

A learnable version of set theoretic merge

Alexander Clark

Department of Computer Science
Royal Holloway, University of London
`alexcl@cs.rhul.ac.uk`

Montreal, September 2011



Classical Chomskyan view

Chomsky, Hauser and Fitch (2005)

Two basic conditions that UG must satisfy are that it (1) accommodate the attainable I-languages, and (2) account for their acquisition.

Chomsky, 1986

To achieve descriptive adequacy it often seems necessary to enrich the system of available devices, whereas to solve our case of Plato's problem we must restrict the system of available devices so that only a few languages or just one are determined by the given data. It is the tension between these two tasks that makes the field an interesting one, in my view.

Where does language come from?

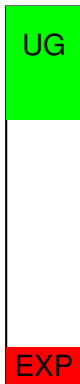
Principles and Parameters





Where does language come from?

MP/Biolinguistics



Why take distributional learning seriously?

Several reasons to take distributional learning seriously:

- Cognitively plausible (Saffran et al. 1996, Mintz, 2002)
- It works in practice: large scale lexical induction (Curran, J. 2003)
- Linguists use it as a constituent structure test (Carnie, A, 2008)
- Historically, PSGs were intended to be the output from distributional learning algorithms.

Chomsky (1968/2006)

“The concept of “phrase structure grammar” was explicitly designed to express the richest system that could reasonable be expected to result from the application of Harris-type procedures to a corpus.”

Rejection was justified at the time

Chomsky, 1965

The only proposals that are explicit enough to support serious study are those that have been developed within taxonomic linguistics. It seems to have been demonstrated beyond reasonable doubt that quite apart from any questions of feasibility, methods of the sort that have been studied in taxonomic linguistics are intrinsically incapable of yielding the systems of grammatical knowledge that must be attributed to the speaker of a language.

Rejection was justified at the time

Now we have precise theories of distributional learning with mathematical proofs that they work.

Many recent results

Context free grammars: Clark and Eyraud (2005,2007), Clark (2006,2010)

Multiple CFGs: Yoshinaka (2009), Yoshinaka and Clark (2010)

Distributional Lattice Grammars: Clark (2010)

Abstract Categorical Grammars: Yoshinaka and Kanazawa (2011)

Context free tree languages: Yoshinaka and Kasprzik (2011)

ITGs Clark (2011)

...

Top down versus bottom up

Kanazawa (2011)

Top down

Chomsky (1956, 1959)

- $N \rightarrow PQ$
- Rewrite N as PQ

Bottom up derivations

Smullyan (1961), Seki et al. (1991), Chomsky (1995), Stabler (1997)

- $N \leftarrow PQ$
- Combine P and Q to make N

Convergence

Partial equivalence

Multiple context free grammars (MCFGs) are partially equivalent to Minimalist Grammars (Stabler) under certain constraints.

- Internal merge: $(u, v) \rightarrow u, v$ rather than $uv \rightarrow u, v$

Learnability results

Recent results show that (some) MCFGs can be learned under a variety of different models.

Assumptions and notations

Classic E-language model

Chomsky (1956, 1959)

- Σ is a set of atomic lexical items
- Σ^* is the set of all finite strings of these items
- If u, v are strings we can write uv for their concatenation
- If X, Y are *sets* of strings we can write XY for their concatenation: the set of all strings formed by concatenating an element of X with an element of Y .
- L is some subset of Σ^* corresponding to the set of grammatical sentences in the language

Input to child learner is unstructured!

Distributional Learning

Complete mutual substitutability: Chomsky (1959)

Let us say that two units A and B are substitutable₁ if there are expressions X and Y such that XAY and XBY are sentences of L.; **substitutable₂ if whenever XAY is a sentence of L then so is XBY and whenever XBY is a sentence of L so is XAY (i.e. A and B are completely mutually substitutable)**. These are the simplest and most basic notions. (footnote: They are discussed by R. Carnap in *The Logical Syntax of Language*, 1934).

Example

Is 'cat' substitutable for 'dog'?

- The cat is over there.
- I want a dog for Christmas.
- I want a Siamese cat for Christmas.
- Put a cat-flap in the door to the kitchen.
- An Alsatian is a breed of dog.
- He continues to dog my footsteps.
- I would rather have a dog than a cat as a pet.

Example

Is 'cat' substitutable for 'dog'?

- The dog is over there.
- I want a cat for Christmas.
- I want a Siamese dog for Christmas.
- Put a dog-flap in the door to the kitchen.
- An Alsatian is a breed of cat.
- He continues to cat my footsteps.
- I would rather have a dog than a dog as a pet.
- I would rather have a cat than a dog as a pet.

Shallow distributional features: privative/monovalued

- The _ is over there.
- I want a _ for Christmas.
- I want a Siamese _ for Christmas.
- Put a _-flap in the door to the kitchen.
- An Alsatian is a breed of _.
- He continues to _ my footsteps.
- I would rather have a _ than a dog as a pet.

Distributional learning

Syntactic Congruence

$[u]$ is the set of all strings that are exactly substitutable for u in a given language

- This is an equivalence class
- It is possible to observe/learn whether two strings are in the same congruence class

Proposition

For all languages and for all strings u, v
 $[uv] \supseteq [u][v]$

Distributional learning

Syntactic Congruence

$[u]$ is the set of all strings that are exactly substitutable for u in a given language

- This is an equivalence class
- It is possible to observe/learn whether two strings are in the same congruence class

Proposition

For all languages and for all strings u, v
 $[uv] \supseteq [u][v]$

Set-theoretic merge

Berwick et al. (2011)

Definition

Two syntactic objects: X and Y
Merge gives $\{X, Y\}$

Labels

What is the label of the merged structure?
What features does it have?

Labels

Standard assumption

Set of features

Syntactic objects/labels

Duality

- Either as a set of strings
- Or as a set of syntactic features

Labels

Alternative view

Sets of strings of lexical items

The set of all strings that can have those features

Syntactic objects/labels

Duality

- Either as a set of strings
- Or as a set of syntactic features

Merge in context free grammars

Context free grammar G

replace arbitrary symbol N

with $Y(N) = \{w \mid N \xRightarrow{*}_G w\}$

e.g. replace NP with $\{\text{the cat, a large car, } \dots\}$

MERGE is concatenation?

Suppose $N \rightarrow PQ$

Then $Y(N) \supseteq Y(P)Y(Q)$.

But this is inclusion not equality.

Congruential approach to distributional learning

Representational assumption

Safe to assume that $Y(N)$ is a union of congruence classes

Minimally $Y(N)$ is just a single congruence class.

Nonterminal $[[u]]$ that generates $[u]$

Rule schemas

- $[[uv]] \rightarrow [[u]][[v]]$
- $[[a]] \rightarrow a$ for all $a \in \Sigma$

Results

Results for congruential learning of CFGs

- 1 Substitutable languages from positive data (Clark and Eyraud, 2005, 2007)
- 2 Congruential languages with queries (Clark, 2010)
- 3 NTS languages with stochastic positive examples (Clark, 2006)
- 4 Congruential languages from positive and negative examples (to appear)

Merge in congruential approach

- Syntactic objects are sets of strings (congruence classes)
- $[u][v]$ is just concatenation.
- MERGE combines $[u]$ and $[v]$ to be $[uv]$

Merge operation

$$[uv] \supseteq [u][v]$$

$$\text{Productions } [[uv]] \rightarrow [[u]][[v]]$$

Weak versus strong learning

Berwick, Chomsky et al. (2011)

This is irrelevant because it is “a mere process of acquiring a (weak generative) capacity to produce just the valid word strings of a language”

What is needed is to learn the “structures a language generates”.

One answer

Learning from pairs of strings and sentence meanings
Yoshinaka and Kanazawa (2011)



Weak versus strong learning

Berwick, Chomsky et al. (2011)

This is irrelevant because it is “a mere process of acquiring a (weak generative) capacity to produce just the valid word strings of a language”
What is needed is to learn the “structures a language generates”.

One answer

Learning from pairs of strings and sentence meanings
Yoshinaka and Kanazawa (2011)



Associativity

Definition

$$x + (y + z) = (x + y) + z$$

4 does not have any hierarchical structure

$$1 + 1 + 1 + 1$$

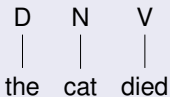
Fundamental issue

String concatenation is associative:

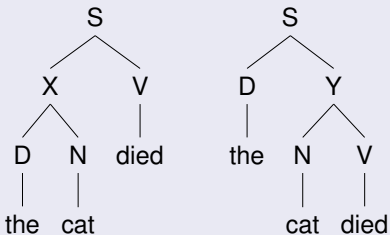
$$((\text{the cat}) \text{ died}) = (\text{the} (\text{cat died}))$$

Trees

MERGE/Tree combination is not associative



Not the same



Fundamental Incompatibility

Associative and non associative structures

Associativity of congruence

$$[u] \circ ([v] \circ [w]) = [uvw] = ([u] \circ [v]) \circ [w]$$

Consequence

We don't get a plausible notion of syntactic structure from these congruential approaches.

The derivations of the learned CFGs are highly ambiguous.

MERGE is completely free

Trivial finite language

L is finite

hello

john likes mary

john likes john

mary likes mary

mary likes john

mary laughed

john laughed

Trivial finite language

L is finite

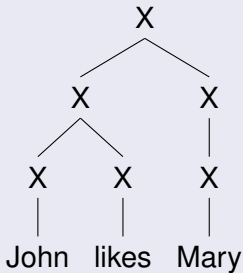
hello
john likes mary
john likes john
mary likes mary
mary likes john
mary laughed
john laughed

Congruence classes

L
{ john, mary }
{ john likes , mary likes }
{ likes mary, likes john, laughed }
{ likes }
{ laughed }

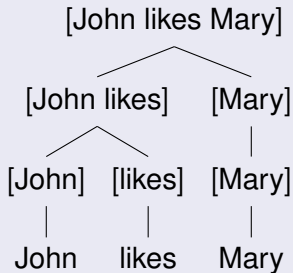
Congruentially labeled trees

String “John likes Mary”

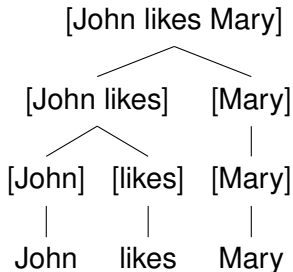


Congruentially labeled trees

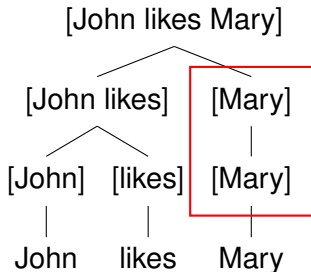
Label each non leaf node with a congruence class



What's wrong with this tree?



What's wrong with this tree?



Redundancy

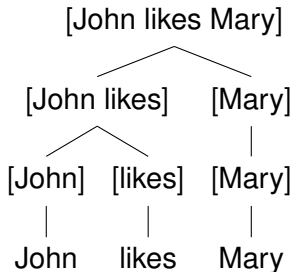
[Mary]

|

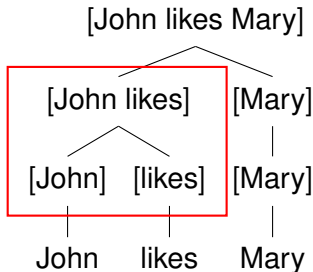
[Mary]

- The set at the root is the same as the set at the child.

What's wrong with this tree?



What's wrong with this tree?



Binary rule

Congruence classes

$$[\text{John}] = \{ \text{john}, \text{mary} \}$$
$$[\text{likes}] = \{ \text{likes} \}$$
$$[\text{John likes}] = \{ \text{john likes}, \text{mary likes} \}$$
$$[\text{John likes}] \supseteq [\text{John}][\text{likes}]$$

In fact:

$$[\text{John likes}] = [\text{John}][\text{likes}]$$

Binary rule

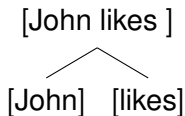
Congruence classes

$$[\text{John}] = \{ \text{john}, \text{mary} \}$$
$$[\text{likes}] = \{ \text{likes} \}$$
$$[\text{John likes}] = \{ \text{john likes}, \text{mary likes} \}$$
$$[\text{John likes}] \supseteq [\text{John}][\text{likes}]$$

In fact:

$$[\text{John likes}] = [\text{John}][\text{likes}]$$

Redundancy

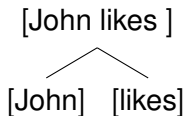


- The set at the root is the same as the concatenation of the sets at the child.

Vacuous local trees

A tree is vacuous if the set labeling the head is equal to the concatenation of the sets labeling the children
Define leaf nodes as singleton sets

Redundancy

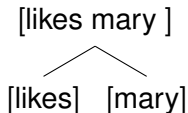


- The set at the root is the same as the concatenation of the sets at the child.

Vacuous local trees

A tree is vacuous if the set labeling the head is equal to the concatenation of the sets labeling the children
Define leaf nodes as singleton sets

Non redundant concatenations

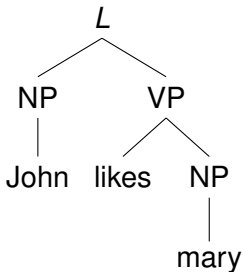


Non vacuous

This is not redundant as

- [likes mary] is larger than [likes] [mary]
- it includes “laughs”

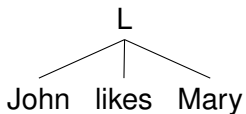
Non vacuous trees



Valid tree

Every local tree is non vacuous!

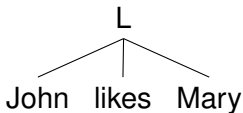
What's wrong with this tree?



Definition: minimal

A local tree is **minimal** if there is no smaller local tree that is non vacuous.

What's wrong with this tree?



Definition: minimal

A local tree is **minimal** if there is no smaller local tree that is non vacuous.

Definition

- A congruentially labeled tree is valid if every local tree is non vacuous and minimal.
- We define the set of SDs for a string to be the set of valid trees.
- If we fix the language, the set of tree structures is fixed, independent of what grammar we use.
- This reduces strong learnability to weak learnability.

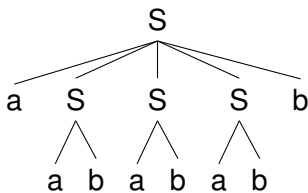
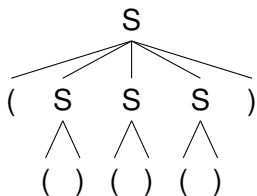
Dyck language

Unambiguously structured

$((()()))$ aabababb

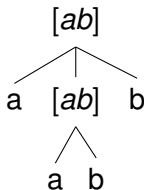
Dyck language

Unambiguously structured



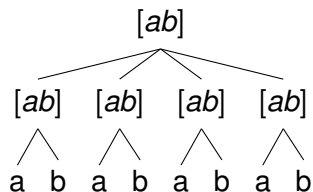
Example from di Sciullo et al. (2010)

$a^n b^n$



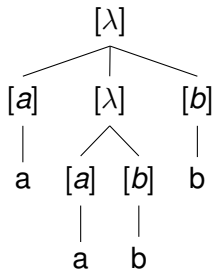
Example from di Sciullo et al. (2010)

$(ab)^*$



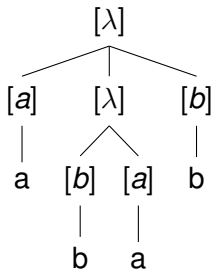
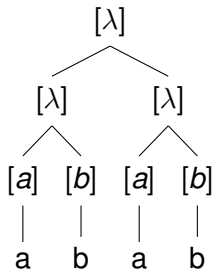
Example

$$\{w \mid |w|_a = |w|_b\}$$



Ambiguous Example

$$\{w \mid |w|_a = |w|_b\}$$



Limitations

- Clearly wrong as the congruential approach is too weak, but this is the right sort of answer.
- We can assign a set of structural trees which make dependencies local.
- Sometimes we will have more than one structural tree for a particular string: “constrained ambiguity”.
- Local trees are not *in general* binary. Restriction to binary merge therefore has nontrivial empirical content.
- Derive constituent structure tests from deeper principles.

Swiss german

... das mer d'chind em Hans es huus lönd hälfe aastrische
 ... that we the children-ACC Hans-DAT house-ACC let help paint

‘... that we let the children help Hans paint the house’

Derivation

Swiss German

comp

N_{acc}

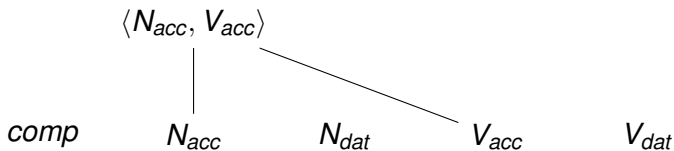
N_{dat}

V_{acc}

V_{dat}

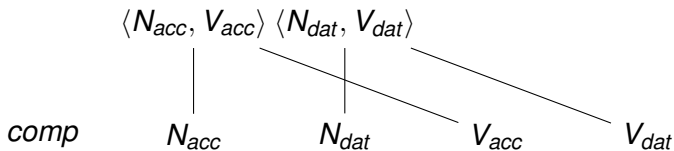
Derivation

Swiss German



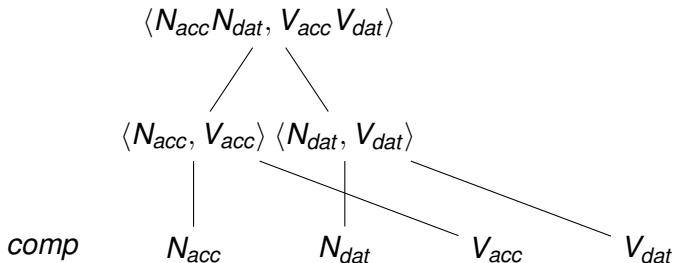
Derivation

Swiss German



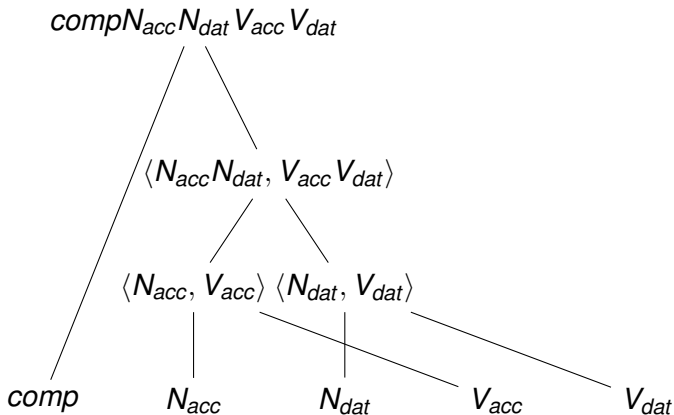
Derivation

Swiss German



Derivation

Swiss German



Extension of congruence

Bicongruence

$[u, v]$ is the set of pairs of strings that can be substituted for u, v

Simplified Swiss German example

$$[N_{acc}, V_{acc}] = \{(N_{acc}, V_{acc}), (N_{dat}, V_{dat}) \dots\}$$

Proper inclusion

$$[u, v]_L \supsetneq [u]_L \times [v]_L$$

Extensions

Details are in “A language theoretic approach to syntactic structure” Mathematics of Language, 2011. (MOL 12).

Two Extensions

- To cross-serial dependencies and displaced constituents using MCFGs.
 - $[u, v]_L \not\supseteq [u]_L \times [v]_L$
 - Generalised composition operations:
 $f(\mathbf{t}_1, \dots, \mathbf{t}_k) \not\supseteq f([\mathbf{t}_1], \dots, [\mathbf{t}_k])$
- A better treatment of ambiguity using distributional lattice based approaches.
 $(PQ)'' \not\supseteq P''Q''$ where $''$ is a closure operator in the Galois lattice.

Conclusion

- There is an interesting confluence between minimalist ideas of merge and the learnable operations of concatenation.
- Distributional version of MERGE is associative, but nontrivial merges have the right sort of hierarchical structure
- MERGE is when composition is strictly more than concatenation:
“the whole must be greater than the sum of the parts”