician's *every* and *some*.

With this interpretation, for each of Aristotle's syllogisms 1-9, provide proofs using our inference system if the syllogisms are sound, and otherwise explain why they are not sound.

(3-optional!!) Do the previous exercise for all 19 of the syllogisms listed above.

(4-optional!!) Indicate which of the syllogisms in (2) (and 3 if you did it) you regard as "obvious" – things that an ordinary, linguistically competent human is likely to notice immediately, without recourse to pencil and paper.

(Are the "obvious" ones distinguished in any way by some characteristic features?)

References for Lecture 17

[1] Burzio, L. *Italian Syntax: A Government-Binding Approach*. Reidel, Boston, 1986.

[2] Chomsky, N. *Syntactic Structures*. Mouton, The Hague, 1957.

[3] Dahlgren, K. *Naive Semantics for Natural Language Understanding*. Kluwer, Boston, 1988.

[4] Hobbs, J. R., and Shieber, S. M. An algorithm for generating quantifier scopings. *Computational Linguistics 13* (1987), 47–63.

[5] Horn, L. R. *A Natural History of Negation*. CSLI Publications, Stanford, California, 2001.

[6] Johnson-Laird, P. N. *Mental Models*. Harvard University Press, Cambridge, Massachusetts, 1983.

[7] Kamp, H. A theory of truth and semantic representation. In *Formal Methods in the Study of Language*, G. Groenendijk, T. Janssen, and M. Stokhof, Eds. Foris, Dordrecht, 1984.

[8] Keim, G. A., Shazeer, N., Littman, M. L., Agarwal, S., Cheves, C. M., Fitzgerald, J., Grosland, J., Jiang, F., Pollard, S., , and Weinmeister, K. Proverb: The probabilistic cruciverbalist. In *Proceedings of the National Conference on Artificial Intelligence, AAAI-99* (1999), Morgan Kaufmann.

[9] Korf, R. E. An optimum admissible tree search. *Artificial Intelligence 27* (1985), 97–109.

[10] Lidz, J., and Musolino, J. Children's command of quantification. *Cognition 84* (2002), 113–154.

[11] Lynch, E. B., Coley, J. D., and Medin, D. L. Tall is typical. *Memory and Cognition 28* (2000), 41–50.

[12] Macnamara, J. *A Border Dispute: The place of logic in psychology*. MIT Press, Cambridge, Massachusetts, 1986.

[13] Minsky, M. L. *The Society of Mind*. Simon and Schuster, NY, 1988.

[14] Rosch, E. Principles of categorization. In *Cognition and categorization*, E. Rosch and B. Lloyd, Eds. Erlbaum, Hillsdale, New Jersey, 1978.

[15] Royce, J. Negation. In *Encyclopedia of Religion and Ethics*, J. Hastings, Ed. Scribner Sons, NY, 1917.

[16] Smith, E. E., and Medin, D. L. *Categories and Concepts*. Harvard University Press, Cambridge, Massachusetts, 1981.

[17] Spade, P. V. *Thoughts, Words and Things: An Introduction to Late Mediaeval Logic and Semantic Theory, Version 1.1a*. Indiana University, Bloomington, 1992.

[18] Stickel, M. E. A prolog technology theorem prover: a new exposition and implementation in prolog. *Theoretical Computer Science 104* (1992), 109–128.

[19] Szabolcsi, A., Ed. *Ways of Scope Taking*. Kluwer, Boston, 1996.

[20] Woolford, E. Burzio's generalization, markedness, and locality constraints on nominative objects. In *New Perspectives on Case Theory*, E. Brandner and H. Zinsmeister, Eds. CSLI, Stanford, California, 2003.

18 Inferences in the syllogistic fragment

> *It helps to think of the syntax and formal lexicon together as defining a family of fragments of English, each member of which is determined by its content lexicon. We denote this family of fragments by* $\mathcal{E}_0$*...*
> Theorem 1. *The problem of determining the satisfiability of a set of sentences in* $\mathcal{E}_0$ *is in PTIME.*
> *One way to generalize the fragment* $\mathcal{E}_0$ *is to add relative clauses [obtaining* $\mathcal{E}_1$*]...*
> Theorem 2. *The problem of determining the satisfiability of a set of sentences in* $\mathcal{E}_1$ *is NP-complete.*
> (Pratt-Hartmann 2004)

18.1 Syllogistic fragment (similar to $\mathcal{E}_0$)

syntax G: a lexicon Lex of 15 items (but we allow any finite number of nouns and names)

snake::N	leopard::N	(nouns as before)
eagle::N	danger::N	
every::=N Dq -k	a::=N Di -k	(determiners now require case)
some::=N Dq -k	no::=N Dq -k	
Mary::Dq -k	John::Dq -k	(names, requiring case too)
is::=Di V -v	ϵ::=V +k =Dq v	(verb phrase split into V and v)
not::=v Neg		(negation selects v)
ϵ::=v +v +k I	ϵ::=Neg +v +k I	(inflection selects+licenses v,assigns subj case)

structure building rules $\mathcal{F} = \{merge, move\}$, fixed for all grammars.

semantics: any $\mu : \Gamma(G) \to (E, 2)$ defined as follows (where x and y are any distinct feature occurrences):

$\mu(n :: \text{N}) \subseteq E$
$\mu(n :: \text{Dq -k}) \in E$
$\mu(some :: \text{=N Dq -k}) = \textsc{some}$, $\mu(every :: \text{=N Dq -k}) = \textsc{every}$, $\mu(no :: \text{=N Dq -k}) = \textsc{no}$
$\mu(not :: \text{=v Neg}) =$ the function $\neg$ that maps $(A_0, A_1, \ldots, A_n)$ to $(\neg A_0, A_1, \ldots, A_n)$
$\mu(\text{all other lexical elements}) = \text{id}$, the identity function

when B_0 has no -x feature: $\mu(merge((A_0, \ldots, A_i), (B_0, \ldots, B_j))) = (A, A_1, \ldots, A_i, B_1, \ldots, B_j)$

$$\text{where: } A = \begin{cases} \mu(A_0)(\mu(B_0)) & \text{if defined} \\ \mu(B_0)(\mu(A_0)) & \text{otherwise} \end{cases}$$

when B_0 has a -x feature: $\mu(merge((A_0, \ldots, A_i), (B_0, \ldots, B_j))) = (\mu(A_0), \ldots, \mu(A_i), \mu(B_0), \ldots, \mu(B_j))$

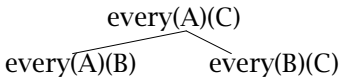
when A_i has no -y feature: $\mu(move(A_0[\text{+x}], \ldots, A_i[\text{-x}], \ldots, A_n)) = (A, \mu(A_1), \ldots, \mu(A_{i-1}), \mu(A_{i+1}), \ldots, \mu(A_n))$

$$\text{where: } A = \begin{cases} \mu(A_0)(\mu(A_i)) & \text{if defined} \\ \mu(A_i)(\mu(A_0)) & \text{otherwise} \end{cases}$$

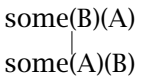
when A_i has a -y feature: $\mu(move(A_0[\text{+x}], \ldots, A_i[\text{-x}], \ldots, A_n)) = (\mu(A_0), \ldots, \mu(A_n))$

Inference: i. *(every reflexive)* every(A)(A)

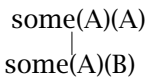
ii. *(every trans)* every is transitive.



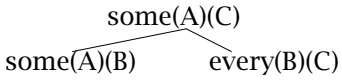
iii. *(some perm)* The order of the nouns in the subject and object does not matter,



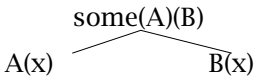
iv. *(some quasi-reflexive)*



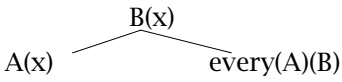
v. *(some2 increasing)*



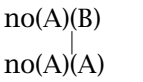
vi. *(name some)*



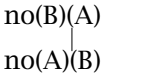
vii. *(name every)*



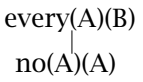
viii. *(no)*



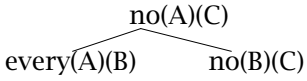
ix. *(no perm)* The order of the nouns in the subject and object does not matter,



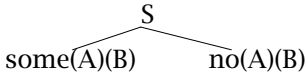
x. *(no to every)*



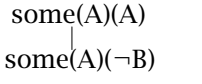
xi. *(every to no)*



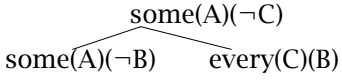
xii. *(contradiction)* For any sentence S:



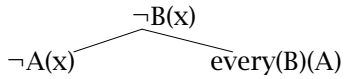
xiii. *(some quasi-reflexive II)*



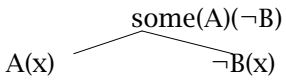
xiv. *(some-not2 decreasing)*



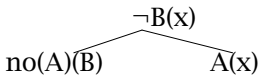
xv. *(name-not2 decreasing)*



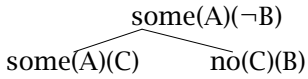
xvi. *(name not to some not)*



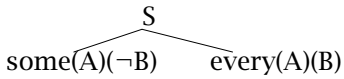
xvii. *(no to name not)*



xviii. *(some no to some not)*



xix. *(contradiction II)* For any sentence S:



18.2 Metatheory

(1) This kind of modern proof system for syllogistic logic has been studied by Corcoran and Moss. It is sound and complete. [1, 6]

(2) (No canonical models) There are sets of expressions A, such that no model verifies exactly the sentences S such that A⊢S.

(3) The semantic negation of every sentence is expressible.

¬every A is a B	≡	some A is not a B
¬some A is a B	≡	every A is not a B
¬no A is a B	≡	some A is a B
¬every A is not a B	≡	some A is not a B
¬some A is not a B	≡	every A is a B
¬no A is not a B	≡	some A is not a B
¬x is a A	≡	x is not a A
¬x is not a A	≡	x is a A

(4) Every consistent set extends to a maximal consistent set, which can be used to define a model.

18.3 The complexity of ⊢

(5) In principle, we can decide whether $A \vdash e$ for any particular finite set of sentences A and any particular sentence e by closing A under the inference rules, and then checking to see if e is in that closure.

When A is inconsistent (which can happen with just 2 sentences!) this closure will contain every sentence in the language.

(6) Pratt-Hartmann shows uses a different strategy to show that we can decide instances of ⊢ in polynomial time: he translates the sentences into first order logic, then into clausal form, then applies resolution. [9]

(7) **Convert sentences $A \cup \{\neg e\}$ to first order logic.** I use a prolog notation for first order logic, in which variables begin with uppercase letters:

		every A is a B	↦	`every(X,a(X)=>b(X))`
		some A is a B	↦	`some(X,a(X)&b(X))`
		no A is a B	↦	`-some(X,a(X)&b(X))`
		every A is not a B	↦	`every(X,a(X)=>-b(X))`
		some A is not a B	↦	`some(X,a(X)&-b(X))`
		no A is not a B	↦	`-some(X,a(X)&-b(X))`
		x is a A	↦	`a(x)`
		x is not a A	↦	`-a(x)`
¬every A is a B	≡	some A is not a B	↦	`some(X,a(X)&-b(X))`
¬some A is a B	≡	every A is not a B	↦	`every(X,a(X)=>-b(X))`
¬no A is a B	≡	some A is a B	↦	`some(X,a(X)&b(X))`
¬every A is not a B	≡	some A is not a B	↦	`some(X,a(X)&-b(X))`
¬some A is not a B	≡	every A is a B	↦	`every(X,a(X)=>b(X))`
¬no A is not a B	≡	some A is not a B	↦	`every(X,a(X)&-b(X))`
¬x is a A	≡	x is not a A	↦	`-a(x)`
¬x is not a A	≡	x is a A	↦	`a(x)`

(8) If this translation is adequate, then we can conclude that $\vdash$ is effectively decidable from the more general result that monadic predicate logic is. But Pratt-Hartmann establishes a much stronger result…

(9) **Convert first order logic to clausal form** We can convert any first order formula ϕ into a clausal form ϕ' such that ϕ is satisfiable iff ϕ' is.

I. Convert the sentence to "prefix normal form" PNF.
This step can be done with the following procedure:

1. Eliminate => and <=> by using the following valid sentences in the => direction to transform the all of the applicable subexpressions:

   ```
   (F <=> G) <=> ((F=> G) & (G=> F))
   ( F=> G) <=> (- F or G)
   ```

2. Move negations "inward" as far as possible, using the following valid sentences in the => direction to transform the all of the applicable subexpressions:

   ```
   - every(X, F) <=> some(X, - F)
   - some(X, F) <=> every(X, - F)
   -(F or G) <=> (- F & - G)
   -(F & G) <=> (- F or - G)
   -- F <=> F
   ```

3. Rename variables if necessary, so that no two quantifiers bind the same variable. For example,

   ```
   every(X,p(X) & some(X,q(X))) <=> every(X,p(X) & some(Y,q(Y))).
   ```

4. Now move all quantifiers to the beginning of the formula by applying the following valid rules in the => direction to relevant subformulas:

```
every(X,F) & G <=>every(X,F & G)     if X does not occur free in G
F & every(X,G) <=> every(X,F & G)    if X does not occur free in F
some(X,F) & G <=> some(X,F & G)      if X does not occur free in G
F & some(X,G) <=> some(X,F & G)      if X does not occur free in F

every(X,F) or G <=> every(X,F or G)  if X does not occur free in G
F or every(X,G) <=> every(X,F or G)  if X does not occur free in F
some(X,F) or G <=> some(X,F or G)    if X does not occur free in G
F or some(X,G) <=> some(X,F or G)    if X does not occur free in F
```

II. Eliminate existential quantifiers.
Use the following procedure. Given a PNF sentence,

`Q1(X1,…Qn(Xn,M)…)`

where each `Qi(X,...)` is a quantifier `some(Xi,...)` or `every(Xi,...)` and `M` is a formula that contains no quantifiers, perform the following steps:

1. Take the leftmost existential quantifier `Qr` of the prefix. There are 2 cases to consider:
   a. No universal quantifier occurs before `Qr`. Then choose a new constant `c` and replace `Xr` by `c` wherever `Xr` occurs in `M`. Delete the quantifier `Qr` from the prefix.
   b. The every-quantifiers `Q1 ... Qm` occur before `Qr`, where `m`> 0. (These `m` universal quantifiers are *all* the quantifiers that precede `Qr`.) Choose a new function symbol `f` of arity `m`. Replace all occurrences of `Xr` in `M` by `f(X1,...,Xm)` and delete `Qr` from the prefix.

The constants and functions introduced by the procedure just described are called **Skolem constants** and **Skolem functions**, respectively, after the Norwegian logician Thoralf Skolem [13]. We accordingly call the formula which results from the application of these steps until all the existential quantifiers are eliminated a **Skolem standard form**.

III. Drop the universal quantifiers.
Since there are only universal quantifiers in the prefix, they can be dropped. This yields a **clausal form**: each conjunct is a **clause**.

(10) In sum, our sentences are represented by these clausal forms (shown here in the prolog notation of my implementation, and also in standard set notation):

sentence		prolog notation		set notation
every A is a B	↦	`-a(X) or b(X)`		{-a(X),b(X)}
some A is a B	↦	`a(x) & b(x)`	(for new name: x)	{a(x)},{b(x)}
no A is a B	↦	`-a(X) or -b(X)`		{-a(X),-b(X)}
every A is not a B	↦	`-a(X) or -b(X)`		{-a(X),-b(X)}
some A is not a B	↦	`a(x) & -b(x)`	(for new name: x)	{a(x)},{-b(x)}
no A is not a B	↦	`-a(X) or b(X)`		{-a(X),b(X)}
x is a A	↦	`a(x)`		{a(x)}
x is not a A	↦	`-a(x)`		{-a(x)}

(11) **Decide using resolution.**

Def unification. Two expressions unify with each other just in case there is a substitution of terms for variables that makes them identical. To unify *human(montague)* and *human(X)* we substitute the term *montague* for the variable *X*. We will represent this substitution by the expression $\{X \mapsto montague\}$. Letting $\theta = \{X \mapsto montague\}$, and writing the substitution in "postfix" notation – after the expression it applies to – we have

$$human(X)\theta = human(montague)\theta = human(montague).$$

Notice that the substitution θ has no effect on the term *human(montague)* since this term has no occurrences of the variable *X*.

We can replace more than one variable at once. For example, we can replace *X* by *s(Y)* and replace *Y* by *Z*. Letting $\theta = \{X \mapsto s(Y), Y \mapsto Z\}$, we have:

$$sum(X,Y,Y)\theta = sum(s(Y),Z,Z).$$

Notice that the *Y* in the first term has not been replaced by *Z*. This is because all the elements of the substitution are always applied *simultaneously*, not one after another.

After a little practice, it is not hard to get the knack of finding the (least specific) substitutions that make two expressions identical, if there is one. These substitutions are called (most general) **unifiers** (mgu's), and the step of finding and applying them is called (term) **unification.**[1]

Def resolution. A deduction step for clauses, where for some $0 < i \leq m,\ 0 < j \leq n,\quad \theta = mgu(A_i, \overline{B_j})$:

$$\{A_1, \ldots, A_{i-1}, A_{i+1}, \ldots, A_m, B_1, \ldots B_{j-1}, B_{j+1}, \ldots, B_n\}\theta$$

$$\{A_1, \ldots, A_m\} \qquad \{B_1, \ldots, B_n\}$$

(To avoid variable binding conflicts, we can rename the variables of a clause so that we are never attempting to unify elements of clauses that have any variables in common.)

[11] **Thm.** (Robinson) *S* is an unsatisfiable set of clauses iff the empty clause $\square \in closure(S, \text{resolution})$.

Example. Let's step through the procedure that Pratt-Hartmann uses to decide whether

{some A is a B,every B is a C} ⊢ some A is a C

1. Convert to FOL:

		some A is a B	↦	`some(X,a(X)&b(X))`
		every B is a C	↦	`every(X,b(X)=>c(X))`
¬some A is a C	≡	every A is not a C	↦	`every(X,a(X)=>-c(X))`

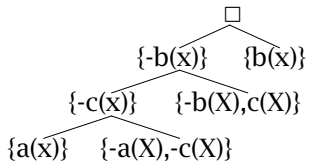
2. Convert FOL to clausal form:

`some(X,a(X)&b(X))`	↦	`a(x) & b(x)`	(for new name: x)	{a(x)},{b(x)}
`every(X,b(X)=>c(X))`	↦	`-b(X) or c(X)`		{-b(X),c(X)}
`every(X,a(X)=>-c(X))`	↦	`-a(X) or -c(X)`		{-a(X),-c(X)}

So the whole set of clauses is this:

{ {a(x)}, {b(x)}, {-b(X),c(X)}, {-a(X),-c(X)} }

3. Closing this set with respect to resolution, we find this derivation:



It follows that yes, it is true that

{some A is a B,every B is a C} ⊢ some A is a C

(12) **Thm.** The problem of determining the satisfiability of a set of sentences in $\mathcal{E}_0$ is in P.

[1] So-called "unification grammars" involve a related, but slightly more elaborate notion of unification [8, 12, 7].

Proof: The steps from sentences $A \cup \{\neg e\} \rightarrow$ FOL and from FOL $\rightarrow$ clauses are polynomial.

The resolution closure will be finite since there are no function symbols. And in fact, we can see that the number of resolution steps is bounded by a quadratic function of the number of different predicates in the set $A \cup \{\neg e\}$, since every resolvent has one of the forms, for term t either a variable or a name,

{p(t),-q(t)} {-p(t),-q(t)} {p(t)} {-p(t)} {}=□

for predicates p, q appearing in $A \cup \{\neg e\}$, and so if there are n predicates there are only $n^2 + n^2 + n + n = 2(n + n^2)$ possible resolvents altogether. □

(13) Noting the applicability of this simple resolution method to the first order translations, Pratt-Hartmann writes, [9]

> …the foregoing analysis should help to lay to rest some appealing but ultimately confused ideas concerning the validity of arguments couched in natural-language-friendly logic. According to its proponents, we obtain a better (i.e. more efficient) method of assessing the validity of arguments couched in natural language if we reason within a logical calculus whose syntax is closer to that of natural language than is – say – first-order logic. The idea is attractive because it suggests an ecological dictum: treat the syntax of natural language with the respect it is due, and your inference processes will run faster. Writers apparently expressing support for such views include Fitch (1973), Hintikka (1974), Suppes (1979), Purdy (1991), and (perhaps) McAllester and Givan (1992). The observations of this paper lend no support to such views, and indeed cast doubt on them. There is no reason, having identified a fragment of natural language, why satisfiability within that fragment should not be decided by first translating into first-order logic and then using procedures appropriate to the fragment of first-order logic so obtained. Indeed, from a complexity-theoretic point of view, there is every reason to believe that, for all but the most impoverished fragments, reasoning using schemata based on the syntax of natural language will confer no advantage whatever.

18.4 Digression: logical form

(14) In a recent manuscript Hobbs writes [4]

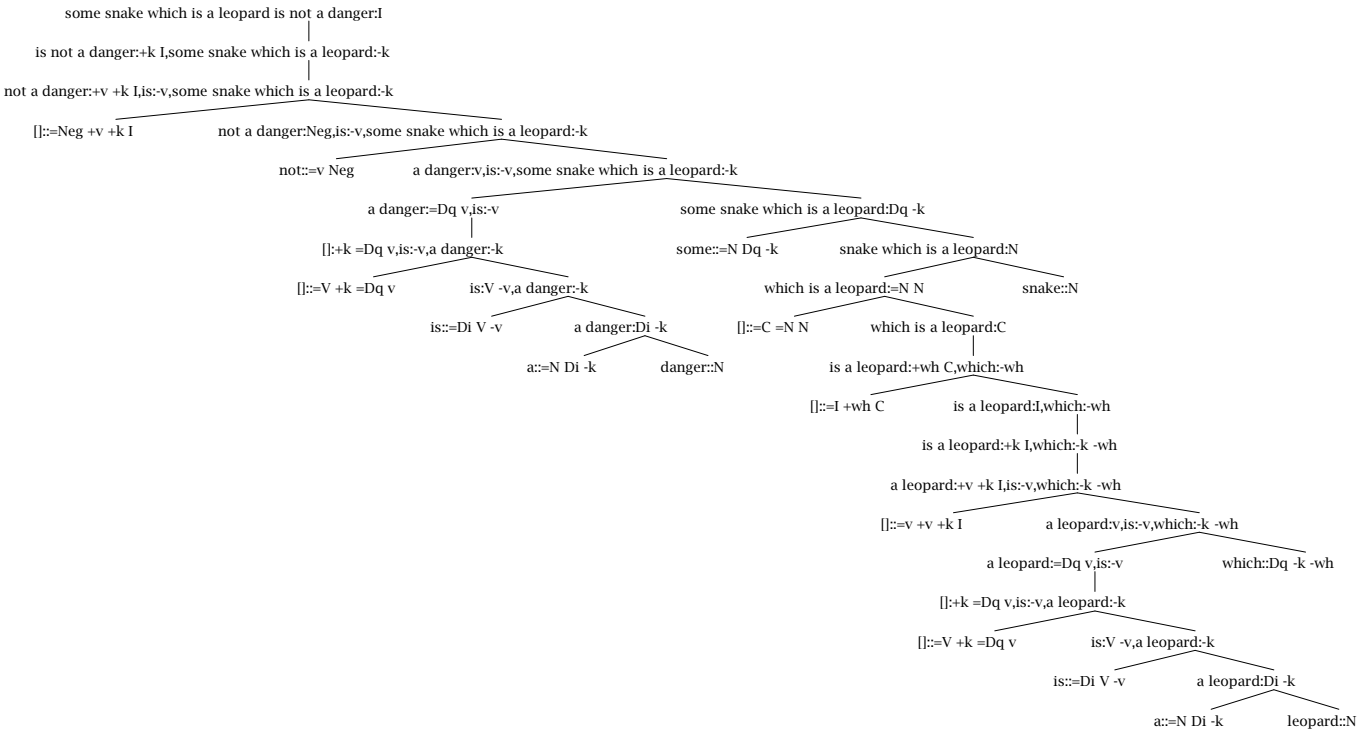
> The role of a logical notation in a theory of discourse interpretation is for representing the knowledge required for understanding texts (which in ordinary life we express in English), and for manipulation by the interpretation process. These uses lead to two principal criteria for a logical notation.
> Criterion I: The notation should be as close to English as possible. This makes it easier to specify the rules for translation between English and the formal language, and also makes it easier to encode in logical notation facts we normally think of in English. The ideal choice by this criterion is English itself, but it fails monumentally on the second criterion.
> Criterion II: The notation should be syntactically simple. Since inference is defined in terms of manipulations performs on expressions in the logical notation, the simpler that notation, the easier it will be to define the inference process.
> A notation is proposed here which is first-order and non-intensional and for which semantic translation can be naively compositional…

18.5 Syllogistic fragment with relative clauses (similar to $\mathcal{E}_1$)

syntax G: a lexicon *Lex* of 15 items (but we allow any finite number of nouns and names)

snake::N	leopard::N	(nouns as before)
eagle::N	danger::N	
every::=N Dq -k	a::=N Di -k	(determiners now require case)
some::=N Dq -k	no::=N Dq -k	
Mary::Dq -k	John::Dq -k	(names, requiring case too)
is::=Di V -v	ϵ::=V +k =Dq v	(verb phrase split into V and v)
not::=v Neg		(negation selects v)
ϵ::=v +v +k I	ϵ::=Neg +v +k I	(inflection selects+licenses v,assigns subj case)
which::Dq -k -wh	ϵ::=I +wh C	
	ϵ::=C =N N	

structure building rules $\mathcal{F} = \{merge, move\}$, fixed for all grammars.



References for Lecture 18

[1] CORCORAN, J. Completeness of ancient logic. *Journal of Symbolic Logic 37* (1972), 696–702.

[2] FITCH, F. Natural deduction rules for English. *Philosophical Studies 24* (1973), 89–104.

[3] HINTIKKA, J. Quantifiers vs. quantification theory. *Linguistic Inquiry 5* (1974), 153–178.

[4] HOBBS, J. R. *Discourse and Inference*. in preparation, ISI/University of Southern California, 2004.

[5] MCALLESTER, D., AND GIVAN, R. Natural language syntax and first order inference. *Artificial Intelligence 56* (1992), 1–20.

[6] MOSS, L. Natural language, natural logic, natural deduction. *Forthcoming* (2004). Indiana University.

[7] POLLARD, C., AND SAG, I. *Head-driven Phrase Structure Grammar*. The University of Chicago Press, Chicago, 1994.

[8] POLLARD, C., AND SAG, I. A. *Information-based Syntax and Semantics*. No. 13 in CSLI Lecture Notes Series. CSLI Publications, Stanford, California, 1987.

[9] PRATT-HARTMANN, I. Fragments of language. *Journal of Logic, Language and Information 13*, 2 (2004), 207–223.

[10] PURDY, W. C. A logic for natural language. *Notre Dame Journal of Formal Logic 32* (1991), 409–425.

[11] ROBINSON, J. A. A machine-oriented logic based on the resolution principle. *Journal of the Association for Computing Machinery 12* (1965), 23–41.

[12] SHIEBER, S. M. *Constraint-based Grammar Formalisms*. MIT Press, Cambridge, Massachusetts, 1992.

[13] SKOLEM, T. Über die mathematische Logik. In *Selected Works in Logic*. Universitetsforlaget, Oslo, 1928, pp. 189–206. An English translation with an introduction by B. Dreben and J. van Heijenoort is reprinted in J. van Heijenoort, ed., *From Frege to Gödel: A Sourcebook in Mathematical Logic, 1879-1931*. Cambridge, Massachusetts: Harvard University Press.

[14] SUPPES, P. Logical inference in English: A preliminary analysis. *Studia Logica 38* (1979), 375–391.

19 Learning the syllogistic fragment

The metaphysically-minded person feels that the actual world is made up solely of positive, specific, determinate, concrete, contingent, individual, sensory facts, and that the appearance of a penumbra of fictional, negative, general, indeterminate, abstract, necessary, super-individual, physical facts is somehow only an appearance due to a lack of penetration upon our part. (Wisdom 1969)

It is commonly observed that our intuitive application of the term 'false' is largely governed by the principle that a statement is false if and only if its negation is true, supplemented by a general disposition on our part to construe as the negation of a statement the simplest plausible candidate for that role. (Dummett 1981, p.108)

19.1 Syllogistic fragment

syntax G:

Lex:	snake::N	leopard::N	(nouns as before)
	eagle::N	danger::N	
	every::=N Dq -k	a::=N Di -k	(determiners now require case)
	some::=N Dq -k	no::=N Dq -k	
	Mary::Dq -k	John::Dq -k	(names, requiring case too)
	is::=Di V -v	ϵ::=V +k =Dq v	(verb phrase split into V and v)
	not::=v Neg		(negation selects v)
	ϵ::=v +v +k I	ϵ::=Neg +v +k I	(inflection selects+licenses v,assigns subj case)

structure building rules $\mathcal{F} = \{merge, move\}$**, fixed for all grammars.**

semantics: any $\mu : \Gamma(G) \to (E, 2)$ defined as follows (where x and y are any distinct feature occurrences):

$\mu(n :: \text{N}) \subseteq E, \quad \mu(n :: \text{Dq} - \text{k}) \in E$

$\mu(some :: \text{=N Dq -k}) = \textsc{some}, \quad \mu(every :: \text{=N Dq -k}) = \textsc{every}, \quad \mu(no :: \text{=N Dq -k}) = \textsc{no}$

$\mu(\text{all other lexical elements}) = \text{id}$, the identity function

when B_0 has no -x: $\mu(merge((A_0, \ldots, A_i), (B_0, \ldots, B_j))) = (A, A_1, \ldots, A_i, B_1, \ldots, B_j)$

where: $A = \begin{cases} \mu(A_0)(\mu(B_0)) & \text{if defined} \\ \mu(B_0)(\mu(A_0)) & \text{otherwise} \end{cases}$

when B_0 has a -x: $\mu(merge((A_0, \ldots, A_i), (B_0, \ldots, B_j))) = (\mu(A_0), \ldots, \mu(A_i), \mu(B_0), \ldots, \mu(B_j))$

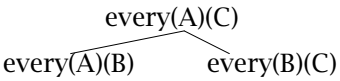
when A_i has no -y: $\mu(move(A_0[\text{+x}], \ldots, A_i[\text{-x}], \ldots, A_n)) = (A, \mu(A_1), \ldots, \mu(A_{i-1}), \mu(A_{i+1}), \ldots, \mu(A_n))$

where: $A = \begin{cases} \mu(A_0)(\mu(A_i)) & \text{if defined} \\ \mu(A_i)(\mu(A_0)) & \text{otherwise} \end{cases}$

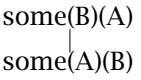
when A_i has a -y: $\mu(move(A_0[\text{+x}], \ldots, A_i[\text{-x}], \ldots, A_n)) = (\mu(A_0), \ldots, \mu(A_n))$

Inference: i. *(every reflexive)* every(A)(A)

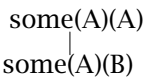
ii. *(every trans)* every is transitive.



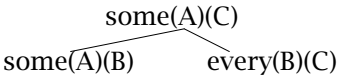
iii. *(some perm)* The order of the nouns in the subject and object does not matter,



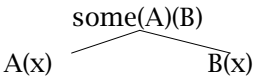
iv. *(some quasi-reflexive)*



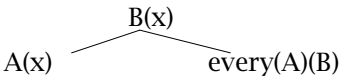
v. *(some2 increasing)*



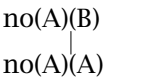
vi. *(name some)*



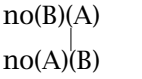
vii. *(name every)*



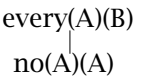
viii. *(no)*



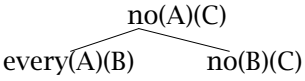
ix. *(no perm)* The order of the nouns in the subject and object does not matter,



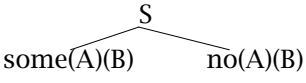
x. *(no to every)*



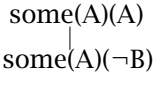
xi. *(every to no)*



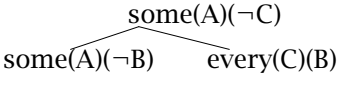
xii. *(contradiction)* For any sentence S:



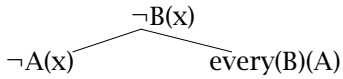
xiii. *(some quasi-reflexive II)*



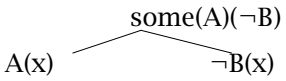
xiv. *(some-not2 decreasing)*



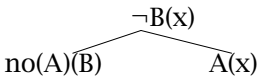
xv. *(name-not2 decreasing)*



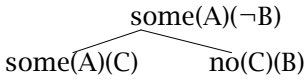
xvi. *(name not to some not)*



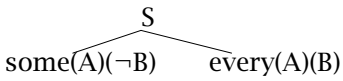
xvii. *(no to name not)*



xviii. *(some no to some not)*



xix. *(contradiction II)* For any sentence S:



19.2 Assessing the syllogistic fragment: linguistically/psychologically plausible?

(1) **Have we got the meaning of *not* right, or even close, in our semantics?** not at all obvious!

(2) **Do any of our inference rules stand out as implausible, from lx or psych perspective?** yes!

(3) **Challenges for (1):** you don't have to be a specialist in metaphysics to notice problems

a. presupposition failure: "The student with 3 eyes left early"
or Strawson's famous example: "He neither cares nor doesn't care; he's dead"
or Carlsons': "Friday is not in bed; it is a date"

b. vague properties: "This is a chair"

c. (practically or theoretically) undecidable properties: "P≠NP"

d. law of contradiction: $\neg(p \wedge \neg p)$

e. law of excluded middle: $p \vee \neg p$

f. law of double negation: $p \equiv \neg\neg p$

(4) **Challenges for (2):**

a. the rules for contradictions are absurd! Mathematicians reason that way, but only in certain specific settings, namely, in the setting of *reductio ad absurdum* arguments. And some mathematicians avoid *reductio* arguments altogether…

b. Our rules miss scope effects:

everyone didn't come

c. our rules miss tense effects – also very important and we did not explore them:

Socrates was a newborn; every newborn is an illiterate; Socrates is/was an illiterate

d. our rules miss contextual and discourse effects – emphasized by Geurts, but we did not dig into such things yet:

there's no place like home
I went to the party but no one was there
Everybody loves my baby, but my baby loves only me

The first issue a is much more serious than missing aspects like b,c,d,…

19.3 Negation across languages

[18] (5) First we should note that, unlike logic, few languages have explicitly propositional negation. This is noted by the philosopher Geach for example:

> *Propositional negation was as foreign to ordinary Greek as to English, and [Aristotle] never attained a distinct conception of it…In ordinary language, it is rather rare to negate a statement by prefixing a sign of negation that governs the whole statement; negation is almost always applied primarily to some part of a statement, though it often has the effect of negating the statement as a whole*

[17] (6) In English, the following types of negation are sometimes distinguished:

explicitly negative free morphemes: not, no, except, but, never, rarely,…

explicitly negative bound morphemes: un-, in-, a-, never-,…

implicitly negative morphemes: deny, doubt, fail,…

purely definitionally negative morphemes: bachelor, kill,…

Various kinds of reaction time studies show that sentences containing any of the first 3 types are more difficult to process.

We noticed in lecture #5 (in connection with the possibility of defining most lexical items in terms of more basic concepts) that purely definitional negatives do not really act like the others: it is as if they are not really negative at all.

[33] (7) In Japanese:

auxiliary *nai*: not

adjectival *nai*: nonexistent

iya: not desired

iiya: contrastive

McNeill and McNeill claim that children understand the adjectival type of use "nonexistent" or "not here" first, then generalizing to the more abstract "not"

[46, 25] (8) Finnish has no negative quantifiers, but has negative adverbs and normal verbal negation appears as an agreeing auxiliary verb:

a. Liisa ostaa kirjan
Lissa-NOM buys-sg.3 book-ACC
'Lisa buys a/the book'

b. Liisa ei osta kirjaa
Lissa-NOM neg-sg.3 buy book-PART
'Lisa does not buy a/the book'

The negation can also be preposed to sentence-initial position:

a. Ei Pekalla ole lapsia!
neg-sg.3 Pekka-ABL is children-PART
'Pekka doesn't have any children!'

b. En minä viitsi riskeerata mitään – viela
neg-sg-1 I-NOM feel-like risk anything-PART yet
'I don't feel like taking any risks – yet'

(9) In many languages negation normally involves two morphemes, as in Quechua's constructions with *mana…-chu*:[1] [47, p19]

a. mana maqasha-chu Hwan-ta-qa
not hit-sg.3-NEG Juan-ACC-TOP
'he didn't hit Juan'

b. mana Pillku-man aywa-q-chu kay
not Pillku-GOAL go-SUBJ-NEG be
'I didn't habitually go to Pillku'

(10) Based on a survey of 345 languages, Dryer reports: [13]

a. The most common position for negatives in SVO languages is between the subject and the verb, yielding S¬VO.
But this is only a tendency: out of 67 languages/15 families this position is found in 47 languages/13 families.

b. In SOV languages the negation is commonly placed before or after the verb.
SO¬V (found in 39 languages/15 families out of 117 languages/23 families) and
SOV¬ (found in 64 languages/18 families out of 117 languages/23 families).

c. If a language is verb-initial, then the negative will precede the verb.
This universal is a strong statistical universal, true for 52 of the 53 verb-initial languages in the 345 languages of Dryer's sample.
Another study by Dahl confirms this: "... all verb-initial and verb-second languages have preverbal placement [of negation]…The only examples of sentence-initial Neg placement are verb-initial languages." In Dahl's sample of 240 languages there are only two non-verb-initial languages, Zuñi and Takelma (both isolates, both SOV), which place the Neg particle initially. [11]

d. If there is double negation within a language, then the negatives normally precede and follow the verb.
In Dryer's 345-language sample, 20 languages exhibit double negation, with negatives normally both preceding and following the verb.

(11) In many languages the particular expressions of negation vary depending on the mood of the sentence (e.g. ancient Greek and Latin) or on the scope of the negation, as in the Nilo-Saharan language Turkana: [49, 24, 12]

a. mɛ̀ɛ̀rɛ̀ a-yɔ̀ŋ ɛ-ka-pɪl-a-ḭ̀
NEG I witch
'I am not a witch'

b. pɛ̀-ɛ̀-a-ra-ì ŋesì ɛ-ka-ɪtà-tam-à-nɪ̰
NEG-3-PAST-be-ASP he teacher
'He was not a teacher' (VP scope)

c. ŋ-e-los-een-è-tè ŋi-kìlyòk a-pey-ò ɛ̀-mamʊ̰̀ e-kìcoloŋ
NEG-3-go-HAB-ASP-PL men visit 3-lack headrest
'Men don't go on a visit without a headrest' (adverbial scope)

And in the Dravidian language Kannada, negation varies between declarative, interrogative and modal moods: [4]

[1] Kahrel questions whether *-chu* is actually a negative particle, on the grounds that it is common for *-chu* to occur by itself in questions, so maybe *-chu* indicates 'non-factualness' [24, §4.2.1]. But Weber points out some cases where leaving *mana* out yields a structure interpreted not as a question but as a negative [47, p19].

a. yār-ū maneyoḷage baral-illa
someone-ENCL house-into came-NEG
'no-one came into the house'

b. nīnu allige hōga-bēḍa
you there go-NEG
'don't go there'

c. nīnu allige hōga-bāradu
you there go-MODAL-NEG
'you should not go there'

Cf. Cree and related American languages for even more elaborate systems. [44]

(12) Aquinas (1258) notices the most basic tendency that we see in all these cases:

> *With respect to vocal sound, affirmative enunciation is prior to negative because it is simpler, for the negative enunciation adds a negative particle to the affirmative.*

Much earlier, in Gautama's (300) *Nyāya-Sūtra*, in Sūtra ii.2.8, noted that absence is marked,

> *…because when there are certain objects marked, the unmarked objects by the non-existence of the mark.*

And Dharmakīrti says:

> *There can be no affirmation which does not exclude the other; nor can there be a negation of that which cannot be affirmed.*

The linguist Roman Jakobson (1939) picks up this ancient idea, noting that in language one member of an opposed pair is overtly marked, while the other is typically signaled by the absence of an overt signal.

This suggests that negation must be learned at some point after some corresponding positive expressions are understood. (we'll see some accounts below that need not accept this)

19.4 Learning *not* (and other negative elements)

(13) **Learning the semantic value from some range of options.**

For the determiners, we considered what kinds of semantic values could be denoted.

In English, we gave *not* its own category: is this right? In other languages, is verbal negation a category unto itself, with no other words having the same distribution?

Putting these questions aside, we could compare the verbal negation with other expressions that map properties to properties: adverbs, etc.

I don't know of any serious proposal, or even any exploration along these lines.

(14) **Learning meaning from identification of inferential relations?**

i. There is a mathematical tradition that aims to square our ideas about the meanings of expressions with our use of them: Gentzen, Hertz, Prawitz, Geach, Dummett,… [15, 14, 36, 37, 34, 38]

In the first place, much of mathematics can be done without *reductio* arguments, and without assuming that $\neg\neg p \equiv p$. This is the program of "intuitionism."

Furthermore, as Dummett and some others in this tradition have noted: This idea should also be relevant to any reasonable conception about how expressions are learned.

ii. John Harris proves that in any classical logic, if you accept this deduction for all sentences S1 and S2: [22]

$$\begin{array}{c} S1 \\ | \\ S1 \wedge S2 \end{array}$$

and then you learn that some new expression $\diamond$ allows the similar inference

$$\begin{array}{c} S1 \\ | \\ S1 \diamond S2 \end{array}$$

Then you will be able to prove that $S1 \wedge S2 \equiv S1 \diamond S2$

And he explores how far the similar argument can go for different ideas about negation, depending on which inference rules are accepted.

iii. Clark ridicules this kind of approach to learning what quantifiers mean: [10]

> On [Geach's] theory, quantified expressions have no reference. Their meaning is acquired from the part they play in inference. [footnote: This point of view has won wide acceptance in generative grammar; see for example the discussion of Quantifier Raising in May (1985) or Chomsky (1986).] ...Presumably, the Geachian learner observes its caretakers engaging in various sorts of syllogistic reasoning, and, then, associates each quantifier with the proper syllogism. This seems a rather unlikely scenario; it presupposes that, once a learner detects a quantifier, it must wait until it observes someone in its environment producing an inferential pattern, a behavior that has a rather low probability of overtly manifesting itself in the normal course of events.

What we want to say here is that the learner need not wait for such "overt manifestations" of inference! We should consider more likely sorts of evidence, like this:

> If the learner hears S in a situation where the learner knows it would be appropriate to say S1 and S2, this is evidence that {S1,S2}⊢S.
> If the learner says S1 and S2, and the audience indicates that S has been expressed, this is evidence that {S1,S2}⊢S.

But obviously, these are too unconstrained! We expect regular, structural relations between the elements of {S1,S2} and the conclusion S. The idea that there might be regularities in such relations is suggested by perspectives like Shieber's [40, 19]

iv. **Shieber: inference as tree transduction**

Only the contradiction rules, used in *reductio* arguments, introduce conclusions with structures that are unrelated to their arguments. Let's dump them!

We can extend the transduction idea to take two sentences as input in various ways: one simple way is just to combine the two sentences into a conjunction.

The (some quasi-reflexive) and (some quasi-reflexive II) rules are non-linear, but they seems suspect from a psychological point of view too.

Some other rules are non-linear because they allow the introduction of arbitrary B, which when there are relative clauses or other modifiers, could be arbitrarily large: (no), (no to every). I am inclined to think these too are suspect.

Notice that the permutation rules involve what Shieber calls "rotations," definable with linear, complete bimorphisms: (some perm), (no perm).

The rules (every trans), (some2 increasing), (every to no), (some-not2 decreasing), basically involve dropping an intermediate element.

And rules like (name some), (name every), rearrange structure in a fixed way.

Many interesting open questions here!

(15) **Learning meaning from syntax? syntactic bootstrapping**

The Guasti text does not give much attention to the acquisition of negation, but a syntactic perspective has been recently considered in a UCLA report [20].

Some children go through a stage where they say things like this:

No the sun shining
No sit there

Then later we sometimes get things like

I no want the envelope
He not little, he big

This led some psychologists to propose that at first, negation is placed in sentence-initial position, and a structural adjustment follows at a later stage.

[45] But looking carefully at the context of these utterances, Stromswold argues that most cases of sentence-initial negation are instances of denying what was previously said or suggested by context, followed by a positive sentence, as in adult discourses like

Is it raining? No, the sun is shining.

Gilkerson et al call this use of negation "anaphoric."

Gilkerson et al. explore children's understanding of negation at stages prior to their production of multiword sentences using a preferential looking paradigm (which we discussed before, in Lecture #6). Like Guasti, they say:

> We take Continuity [the position the child languages fall within UG parameters] and [Early Morphsyntactic Convergence] as null hypotheses because they postulate no difference between the two groups, adults and children.

They showed images of a girl sitting and of a girl sleeping, and compared the child's looking response to *the girl's not sleeping* and *Is the girl sitting? No, she's sleeping.* Their results suggests that, contrary to Stromswold's suggestion, children do not always interpret sentence-initial negation as anaphoric.

(16) **learning meaning from distribution, in context?**

i. We might suppose that the language learner already uses negation in reasoning about things, and that the learner's task is to associate the word *not* with that concept.

Various versions of this idea have been proposed by Augustine, Pinker, Fodor, Gleitman, Siskind, Brent, and many others

ii. Snedeker&Gleitman: To learn that cat is the English-language word for the concept 'cat,' the child need only note that cats are the objects most systematically present in scenes wherein the sound /kat/ is uttered (just as proposed by Augustine (398); Locke (1690); Pinker (1984); and many other commentators).

[42, 39, 35, 41, 27]

Given word-sequence concept-sequence pairs $(w_1 \ldots w_i, s_1 \ldots s_j)$, construct m: words → concepts as follows:

for each w_k,

if $w_k \in \mathrm{dom}(m)$, $m(w_k) \leftarrow m(w_k) \cap (s_1, \ldots, s_j)$

else $m(w_k) \leftarrow (s_1, \ldots, s_j)$

Convergence to perfect, correct match intractable! Prospects in actual cases appear dim. Maybe helped by...

[2, 26, 31, 48] 'whole object bias': assume new words name objects as wholes

[30] 'taxonomic assumption': assume new terms extend to other similar objects

[29] 'mutual exclusivity': assume each object has one and only one name

[21] 'novel name-nameless category': assume novel name maps to an unnamed object

[43, 16, 7] 'short utterances': pause-separated short phrases and isolated words in the data

[5, 6, 3] 'learner-directed speech': data sometimes directly relevant to current focus of learner's attention

iii. Snedeker&Gleitman: ...this situational evidence taken alone may be insufficient for the mapping of most of the vocabulary...We will demonstrate that the vocabulary could become rich and diversified owing to a succession of bootstrapping operations grounded in the prior acquisition of concrete nominals... [42, 28]

References for Lecture 19

[1] AQUINAS, T. Aristotle on interpretation: Commentary by St. Thomas and Cajetan. *Mediaeval Philosophical Texts in Translation 11* (1258). 1962 translation by Jean T. Oesterle.

[2] BALDWIN, D. Priorities in children's expectations about object label reference: form over color. *Child Development 60* (1989), 1291–1306.

[3] BALDWIN, D., MARKMAN, E., BILL, B., DESJARDINS, N., IRMIN, J., AND TIDBALL, G. Infants' reliance on a social criterion for establishing word-object relations. *Child Development 67* (1996), 3135–3153.

[4] BHATIA, T. K. *Negation in South Asian Languages*. Indian Institute of Language Studies, New Delhi, 1993.

[5] BLOOM, L. *The Transition from Infancy to Language: Acquiring the Power of Expression*. Cambridge University Press, Cambridge, 1993.

[6] BLOOM, L. The intentionality model of word learning: How to learn a word, any word. In *Becoming a Word Learner: The Debate on Lexical Acquisition*, R. Golinkoff, K. Hirsh-Pasek, L. Bloom, L. Smith, A. Woodward, N. Akhtar, M. Tomasello, and G. Hollich, Eds. Oxford University Press, NY, 2000.

[7] BRENT, M. R., AND SISKIND, J. M. The role of exposure to isolated words in early vocabulary development. *Cognition 81* (2001), B33–B34.

[8] CHOMSKY, N. *Knowledge of Language*. Praeger, NY, 1986.

[9] CLARK, E. V. Meaning and concepts. In *Handbook of Child Psychology*, J. Flavell and E. Markman, Eds. Wiley, NY, 1983, pp. 787–840.

[10] CLARK, R. Learning first order quantifier denotations: An essay in semantic learnability. Tech. rep., Institute for Research in Cognitive Science, University of Pennsylvania, 1996.

[11] DAHL, O. Typology of sentence negation. *Linguistics 17* (1979), 79–106.

[12] DIMMENDAAL, G. *The Turkana language*. PhD thesis, University of Leiden, 1982.

[13] DRYER, M. S. Universals of negative position. In *Studies in Syntactic Typology*, M. Hammond, E. Moravcsik, and J. Wirth, Eds. Benjamins, Amsterdam, 1988, pp. 93–124.

[14] DUMMETT, M. *Frege: Philosophy of Language, 2nd Edition*. Duckworth, London, 1981.

[15] DUMMETT, M. *The Logical Basis of Metaphysics*. Duckworth, London, 1991.

[16] FERNALD, A., TAESCHNER, T., DUNN, J., PAPOUSEK, M., BOYSSON-BARDIES, B., AND FUKUI, I. A cross-language study of prosodic modifications in mothers' and fathers' speech to preverbal infants. *Journal of Child Language 16* (1989), 477–501.

[17] FODOR, J., FODOR, J., AND GARRETT, M. The psychological unreality of semantic representations. *Linguistic Inquiry 6* (1975), 515–532.

[18] GEACH, P. *Logic Matters*. University of California Press, Los Angeles, 1972.

[19] GÉCSEG, F., AND STEINBY, M. Tree languages. In *Handbook of Formal Languages, Volume 3: Beyond Words*, G. Rozenberg and A. Salomaa, Eds. Springer, NY, 1997, pp. 1–68.

[20] GILKERSON, J., HYAMS, N., AND CURTISS, S. On the scope of negation: More evidence for early parameter setting. In *Proceedings of the Generative Approaches to Language Acquisition (GALA)* (2003).

[21] GOLINKOFF, R., MERVIS, C., AND HIRSH-PASEK, K. Early object labels: The case for a developmental lexical principles framework. *Journal of Child Language 21* (1990), 125–155.

[22] HARRIS, J. What's so logical about the 'logical' axioms? *Studia Logica 41* (1982), 159–171.

[23] JAKOBSON, R. Signe zéro. In *Mélanges de linguistique offerts á Charles Bally sous les auspices de la Faculté des lettres de l'Université de Genêve par des collègues, des confrères, des disciples reconnaissants*. Georg et Cie, Genève, 1939.

[24] KAHREL, P. J. *Aspects of Negation*. PhD thesis, University of Amsterdam, Amsterdam, 1996.

[25] KAISER, E. Negation and the left periphery in Finnish. *Lingua 115* (2005).

[26] KOBAYASHI, H. How 2-year-olds learn novel part names of unfamiliar objects. *Cognition 68*, 2 (1998), B41–B51.

[27] KOBELE, G. M., RIGGLE, J., COLLIER, T., LEE, Y., LIN, Y., YAO, Y., TAYLOR, C., AND STABLER, E. Grounding as learning. In *Language Evolution and Computation Workshop, ESSLLI'03* (2003).

[28] LIDZ, J., GLEITMAN, H., AND GLEITMAN, L. R. Kidz in the 'hood: Syntactic bootstrapping and the mental lexicon. In *Weaving a Lexicon*, D. Hall and S. Waxman, Eds. MIT Press, Cambridge, Massachusetts, 2004, pp. 603–636.

[29] MARKMAN, E. M. *Categorization and Naming in Children: Problems of Induction*. MIT Press, Cambridge, Massachussetts, 1989.

[30] MARKMAN, E. M., AND HUTCHINSON, J. Children's sensitivity to constraints on word meaning: Taxonomic versus thematic relations. *Cognitive Psychology 16* (1984), 1–27.

[31] MARKMAN, E. M., AND WACHTEL, G. F. The role of semantic and syntactic constraints in the memorization of English sentences. *Cognitive Psychology 20*, 1 (1988), 121–157.

[32] MAY, R. *Logical Form: Its Structure and Derivation*. MIT Press, Cambridge, Massachusetts, 1985.

[33] MCNEILL, D., AND MCNEILL, N. B. What does a child mean when a child says 'no'? In *Proceedings of the Conference on Language and Language Behavior* (NY, 1968), E. M. Zale, Ed., Appleton-Century-Crofts, pp. 51–62.

[34] PAGIN, P. Bivalence: meaning theory vs. metaphysics. *Theoria; a Swedish journal of philosophy and psychology 64* (1998), 157–186.

[35] PINKER, S. *Language Learnability and Language Development*. Harvard University Press, Cambridge, Massachusetts, 1984.

[36] PRAWITZ, D. Ideas and results in proof theory. In *Proceedings of the Second Scandinavian Logic Symposium*, J. Fenstad, Ed. North-Holland, Amsterdam, 1971, pp. 235–307. Partially reprinted as "Gentzen's analysis of first order proofs," in R.I.G. Hughes, *A Philosophical Companion to First Order Logic*, Hackett: Indianapolis, 1993.

[37] PRAWITZ, D. Comment on Peter Pagin's paper. *Theoria; a Swedish journal of philosophy and psychology 64* (1998), 304–318.

[38] PRAWITZ, D. Meaning approached via proofs. *Synthese , to appear* (2005).

[39] REGIER, T., CORRIGAN, B., CABASAN, R., WOODWARD, A., GASSER, M., AND SMITH, L. The emergence of words. In *Proceedings of the Cognitive Science Society* (2001).

[40] SHIEBER, S. Towards a universal framework for tree transduction. Seminar, Center for Language and Speech Processing, Johns Hopkins University, November, 2004.

[41] SISKIND, J. M. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition 61* (1996), 39–91.

[42] SNEDEKER, J., AND GLEITMAN, L. R. Why it is hard to label our concepts. In *Weaving a Lexicon*, D. Hall and S. Waxman, Eds. MIT Press, Cambridge, Massachusetts, 2004, pp. 257–293.

[43] SNOW, C. The development of conversation between mothers and babies. *Journal of Child Language 4*, 1 (1977), 1–22.

[44] STARK, D. Negation: An insight into the structure of main clauses in Woods Cree. *Journal of the Atlantic Provinces Linguistic Association 9* (1987), 21–42.

[45] STROMSWOLD, K. The acquisition of inversion and negation in English: a reply to Deprez and Pierce. Manuscript, Rutgers University, 1997.

[46] VILKUNA, M. *Free Word Order in Finnish: Its Syntax and Discourse Functions*. Suomalaisen Kirjallisuuden Seura, Helsinki, 1989.

[47] WEBER, D. J. *A Grammar of Huallaga (Huánuco) Quechua*. University of California Press, Los Angeles, 1989.

[48] WOODWARD, A. L. *The Role of the Whole Object Assumption in Early Word Learning.* PhD thesis, Stanford University, 1992.

[49] ZWICKY, A., AND PULLUM, G. Cliticization and English *n't. Language 59* (1983), 502–513.

20 Some next steps

20.1 Steps to a fragment with transitive verbs, questions, briefly

(1) The extension from our language to intransitive and transitive sentences not difficult:

Mary is a student	tree	student(Mary)
Every student is a human	tree	every(student)(human)
Mary sings	tree	sings(Mary)
Mary likes Sue	tree	likes(Sue)(Mary)
Mary likes Sam	tree	likes(Sam)(Mary)

Call the set of sentences above C.

(2) Yes-no questions: a simple approach

i. Syntax: invert the subject and auxiliary verb.

ii. Semantics: Unlike declaratives which denote truth values and typically assert truth, YNQs denote truth values and typically are requests for the audience to assert truth or falsity.

Is Mary is a student?	≡	student(Mary)
Is every student a human?	≡	every(student)(human)
Does Mary sing	≡	sings(Mary)
Does Mary like Sue?	≡	likes(Sue)(Mary)
Does Mary like Sam?	≡	likes(Sam)(Mary)

iii. Inference/pragmatics:

Is Mary is a student?	tree	C⊢student(Mary)?	or C⊢student(Mary)?	or C∪{¬student(Mary)}⊢ □
Is every student a human?	tree	C⊢every(student)(human)?	…	…
Does Mary sing	tree	C⊢sings(Mary)?	…	…
Does Mary like Sue?	tree	C⊢likes(Sue)(Mary)?	…	…
Does Mary like Sam?	tree	C⊢likes(Sam)(Mary)?	…	…

Note that if C⊬student(Mary), this does <u>not</u> mean that C⊨ ¬student(Mary)!

(3) Wh-questions: a simple approach

i. Syntax: fronting wh-phrase, sometimes triggering inversion

ii. Semantics:

Who is a student?	tree	{x| student(x)}
Who sings	tree	{x| sing(x)}
Who likes Sue?	tree	{x| likes(Sue)(x)}
Who does Mary like?	tree	{x| likes(x)(Mary)}

Questions like "What is every student?" require special treatment…

Conservative

iii. Inference: forward chaining to get complete list of x such that student(x) not feasible!

Who is a student?	derive minimal models refuting C∪{nothing is a student}
Who sings	minimal models refuting C∪{nothing sings}
Who likes Sue?	minimal models refuting C∪{nothing likes Sue}
Who does Mary like?	minimal models refuting C∪{Mary likes nothing}

20.2 Idioms and learning as 'concept labeling'

(4) Recall these simple ideas:

a. Snedeker&Gleitman: To learn that cat is the English-language word for the concept 'cat,' the child need only note that cats are the objects most systematically present in scenes wherein the sound /kat/ is uttered (just as proposed by Augustine (398); Locke (1690); Pinker (1984); and many other commentators).

[31, 27, 25, 30, 16]

…this situational evidence taken alone may be insufficient for the mapping of most of the vocabulary…We will demonstrate that the vocabulary could become rich and diversified owing to a succession of bootstrapping operations grounded in the prior acquisition of concrete nominals…

[2, 20, 4, 3, 19, 14, 7]

b. number of nouns in an utterance → clues about verb subcategorization

[31, p260]

c. Lexical acquisition as word-concept mapping: One cannot acquire productive use of a term that expresses a concept that one cannot entertain.

d.

Given word-sequence concept-sequence pairs $(w_1 \dots w_i, s_1 \dots s_j)$, construct m: words → concepts as follows:

for each w_k,

if $w_k \in \mathrm{dom}(m)$, $m(w_k) \leftarrow m(w_k) \cap (s_1, \dots, s_j)$

else $m(w_k) \leftarrow (s_1, \dots, s_j)$

(5) In this last version of the Augustinian idea, convergence maps 'words' to meanings…
Then: how to handle semantic-atom/syntactic-atom mismatches: verb-prt, idioms, etc?

(6) Natural languages have many phrases with idiosyncratic interpretations:

verb-particle constructions: take up/down/in/out/back/over, turn up/down/over/in/out/around/off, hold up/down/off/out/over, head up/out/for, hand out/in/over, give up/in/out, play out/up/down, drum up/out, work up/out/over, roll out/over, find out, lose out, bottom out, sell out/off, stir up, write off, wake up, …

'fixed phrases', idiomatic compounds, etc: by and large, in short, every which way, do away with, spick and span, break a leg, monkey wrench, sunflower, traffic light, deadline, …

'non-compositional phrasal idioms': kick the bucket, pop the question, chew the fat, hit the sauce, go out on a limb, get hot under the collar, make the scene, bark up the wrong tree, …

'compositional phrasal idioms': let the cat out of the bag, strike paydirt, spill the beans, pull strings, play hooky, sweep X under the rug, swallow one's pride, keep X under one's hat, pull X's leg, …

(7) Some linguists speculate that the number of idiosyncratic phrases is of the same order as the number of words; others think the number may be much larger.

[12, 29, 28]

(8) Do they violate compositionality? Many phrasal idioms have meaningful components.

[24]

internal modification: leave no legal stone unturned, kick the filthy habit, get the job by pulling strings that weren't available to anyone else. touch a nerve that I didn't know even existed, …

topicalization, anaphora: Those strings, he wouldn't pull even for you,
His closets, you might find skeletons in,
Those windmills, not even he would tilt at,
Once someone lets the cat out of the bag, it's out of the bag for good,
Close tabs were kept on Jane Fonda, but none were kept on Venessa Redgrave, …

families of idioms: hit the hay/sack, pack a punch/wallop, throw X to the dogs/lions/wolves, keep/lose/blow one's cool, step/tread on X's toes, lay/put one's cards on the table, …

(9) learning idioms: learners look for compositional analyses [9, 10]

(10) familiar phrases (metaphors etc): cry one's eyes out, lose one's balance, flocking to see it, had it up to here, at your mercy, clear as day, pop quiz, power tie, running rampant, sitting pretty,…

(11) Define: [15, 33, 11, 34]

grammar:	(Lexicon,F)
language:	L = closure(Lexicon,F), with derivations Γ
(partial) semantics:	$\mu : \Gamma \to M$
factor away M:	$a \equiv_\mu b$ iff $a, b \in \mathrm{dom}(\mu)$ and $\mu(a) = \mu(b)$.

μ is compositional iff $f(a_1, \ldots, a_n) \equiv_\mu f(b_1, \ldots, b_n)$ whenever $a_i \equiv_\mu b_i$ (all $f \in F$)
ν extends μ iff ν restricted to $\mathrm{dom}(\mu)$ is exactly μ

Example 1: Consider a grammar $(\{a, b\}, \{f\})$ where

$$a \equiv_\mu b$$
$$f(a) \not\equiv_\mu f(b)$$

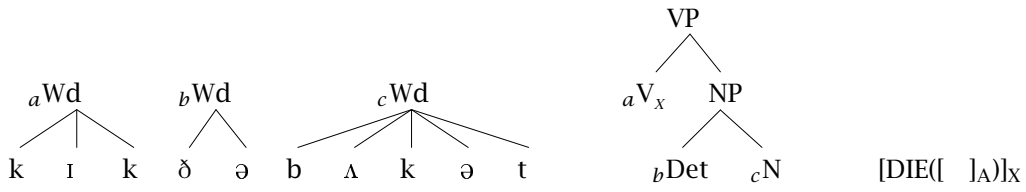
Not compositional.

Example 2: Consider a grammar $(\{a, b, c\}, \{f, g\})$ where

$$a \equiv_\mu b$$
$$f(b) \equiv_\mu c$$
$$g(f(a)) \not\equiv_\mu g(c)$$

Partial and compositional, with no total compositional extension.

(12a) complex lexical elements. E.g. Jackendoff – 'listed in the lexicon as an ordinary VP' [12, 13, 1, 28]



Does the idiom *kick the bucket* have the familiar syntactic parts V, Det, N?

Yes: These parts are assembled by the usual syntactic rules, but the meanings of these parts do not determine the value of the complex. The complex itself gets the specified interpretation.
(Not compositional. & Why would such things exist in the language?)

Yes: These parts are assembled by the usual syntactic rules, but these parts are uninterpreted elements (homophonous with interpreted ones). Only the complex is interpreted.
(Compositional. & Why would such things exist in the language?)

No: The 'complex' is lexical; it does not have syntactic parts that are assembled by the usual rules. Instead it is just looked up, and is assigned the meaning lexically.

(Hard to make sense of (lexical parts?). Predicts discontinuity with collocations. & Why?)

[34] (12b) special new rules for each idiomatic construction. E.g. Westerståhl

(12c) the parts of the idiom as new atoms.

[12, 13] Jackendoff: A possibility...is to more or less simulate the listing of *kick the bucket* with monomorphemic lexical entries, by stipulating that *bucket* has a special interpretation in the context of *kick* and vice versa...-Given a body of fixed expressions as numerous as the single words, such clumsy encoding should be suspect. Indeed, because of the complications of such 'contextual specification,' no one (to my knowledge) has really stated the details. In particular, it is (to me at least) totally unclear how to 'contextually specify' idioms of more than two morphemes. ...I conclude that this alternative rapidly collapses of its own weight.

[21, 23] NB: *kick the bucket* does not mean *die*

Hermione was dying for weeks / #Hermione was kicking the bucket for weeks

[26] NB: Using L with semantics $\mu : \Gamma \to M$ does not involve computing (or specifying or encoding) μ

It is maybe useful to avoid this confusion with the equivalence relation $\equiv_\mu$, where μ is factored out.

[8] Frege: It is enough if the sentence as a whole has meaning; it is this that confers on its parts also their content.

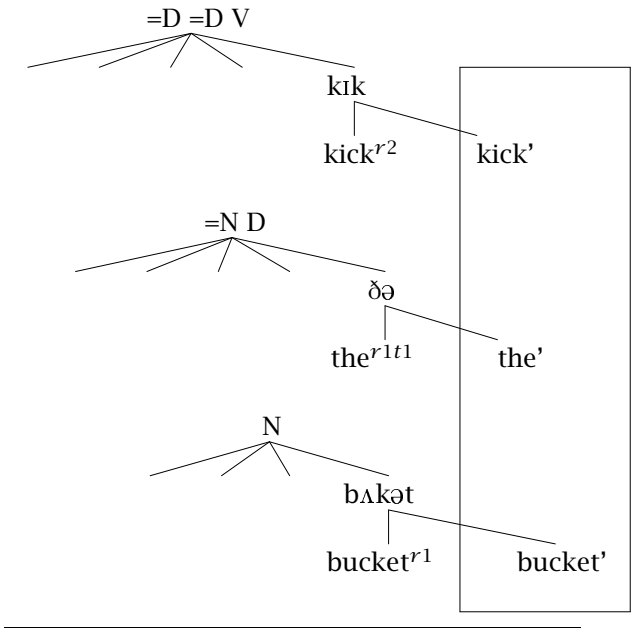
Suppose a language learner has acquired a fragment of English with a total, compositional semantics $\equiv_\mu$, and comes across the new expression *kick the bucket*, noticing that it has an unusual meaning. Is there a natural compositional extension $\equiv$ that can accommodate it? Yes.[1]

the fregean cover: $a \equiv b$ iff whenever $f(\ldots,a,\ldots) \in \mathrm{dom}(\mu)$ then $f(\ldots,b,\ldots) \in \mathrm{dom}(\mu)$, and for every $f(\ldots,a,\ldots) \in \mathrm{dom}(\mu)$, $f(\ldots,a,\ldots) \equiv_\mu f(\ldots,b,\ldots)$

(4b') Given word-sequence concept-sequence pairs $(w_1 \ldots w_i, s_1 \ldots s_j)$,
if all $w_i \in \mathrm{dom(m)}$ but $(\mathrm{m}(w_1) \cup \ldots \cup \mathrm{m}(w_i)) \cap \{s_1 \ldots s_j\} = \emptyset$,
assume these are new senses $w'_1 \ldots w'_i$ and interpret with a fregean cover.

(13) For illustration, we adapt Jackendoff's simplified example, regarding the lexicon as a network of syn-phon-sem associations. With (4b'), we get just the boxed elements in the lexicon, and a new belief:

[1] Remembering example 2 we need to be careful here: Hodges (2001) proves this compositional extension is available if (i) $\equiv_\mu$ is compositional, (ii) if $a \equiv_\mu b$, then any occurrence of a in any $d \in \mathrm{dom}(\mu)$ can be replaced by b to get another $d' \in \mathrm{dom}(\mu)$, and (iii) everything in $\mathrm{dom}(\equiv)$ is a constituent of some element of $\mathrm{dom}(\equiv_\mu)$. Westerståhl (2004) shows that a compositional extension is available whenever $\equiv_\mu$ is compositional and closed under subterms.



bucket=pail
kick<act on
the<some
…

kick'(the'(bucket'))^{r1}=die

(14) Jackendoff's worries addressed: not clumsy, and 'contextual specification' not needed: …in order to specify *let the cat out of the bag* one word at a time, contextual specifications must be provided for *let*, *cat*, *out of*, and *bag*, each of which mentions the others (and what about *the*?). Moreover, the correct configuration must also be specified so that the special interpretation does not apply in, say *let the bag out of the cat*. [12, 13]

(15) Fix 1: the new kɪk/kick' should get some but not all of the semantic and syntactic features of the original.

a. # the bucket was kicked / # the breeze was shot

b. some strings were pulled / the beans were spilled

Rough idea: In the former cases, the meaning of the DP is special enough that the learner is unwilling to give it full status as a regular DP. E.g. in (13): to block passive and topicalization, it suffices to change D in the syntactic specification of ðə/the' to a subcategory that does not participate in these.

(16) Fix 2: dependencies and 'idiom chunks': allow separated constituents in (4b') [32, 17, 6, 5, 18] vs. [28, 12, 24, 22]

a. [the shit] seems to have [hit the fan] / excellent [care] was [take]n of the orphans

b. the [strings] that the coach [pull]ed / the close [tabs] he [keep]s [on] his operation

c. [get X's goat] / [the bottom fell out of X] / [take issue with X]

(17) Why do idioms exist in human language? Answer: the standard learning strategy discovers them! the very strategy for discovering recursive structure, and particularly the semantic values of constituents, readily discovers complexes with constituents about which little is assumed.

References for Lecture 20

[1] ABEILLÉ, A., AND SCHABES, Y. Parsing idioms in lexicalized TAGs. In *Proceedings of the European Association for Computational Linguistics*. 1989, pp. 161–165.

[2] BRENT, M. R. Automatic acquisition of subcategorization frames from untagged text. In *Proceedings of the of the 29th Meeting of Association for Computational Linguistics* (1991), pp. 209–214.

[3] BRISCOE, T., AND CARROLL, J. Automatic extraction of subcategorization from corpora. To appear in ANLP-97, 1997.

[4] CARROLL, G., AND ROOTH, M. Valence induction with a head-lexicalized PCFG. In *Proceedings of the of the 3rd Conference on Empirical Methods in Natural Language Processing, EMNLP'98* (1998).

[5] CHOMSKY, N. *Rules and Representations*. Columbia University Press, NY, 1980.

[6] CHOMSKY, N. *Lectures on Government and Binding*. Foris, Dordrecht, 1981.

[7] FERRER, E. E. Towards a semantic classification of Spanish verbs based on subcategorisation information. In *Proceedings of the of the 42nd Meeting of Association for Computational Linguistics, Student Session* (2004).

[8] FREGE, G. *Die Grundlagen der Arithmetik*. Koebner, Breslau, 1884. J.L. Austin's translation available as *The Foundations of Arithmetic*, Evanston, Illinois: Northwestern University Press, 1980.

[9] GIBBS, R. W. Semantic analyzability in children's understanding of idioms. *Journal of Speech and Hearing Research 35* (1991), 613–620.

[10] GIBBS, R. W. *The Poetics of Mind*. Cambridge University Press, NY, 1994.

[11] HODGES, W. Formal features of compositionality. *Journal of Logic, Language and Information 10* (2001), 7–28.

[12] JACKENDOFF, R. S. *The Architecture of the Language Faculty*. MIT Press, Cambridge, Massachusetts, 1997.

[13] JACKENDOFF, R. S. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. Oxford University Press, NY, 2003.

[14] KAWAHARA, D., AND KUROHASHI, S. Fertilization of case frame dictionary for robust Japanese case analysis. In *Proceedings of the 19th International Conference on Computational Linguistics, COLING'02* (2002).

[15] KEENAN, E. L., AND STABLER, E. P. *Bare Grammar*. CSLI Publications, Stanford, California, 2003.

[16] KOBELE, G. M., RIGGLE, J., COLLIER, T., LEE, Y., LIN, Y., YAO, Y., TAYLOR, C., AND STABLER, E. Grounding as learning. In *Language Evolution and Computation Workshop, ESSLLI'03* (2003).

[17] KOOPMAN, H., AND SPORTICHE, D. The position of subjects. *Lingua 85* (1991), 211–258. Reprinted in Dominique Sportiche, *Partitions and Atoms of Clause Structure: Subjects, agreement, case and clitics*. NY: Routledge.

[18] KOOPMAN, H., SPORTICHE, D., AND STABLER, E. *An Introduction to Syntactic Analysis and Theory*. UCLA Lecture Notes, forthcoming, 2002.

[19] KORHONEN, A. *Subcategorization Acquisition*. PhD thesis, University of Cambridge, Cambridge, 2002.

[20] MANNING, C. D. Automatic acquisition of a large subcategorization dictionary from corpora. In *Proceedings of the of the 31st Meeting of Association for Computational Linguistics* (1993), pp. 235–242.

[21] MARANTZ, A. No escape from syntax: Don't try morphological analysis in the privacy of your own lexicon. In *Proceedings of the 21st Annual Penn Linguistics Colloquium* (University of Pennsylvania, 1997), pp. 201–225.

[22] MCCAWLEY, J. D. The syntax and semantics of English relative clauses. *Language 53* (1981), 99–149.

[23] McGinnis, M. On the systematic aspect of idioms. *Linguistic Inquiry 33*, 4 (2002), 665–672.

[24] Nunberg, G., Wasow, T., and Sag, I. A. Idioms. *Language 70*, 3 (1994), 491–538.

[25] Pinker, S. *Language Learnability and Language Development.* Harvard University Press, Cambridge, Massachusetts, 1984.

[26] Putnam, H. Meaning and reference. *Journal of Philosophy 70*, 19 (1973), 699–711.

[27] Regier, T., Corrigan, B., Cabasan, R., Woodward, A., Gasser, M., and Smith, L. The emergence of words. In *Proceedings of the Cognitive Science Society* (2001).

[28] Riehemann, S. Z. *A Constructional Approach to Idioms and Word Formation.* PhD thesis, Stanford University, 2001.

[29] Sag, I., Baldwin, T., Bond, F., Copestake, A., and Flickinger, D. Multiword expressions. In *Proceedings of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2002)* (2002), pp. 1–15.

[30] Siskind, J. M. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition 61* (1996), 39–91.

[31] Snedeker, J., and Gleitman, L. R. Why it is hard to label our concepts. In *Weaving a Lexicon*, D. Hall and S. Waxman, Eds. MIT Press, Cambridge, Massachusetts, 2004, pp. 257–293.

[32] Sportiche, D. Reconstruction, binding and scope. UCLA, 1999.

[33] Stabler, E. P., and Keenan, E. L. Structural similarity. *Theoretical Computer Science 293* (2003), 345–363.

[34] Westerståhl, D. On the compositional extension problem. *Journal of Philosophical Logic forthcoming* (2004).

Index