Learning Formal Grammars:
some selected methods

Wojciech Buszkowski

Faculty of Mathematics and Computer Science
The Adam Mickiewicz University in Poznań

The aim of this lecture is to give some insight into learning methods for formal grammars. We fix some basic notions.

$\mathbb{N}$ denotes the set of natural numbers $1, 2, 3, \ldots$.

$E = \{e_n\}_{n\in\mathbb{N}}$ is a denumerable set of expressions. Subsets of E are called languages (on E).

$\mathcal{P}(E) = \{L : L \subseteq E\}$ (the powerset of E) is the family of all languages on E.

Ω is a class of grammars (finite descriptions of languages). It is called the hypothesis space.

$L : \Omega \mapsto \mathcal{P}(E)$ is a map which to any grammar $G \in \Omega$ assigns a language $L(G) \subseteq E$, called the language of G (or: generated by G).

The triple (E, Ω, L) is called a grammar system.

We consider different kinds of expressions.

Strings on a finite alphabet Σ. Then $E = \Sigma^*$, i.e. the set of all finite strings on Σ, or $E = \Sigma^+$, i.e. the set of all nonempty finite strings on Σ.

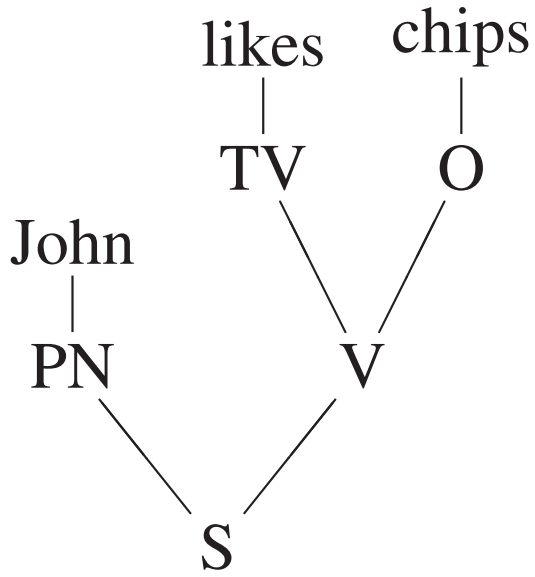
In syntactic considerations Σ consists of all words of the given language; it is called the lexicon or the vocabulary.

Phrase structures on Σ can be identified with bracketed strings. For instance John likes chips is a string, and (John (likes chips)) is a phrase structure.

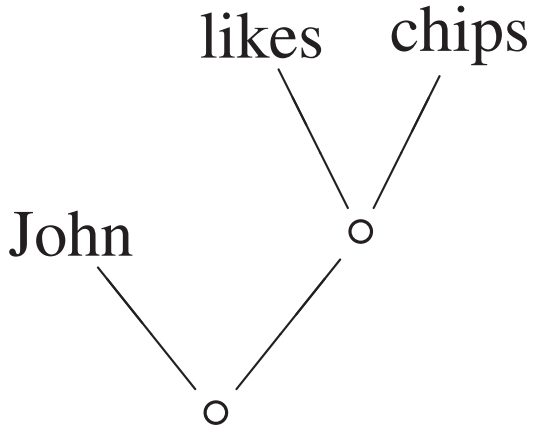
Labeled phrase structures are phrase structures augmented with symbols of categories, e.g. S - sentence, PN - proper noun, V - verb phrase, TV - transitive verb phrase, O - object.
$(\text{John}_{PN}(\text{likes}_{TV}\text{chips}_{O})_{V})_{S}$ is a labeled phrase structure.

Clearly (labeled) phrase structures can be depicted as trees.

$(\text{John}_{PN}(\text{likes}_{TV}\text{chips}_{O})_{V})_{S}$



(John (likes chips))



Production rules:

S →PN,V; V → TV,O; PN → John; TV → likes; O → chips.

One considers classes $\mathcal{G} \subseteq \Omega$, i.e. classes of grammars from Ω.

We denote $L(\mathcal{G}) = \{L(G) : G \in \mathcal{G}\}$; it is the family of all languages generated by grammars from $\mathcal{G}$.

We always assume the following conditions:

(A1) $\mathcal{G}$ is recursively enumerable.

(A2) There is an algorithm which, for any $e \in E$ and $G \in \mathcal{G}$, decides whether $e \in L(G)$ (the global membership problem for $\mathcal{G}$ is decidable).

(A1) means that one can effectively enumerate all grammars from $\mathcal{G}$; so $\mathcal{G} = \{G_n\}_{n \in \mathbb{N}}$, and the sequence $G_1, G_2, \ldots$ is recursive.

One may denote $L_n = L(G_n)$. Then (A2) is equivalent to the existence of a total recursive function f such that $f(e, n) = 1$, if $e \in L_n$, and $f(e, n) = 0$, if $e \notin L_n$.

A learning function is a (partial) function $\varphi : E^+ \mapsto \Omega$. Often a partial function can be replaced by a total function $\varphi' : E^+ \mapsto \Omega'$, where $\Omega' = \Omega \cup \{\perp\}$; one sets $\varphi'(\sigma) = \perp$ iff $\varphi(\sigma)$ is undefined.

A text for $L \subseteq E$ is an infinite sequence $\varepsilon = (\varepsilon_n)_{n\in\mathbb{N}}$ such that $L = \{\varepsilon_n\}_{n\in\mathbb{N}}$. Clearly $\varepsilon_n \in E$. We denote $\varepsilon \restriction n = (\varepsilon_1, \ldots, \varepsilon_n)$.

One says that a learning function φ converges on a text ε to $G \in \Omega$, if $\varphi(\varepsilon \restriction n) = G$, for all but finitely many n.

Let $\mathcal{G} \subseteq \Omega$. One says that a learning function φ learns a language $L \in L(\mathcal{G})$ (with respect to $\mathcal{G}$), if, for every text ε for L, there exists $G \in \mathcal{G}$ such that $L = L(G)$ and φ converges to G on ε.

One says that φ learns a class $\mathcal{G}$, if φ learns every language from $L(\mathcal{G})$. $\mathcal{G}$ is said to be (resp. effectively) learnable, if there exists a (resp. computable) learning function φ which learns $\mathcal{G}$.

The above definitions correspond to *learning from positive data*. Now we admit negative data.

A full text for $L \subseteq E$ is an infinite sequence ε whose $n-$th element is a pair (ε_n, l_n) such that $E = \{\varepsilon_n : n \in \mathbb{N}\}$ and $l_n = 1$, if $\varepsilon_n \in L$, $l_n = 0$, if $\varepsilon_n \notin L$.

One can adapt the notions of a learning function, convergence, learning a language and learning a class of grammars in an obvious way, which yields the notion of *learnability from positive and negative data*.

Gold (1967) shows: *every class $\mathcal{G}$ (satisfying (A1), (A2)) is effectively learnable from positive and negative data.*

PROOF. We have $\mathcal{G} = \{G_n\}_{n\in\mathbb{N}}$. For any nonempty finite sequence σ on $E \times \{0, 1\}$, define $\varphi(\sigma)$ as the first G_i which is compatible with σ (undefined, if no G_i compatible with σ exists). Clearly, if ε is a full text for $L \in L(\mathcal{G})$, then φ converges on ε to the first G_i such that $L = L(G_i)$.

Learning from positive data is more restricted.

One says that a class $\mathcal{L}$ of languages has a limit point, if there is an infinite, strictly ascending chain (L_n) of languages from $\mathcal{L}$ whose union also belongs to $\mathcal{L}$.

It is known that *no class $\mathcal{G}$ such that $L(\mathcal{G})$ has a limit point is learnable from positive data* (Kapur 1991, Kanazawa 1998).

As a consequence, *if $L(\mathcal{G})$ contains all finite languages and at least one infinite language, then $\mathcal{G}$ is not learnable* (Gold 1967).

This shows that standard classes of the Chomsky hierarchy are not learnable: regular grammars (finite-state automata), context-free grammars, context-sensitive grammars.

Therefore one considers subclasses of those classes, e.g. reversible finite-state automata, context-free grammars with at most k production rules and others, which are effectively learnable.

Let $\mathcal{L}$ be a class of languages and $L \in \mathcal{L}$. A set $D \subseteq L$ is called a finite tell-tale for L in $\mathcal{L}$, if D is finite, and there exists no language $L' \in \mathcal{L}$ such that $D \subseteq L'$ and $L' \subset L$.

Theorem 1. (Jain et al. 1999, based on Angluin 1980). *A class $\mathcal{G}$ is learnable iff, for every $L \in L(\mathcal{G})$, there exists a finite tell-tale for L in $L(\mathcal{G})$.*

Angluin (1980) proved a similar characterization of effective learnability (see former lectures).

Using Theorem 1, one easily shows that, if $L(\mathcal{G})$ has a limit point, then $\mathcal{G}$ is not learnable (a direct proof is also easy).

Several positive and negative results on learnability were obtained with applying finite tell-tales. On the other hand, it is often difficult to prove that the given class of grammars fulfills the condition of Theorem 1 (and more difficult for its effective version).

Finite Elasticity

We discuss another criterion of effective learnability, introduced in (Wright 1989, Motoki et al. 1991).

A class of languages $\mathcal{L}$ is said to have infinite elasticity, if there exist infinite sequences $(L_n)_{n\in\mathbb{N}}$ of languages from $\mathcal{L}$ and $(w_n)_{n\in\mathbb{N}}$ of expressions from E such that, for any $n \in \mathbb{N}$, $w_n \notin L_n$ and $\{w_1, \ldots, w_n\} \subseteq L_{n+1}$.

One says that $\mathcal{L}$ has finite elasticity, if $\mathcal{L}$ does not have infinite elasticity.

Obvious properties:

(1) *Every finite class has finite elasticity.*

(2) *If $\mathcal{L}$ has finite elasticity and $\mathcal{L}' \subseteq \mathcal{L}$, then $\mathcal{L}'$ has finite elasticity.*

(3) *The class of all finite languages has infinite elasticity.*

In the literature one can find different variants and consequences of finite elasticity. Buszkowski (2003) formulates some equivalent conditions.

One says that a class of languages $\mathcal{L}$ satisfies ascending chain condition (ACC), if there exists no infinite, strictly ascending chain of languages from $\mathcal{L}$.

INT($\mathcal{L}$) denotes the smallest class containing $\mathcal{L}$ and being closed under arbitrary intersections.

Theorem 2. *The following conditions are equivalent.*

(FE1) $\mathcal{L}$ *has finite elasticity.*

(FE2) *For any set* $L \subseteq E$*, there exists a finite set* $D \subseteq L$ *such that every language* $L' \in \mathcal{L}$ *containing* D *also contains* L*.*

(FE3) *INT*($\mathcal{L}$) *satisfies (ACC).*

(FE2) directly implies:

(FE2') *For any $L \in \mathcal{L}$, there exists a finite set $D \subseteq L$ such that L is the smallest language in $\mathcal{L}$ which contains D.*

(FE1) $\Rightarrow$ (FE2') was shown by Kapur (1991).

The set D in (FE2') is a finite tell-tale for L in $\mathcal{L}$.

Hence Theorem 1 yields: *if $L(\mathcal{G})$ has finite elasticity, then $\mathcal{G}$ is learnable*.

In fact, a stronger theorem holds (essentially Wright 1989).

Theorem 3. *If $L(\mathcal{G})$ has finite elasticity, then $\mathcal{G}$ is effectively learnable.*

PROOF. Assume that $L(\mathcal{G})$ has finite elasticity.

Fix a standard enumeration $(e_n)_{n\in\mathbb{N}}$ of all expressions from E.

For $L \in L(\mathcal{G})$, define $L^{(n)} = L \cap \{e_1, \ldots, e_n\}$.

The function φ is defined as follows.

Let $\sigma \in E^+$ be of length n. From the set $\{G_1, \ldots, G_n\}$ we choose all grammars G_i such that $\mathrm{set}(\sigma) \subseteq L(G_i)$. The chosen grammars form a set X_n.

If $X_n \neq \emptyset$, we define $\varphi(\sigma)$ as the first $G_j \in X_n$ such that $(L(G_j))^{(n)}$ properly contains no $(L(G_i))^{(n)}$ such that $G_i \in X_n$. If $X_n = \emptyset$, we define $\varphi(\sigma) = \perp$.

Let $L \in L(\mathcal{G})$. Let $G_k \in \mathcal{G}$ be the first grammar such that $L = L(G_k)$.

Let D be as in (FE2'), it means: D is finite, $D \subseteq L$ and L is the smallest language in $L(\mathcal{G})$ such that $D \subseteq L$.

Let ε be a text for L.

If n is sufficiently large, then $D \subseteq \text{set}(\varepsilon \upharpoonright n)$ and $k \leq n$. Accordingly, G_k is in the set $\{G_1, \ldots, G_n\}$ and $\text{set}(\varepsilon \upharpoonright n) \subseteq L(G_k)$. So $G_k \in X_n$.

Clearly $(L(G_k))^{(n)}$ properly contains no $(L(G_i))^{(n)}$ such that $G_i \in X_n$. It, however, may happen that the same holds for some $k' < k$, whence $\varphi(\varepsilon \upharpoonright n) \neq G_k$.

But we have $L(G_k) \subset L(G_i)$, for all $i < k$ such that $G_i \in X_n$. For any such i, there exists a witness $w_i \in L(G_i) - L(G_k)$.

If n is sufficiently large, then $k \leq n$, $D \subseteq \text{set}(\varepsilon \upharpoonright n)$, and, for any $i < k$ such that $G_i \in X_n$, at least one witness w_i belongs to $\{e_1, \ldots, e_n\}$.

Consequently, $(L(G_k))^{(n)}$ is a proper subset of $(L(G_i))^{(n)}$, for any $i < k$ such that $G_i \in X_n$, and a subset of any other $(L(G_i))^{(n)}$ such that $G_i \in X_n$. We obtain $\varphi(\varepsilon \upharpoonright n) = G_k$, for all sufficiently large n.

Finite elasticity is preserved under different operations on classes of languages as e.g.:

$\mathcal{L} \oplus \mathcal{M} = \{L \cup M : L \in \mathcal{L}, M \in \mathcal{M}\}$

$\mathcal{L} \otimes \mathcal{M} = \{LM : L \in \mathcal{L}, M \in \mathcal{M}\}$ (here $E_1 = E_2 = \Sigma^*$)

These results have been generalized by Kanazawa (1996, 1998).

Let R be a binary relation, $R \subseteq E_1 \times E_2$.

For $M \subseteq E_2$, define $R^{-1}(M) = \{s \in E_1 : \exists u(sRu \wedge u \in M)\}$.

For $\mathcal{M} \subseteq \mathcal{P}(E_2)$, define $R^{-1}(\mathcal{M}) = \{R^{-1}(M) : M \in \mathcal{M}\}$.

A relation $R \subseteq E_1 \times E_2$ is said to be finitely valued, if, for any $s \in E_1$, there exist only finitely many $u \in E_2$ such that sRu.

Theorem 4. *Let $R \subseteq E_1 \times E_2$ be finitely valued. If $\mathcal{M} \subseteq \mathcal{P}(E_2)$ has finite elasticity, then $R^{-1}(\mathcal{M})$ has finite elasticity.*

The proof uses König's lemma: *every infinite, finitely branching tree has an infinite branch.*

We show that $\mathcal{L} \otimes \mathcal{M}$ has finite elasticity, if both $\mathcal{L}$ and $\mathcal{M}$ have finite elasticity. First we define:

$L \times M = \{\langle s, t\rangle : s \in L, t \in M\}$, $\mathcal{L}\overline{\otimes}\mathcal{M} = \{L \times M : L \in \mathcal{L}, M \in \mathcal{M}\}$

We show that $\mathcal{L}\overline{\otimes}\mathcal{M}$ has finite elasticity, if this holds for $\mathcal{L}$ and $\mathcal{M}$.

Suppose that $\mathcal{L}\overline{\otimes}\mathcal{M}$ has infinite elasticity. Let $(\langle s_n, t_n\rangle)_{n\in\mathbb{N}}$ and $(L_n \times M_n)_{n\in\mathbb{N}}$ be such that $\langle s_n, t_n\rangle \notin L_n \times M_n$ and $\{\langle s_1, t_1\rangle, \ldots, \langle s_n, t_n\rangle\} \subseteq L_{n+1} \times M_{n+1}$, for all $n \in \mathbb{N}$. There exist infinitely many integers $n_1 < n_2 < \ldots < n_k < \ldots$ such that either $s_{n_i} \notin L_{n_i}$, for all $i \in \mathbb{N}$, or $t_{n_i} \notin M_{n_i}$, for all $i \in \mathbb{N}$.

For the first case, $(s_{n_i})_{i\in\mathbb{N}}$ and $(L_{n_i})_{i\in\mathbb{N}}$ witness the infinite elasticity of $\mathcal{L}$, and similarly for the second case.

Define $vR\langle s, t\rangle$ iff $v = st$. Clearly $R \subseteq \Sigma^* \times (\Sigma^* \times \Sigma^*)$ is finitely valued.

$\mathcal{L} \otimes \mathcal{M} = R^{-1}(\mathcal{L}\overline{\otimes}\mathcal{M})$, which yields our claim, by Theorem 4.

A generative grammar is a quadruple $G = (V, \Sigma, S, P)$ such that:

V is a finite set of variables (or: nonterminal symbols),

Σ is a finite set of terminal symbols,

$S \in V$ is the start symbol,

P is a finite set of production rules of the form $u \to w$ such that $u, w \in (V \cup \Sigma)^*$, $u \neq \lambda$, $u \neq w$ (λ denotes the empty string).

$x \Rightarrow_G y$ iff there exist $z_1, z_2, u, w \in (V \cup \Sigma)^*$ such that $(u \to w) \in P$, $x = z_1 u z_2$, $y = z_1 w z_2$.

$x \Rightarrow_G^* y$ iff there exists a sequence $(x_0, \ldots, x_n)$ such that $n \geq 0$, $x_0 = x$, $x_n = y$ and $x_{i-1} \Rightarrow_G x_i$, for all $1 \leq i \leq n$.

The set $L(G) = \{w \in \Sigma^* : \mathrm{S} \Rightarrow_G^* w\}$ is called the language of G.

G_1 is said to be equivalent to G_2, if $L(G_1) = L(G_2)$.

A context-free grammar: all production rules are of the form $A \to w$ such that $A \in V$, $w \in (V \cup \Sigma)^*$.

Every context-free grammar can be transformed into an equivalent context-free grammar whose each rule $A \to w$ satisfies $w \neq \lambda$ with the possible exception of $S \to \lambda$ provided that S does not appear on the right-hand side of any rule.

By CFG(Σ, k) we denote the class of all context-free grammars with the terminal alphabet Σ, satisfying the above condition and having at most k production rules.

We show that CFG(Σ, k) *is effectively learnable from strings* (it means: $E = \Sigma^*$); a stronger result is due to Shinohara (1990).

Derivations in context-free grammars can be represented as derivation trees (see slide 4).

Every grammar from $\mathrm{CFG}(\Sigma, k)$ admits at most k variables, appearing in derivation trees of terminal strings. We may assume that these variables belong to the set $\{A_1, \ldots, A_k\}$ and $S = A_1$.

For $G \in \mathrm{CFG}(\Sigma, k)$, by $M(G)$ we denote the set of all derivation trees of strings $w \in L(G)$. Every $w \in L(G)$ is the yield of some $t \in M(G)$ such that no variable is repeated on any unary path (a chain of unary rules) in t:

By $T(\Sigma)$ we denote the set of all possible finite trees such that all leaves are labeled by symbols from Σ, internal nodes by $A_1, \ldots, A_k$, the root by A_1, and no A_i can be repeated on a unary path; we also admit the tree with root A_1 and the only leaf λ.

Clearly, for any $w \in \Sigma^*$, there exist only finitely many $t \in T(\Sigma)$ such that w is the yield of t.

We define a relation: wRt iff $t \in T(\Sigma)$ and w is the yield of t.

$R \subseteq \Sigma^* \times T(\Sigma)$ is finitely valued.

Let $\mathcal{M}$ denote the class of all $M(G)$, for $G \in \mathrm{CFG}(\Sigma, k)$. It is easy to show that $\mathcal{M}$ has finite elasticity. If $t_n \notin M_n$, $\{t_1, \ldots, t_n\} \subseteq M_{n+1}$ hold for $n = 1, 2, 3, \ldots$, then each t_n must introduce a new production rule. Consequently $n \leq k$.

Since $L(\mathrm{CFG}(\Sigma, k)) = R^{-1}(\mathcal{M})$, then $L(\mathrm{CFG}(\Sigma, k))$ has finite elasticity, by Theorem 4. Hence $\mathrm{CFG}(\Sigma, k)$ *is effectively learnable*, by Theorem 3.

Learning by Unification

Buszkowski (1987), Buszkowski and Penn (1990) applied unification for computing type grammars directly from structural data (a similar approach is due to van Benthem 1987). This method can be generalized for other formal grammars, more or less directly.

Atomic types: certain constants, e.g. s - statement, n - proper noun, n^* - common noun.

In serious applications to natural language, one needs much more atomic types, e.g. s_1 - statement in Present, s_2 - statement in Past, π_1 - subject in First Person, π_2 - subject in Second Person, and so on.

Types: atomic types, $A\backslash B$, A/B. $\mathcal{T}$ denotes the set of all types.

Reduction laws: $A,\ A\backslash B \Rightarrow B$; $B/A,\ A \Rightarrow B$

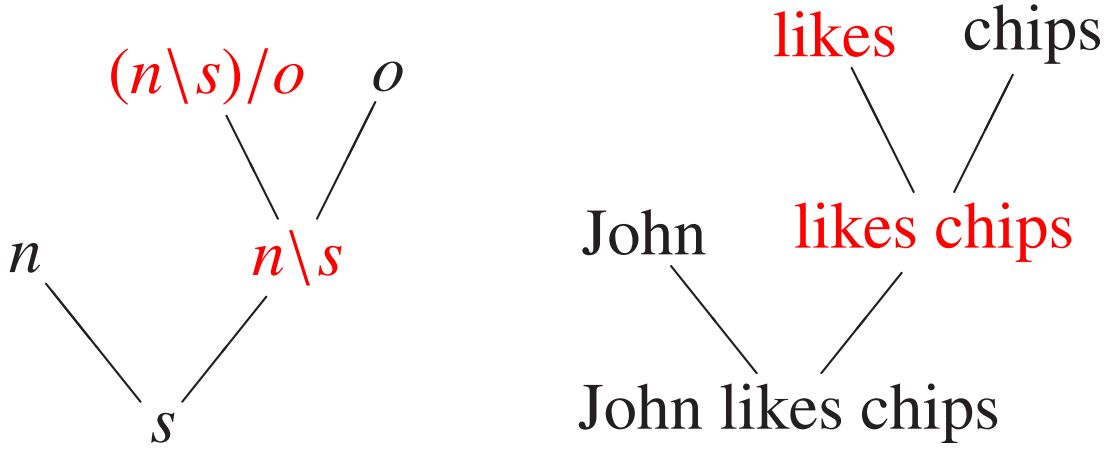
An AB-grammar on Σ: a finite relation $G \subseteq \Sigma \times \mathcal{T}$.

Ajdukiewicz (1935), Bar-Hillel, Gaifman and Shamir (1960)

G : John $\rightarrow n$, likes $\rightarrow (n\backslash s)/o$, chips $\rightarrow o$

John likes chips leads to the sequence $n, (n\backslash s)/o, o$ which reduces to s by reduction laws.

$n, \underline{(n\backslash s)/o,\ o} \Rightarrow n,\ n\backslash s \Rightarrow s$



This reduction determines a functor-argument structure (f-structure), depicted as the right tree: the red constituents are functors, the black constituents are arguments.

A shorter representation: (John (likes chips)$_1$)$_2$

The above grammar is 1-valued: each word is assigned at most one type.

In general, one assigns finitely many types to one word. For instance, drinks can be assigned $n\backslash s$ in John drinks and $(n\backslash s)/o$ in John drinks wine.

A type grammar is said to be $k-$valued, if each word is assigned at most k types.

We fix an atomic type s, called the principal type.

The language of G, denoted by $L(G)$, consists of all strings whose at least one associated sequence of types reduces to s.

The f-language of G, denoted by $L_f(G)$, consists of all f-structures, determined on strings from $L(G)$ in the way described above.

$F(\Sigma)$ denotes the set of all f-structures on Σ.

Subsets of $F(\Sigma)$ are called f-languages on Σ.

We discuss learning AB-grammars from f-structures.

The type s is treated as a constant, and all other atomic types are treated as variables.

A substitution is an assignment α of types to variables, e.g. $\alpha = \{x := y,\ y := s/z\}$. Then $x\alpha = y$, $y\alpha = s/z$, $v\alpha = v$, for all other variables v.

A substitution α is extended to a map $\alpha : \mathcal{T} \mapsto \mathcal{T}$, by setting: $(A\backslash B)\alpha = A\alpha\backslash B\alpha$, $(A/B)\alpha = A\alpha/B\alpha$.

$G\alpha$ is the smallest grammar such that $G\alpha : a \to A\alpha$ if $G : a \to A$, for all $a \in \Sigma$, $A \in \mathcal{T}$.

EXAMPLE.

$G : a \to x,\ b \to s/y$

$\alpha = \{x := z,\ y := z\backslash s\}$

$G\alpha : a \to z,\ b \to s/(z\backslash s)$

A substitution α is called a unifier of G, if $G\alpha$ is 1-valued. G is said to be unifiable, if there exists a unifier of G.

A unifier of G is called a most general unifier (mgu) of G, if, for any unifier β of G, there exists a substitution γ such that $\beta = \alpha\gamma$.

Proposition 1. $L_f(G) \subseteq L_f(G\alpha)$, $L(G) \subseteq L(G\alpha)$.

Proposition 2. *The standard unification algorithm checks whether G is unifiable and, if so, returns an mgu of G.*

Given a nonempty finite set $S \subseteq F(\Sigma)$, one determines a unique grammar GF(S), called the grammar form for S.

(1) Assign s to structures from S and different variables to all occurrences of argument substructures of these structures.

(2) Assign functor types to functor substructures by the rules:

(R/) if $(XY)_1 \to A$ and $Y \to B$, then $X \to A/B$,

(R\) if $(XY)_2 \to A$ and $X \to B$, then $Y \to B\backslash A$.

(3) GF(S) collects all assignments $a \to A$, for $a \in \Sigma$, $A \in \mathcal{T}$, obtained by (1), (2).

EXAMPLE. Let S consist of the structures:

$(\text{John sings})_2$, $((\text{some student})_1$ sings$)_1$

The above structures are assigned s. By (1), we assign:

John $\to x$, sings $\to y$, student $\to z$.

By (2), we assign:

sings $\to x\backslash s$, $(\text{some student})_1 \to s/y$, some $\to (s/y)/z$.

GF(S): John $\to x$, sings $\to y$, $x\backslash s$, student $\to z$, some $\to (s/y)/z$

GF(S) is unifiable with an mgu $\alpha = \{y := x\backslash s\}$

GF(S)α: John $\to x$, sings $\to x\backslash s$, student $\to z$, some $\to (s/(x\backslash s))/z$

One easily shows $L_f(\mathrm{GF}(S)) = S$.

Proposition 3. *$S \subseteq L_f(G)$ iff there exists a substitution α such that $GF(S)\alpha \subseteq G$.*

$\mathrm{AB}(\Sigma, k)$ denotes the class of $k-$valued AB-grammars on Σ.

The class $L_f(\mathrm{AB}(\Sigma, k))$ is denoted by $\mathcal{L}_f^{\Sigma,k}$.

1-valued grammars

Proposition 4. *$S \subseteq L$, for some $L \in \mathcal{L}_f^{\Sigma,1}$, iff $GF(S)$ is unifiable.*

Proposition 5. *If α is an mgu of $GF(S)$, then $L(GF(S)\alpha)$ is the smallest f-language $L \in \mathcal{L}_f^{\Sigma,1}$ such that $S \subseteq L$.*

The grammar $\mathrm{GF}(S)\alpha$ such that α is an mgu of S is denoted by $1VG(S)$ and called the 1-valued grammar determined by S.

Proposition 6. (Kanazawa 1996) *If $L_1 \subset L_2 \subset \cdots \subset L_n$ is a strictly ascending chain of f-languages from $\mathcal{L}_f^{\Sigma,1}$, then $n \leq |\Sigma|$.*

Consequently, $\mathcal{L}_f^{\Sigma,1}$ satisfies (ACC).

Using Propositions 5 and 6 one shows that $\mathcal{L}_f^{\Sigma,1}$ is closed under intersections of nonempty subfamilies.

Consequently $INT(\mathcal{L}_f^{\Sigma,1}) = \mathcal{L}_f^{\Sigma,1} \cup \{F(\Sigma)\}$, and this family satisfies (ACC).

By Theorem 2, $\mathcal{L}_f^{\Sigma,1}$ has finite elasticity. Hence AB$(\Sigma, 1)$ *is effectively learnable from f-structures*, by Theorem 3.

A natural learning function φ is defined as follows.

For $\sigma \in (F(\Sigma))^+$, we set $\varphi(\sigma) = 1VG(\text{set}(\sigma))$, if set$(\sigma)$ is unifiable; otherwise $\varphi(\sigma) = \perp$.

$k-$valued grammars

Every $k-$valued AB-grammar G can be transformed into a 1-valued AB-grammar $G^{(1)}$.

Let $G \in \mathrm{AB}(\Sigma, k)$. The lexicon Σ is replaced by a new lexicon $\Sigma^{(k)}$ in which each $a \in \Sigma$ is replaced by its k copies $a^{(1)}, \ldots, a^{(k)}$. If $A_1, \ldots, A_n$, $n \leq k$, is a fixed enumeration of all types assigned by G to a, then $G^{(1)}$ assigns A_i to $a^{(i)}$.

A relation $R \subseteq F(\Sigma) \times F(\Sigma^{(k)})$ is defined as follows: XRY iff X arises from Y by replacing each $a^{(i)}$ by a.

R is finitely valued, and $\mathcal{L}_f^{\Sigma,k} = R^{-1}(\mathcal{L}_f^{\Sigma^{(k)},1})$.

By Theorem 4, $\mathcal{L}_f^{\Sigma,k}$ has finite elasticity. By Theorem 3, $\mathrm{AB}(\Sigma, k)$ *is effectively learnable from f-structures* (Kanazawa 1996).

To define a natural learning function we need some new notions.

Let G be an AB-grammar. A substitution α is called a $k-$mgu of G, if $G\alpha$ is $k-$valued and, for any substitution β, satisfying the condition:

(*) for any $(a \to A), (a \to B) \in G$, if $\alpha(A) = \alpha(B)$ then $\beta(A) = \beta(B)$,

there exists a substitution γ such that $\beta = \alpha\gamma$.

One can compute all (up to renaming of variables) $k-$mgu's of G.

Let $\sigma \in (F(\Sigma))^+$. Denote $S = \text{set}(\sigma)$. We compute all $k-$mgu's of $\text{GF}(S)$ and all grammars $\text{GF}(S)\alpha$ such that α is a $k-$mgu of $\text{GF}(S)$. As the value $\varphi(\sigma)$ we take one of these grammars whose f-language is $\subseteq-$minimal (among f-languages of these grammars).

Using the finite elasticity of $\mathcal{L}^{\Sigma,k}$ and Proposition 3, one proves that φ *learns* $\text{AB}(\Sigma, k)$ *from f-structures*.

Learning from strings

Since each string $w \in \Sigma^+$ is the yield of only finitely many structures $X \in F(\Sigma)$, then the class of string languages of $k-$valued AB-grammars has finite elasticity. Consequently $\mathrm{AB}(\Sigma, k)$ *is effectively learnable from strings* (Kanazawa 1996).

A natural learning function can be designed as above, but it is less efficient. For $S = \mathrm{set}(\sigma)$, where σ is a sequence of strings from Σ^+, one constructs all minimal sets $S' \subseteq F(\Sigma)$ such that every string in S is the yield of some structure from S'. One computes the set $\mathcal{G}(S)$ of all grammars $\mathrm{GF}(S')\alpha$ such that α is a $k-$mgu of $\mathrm{GF}(S')$.

Since the inclusion problem $L_1 \subseteq L_2$ is undecidable for string languages of AB-grammars, we cannot effectively find a grammar $G \in \mathcal{G}(S)$ whose string language is $\subseteq -$minimal in $L(\mathcal{G}(S))$. Instead we follow the method from the proof of Theorem 3: we choose a minimal language with respect to the relation $L_1^{(n)} \subseteq L_2^{(n)}$, where n is the length of σ.

Belief revision

The natural learning methods for $\mathrm{AB}(\Sigma, k)$ can be presented as effective procedures following a general paradigm of belief-revision based learning methods (Gierasimczuk 2010).

For learning from structures, the initial epistemic state is $(\mathcal{L}_f^{\Sigma,k}, \subseteq)$, denoted here by $\mathcal{L}$. The belief-revision method can be defined as conditioning:

$R(\mathcal{L}, \subseteq, \sigma) = (\mathcal{L}^\sigma, \subseteq)$, where $\mathcal{L}^\sigma = \{L \in \mathcal{L} : \mathrm{set}(\sigma) \subseteq L\}$

The learning method:

$L_R(\mathcal{L}, \subseteq, \sigma) = \min_\subseteq(\mathcal{L}^\sigma)$

can be computed. By the above methods, one computes all grammars $\mathrm{GF}(S)\alpha$ such that $S = \mathrm{set}(\sigma)$ and α is a $k-$mgu of $\mathrm{GF}(S)$, and from them one can effectively choose those whose f-languages are minimal. By Proposition 3, the latter grammars yield precisely the $\subseteq-$minimal f-languages in $\mathcal{L}^\sigma$.

For learning from strings, on each stage the inclusion order is replaced by $\subseteq^{(n)}$:

$L_1 \subseteq^{(n)} L_2$ iff $L_1^{(n)} \subseteq L_2^{(n)}$, where $L^{(n)} = L \cap E_n$

E_n has been defined as the set $\{e_1, \ldots, e_n\}$, but we can also define E_n as the set of all strings from Σ^+ whose length is not greater than n.

Now $\mathcal{L}$ is the class of all string languages of grammars from $\mathrm{AB}(\Sigma, k)$.

σ is a sequence of strings from Σ^+.

$\mathcal{L}^\sigma$ is defined as above.

$R(\mathcal{L}, \sigma) = (\mathcal{L}^\sigma, \subseteq^{(n(\sigma))})$, where $n(\sigma)$ denotes the length of σ.

$L_R(\mathcal{L}, \sigma) = \min_{\subseteq^{(n(\sigma))}}(\mathcal{L}^\sigma \cap L(\mathcal{G}(\mathrm{set}(\sigma))))$.

In both cases, if ε is a text for $L \in \mathcal{L}$, then, for sufficiently great n, L_R stabilizes on $\varepsilon \upharpoonright n$ and returns $\{L\}$.

Syntactic Concept Lattices

Clark (2010, 2011) proposes a new learning method for context-free grammars; it works for the restricted class of context-free grammars, satisfying Finite Context Property.

One uses a Galois connection between sets $S \subseteq \Sigma^*$ and sets of contexts $C \subseteq \Sigma^* \times \Sigma^*$.

Let $L \subseteq \Sigma^*$. For $S \subseteq \Sigma^*$, $C \subseteq \Sigma^* \times \Sigma^*$, define:

$S^{\triangleright} = \{\langle u, v\rangle : (\forall s \in S) usv \in L\}$, $C^{\triangleleft} = \{s : (\forall \langle u, v\rangle \in Y) usv \in L\}$

The adjoint law: $S \subseteq C^{\triangleleft}$ iff $C \subseteq S^{\triangleright}$.

Consequences:

$S \subseteq S^{\triangleright\triangleleft}$, $C \subseteq C^{\triangleleft\triangleright}$

$^{\triangleleft}$,$^{\triangleright}$ are antitone with respect to $\subseteq$.

$S^{\triangleright} = S^{\triangleright\triangleleft\triangleright}$, $C^{\triangleleft} = C^{\triangleleft\triangleright\triangleleft}$

Consequently $^{\triangleright\triangleleft}$ is a closure operation on $\mathcal{P}(\Sigma^*)$, it means:

(C1) $S \subseteq S^{\triangleright\triangleleft}$,

(C2) if $S_1 \subseteq S_2$ then $S_1^{\triangleright\triangleleft} \subseteq S_2^{\triangleright\triangleleft}$,

(C3) $S^{\triangleright\triangleleft\triangleright\triangleleft} = S^{\triangleright\triangleleft}$,

and similarly, $^{\triangleleft\triangleright}$ is a closure operation on $\mathcal{P}(\Sigma^* \times \Sigma^*)$.

A set S (resp. C) is said to be closed, if $S = S^{\triangleright\triangleleft}$ (resp. $C = C^{\triangleleft\triangleright}$).

The family of all closed sets $S \subseteq \Sigma^*$ forms a complete lattice, called the syntactic concept lattice determined by L and denoted by $\mathcal{B}(L)$. The meet in $\mathcal{B}(L)$ is the intersection of sets.

Elements of $\mathcal{B}(L)$, called concepts, are often represented as pairs (S, C) such that $S^{\triangleright} = C$, $C^{\triangleleft} = S$ (hence S, C are closed).

Proposition 7. *$\mathcal{B}(L)$ is finite iff L is a regular language.*

For non-regular languages, we obtain infinite syntactic concept lattices.

For learning purposes, Clark replaces them by relativised syntactic concept lattices (RSCLs). We partially follow the terminology and notation of Leiss (2012).

Let $K \subseteq \Sigma^*$, $F \subseteq \Sigma^* \times \Sigma^*$ be nonempty and finite. For $S \subseteq K$, $C \subseteq F$, we define:

$S^F = S^{\triangleright} \cap F$, $C^K = C^{\triangleleft} \cap K$

Again the adjoint law holds: $S \subseteq C^K$ iff $C \subseteq S^F$.

Consequently ^{FK} and ^{KF} are closure operations on $\mathcal{P}(K)$ and $\mathcal{P}(F)$, respectively.

The closed subsets of K are precisely the sets C^K, for $C \subseteq F$, and the closed subsets of F are precisely the sets S^F, for $S \subseteq K$.

We obtain an RSCL, denoted by $\mathcal{B}(K, L, F)$, which is finite.

The concepts in $\mathcal{B}(K, L, F)$ are finite sets, which can be computed provided that one can check whether $usv \in L$, for $s \in K$, $\langle u, v)\rangle \in F$.

We define a context-free grammar $G(K, L, F)$ in (generalized) Chomsky Normal Form: all production rules are of the form $C \to D, E$, $C \to a$, $C \to \lambda$, where C, D, E are variables, $a \in \Sigma$.

The variables are all nonempty sets $C \subseteq F$. They correspond to concepts $C^K \subseteq K$. In particular, $\{(\lambda, \lambda)\}^K = L \cap K$. The start symbol is $\{(\lambda, \lambda)\}^{KF}$. We assume $\Sigma \subseteq K$, $\lambda \in K$, $(\lambda, \lambda) \in F$.

We admit all rules:

$C \to DE$ such that $C \subseteq (D^K E^K)^F$,

$C \to a$ such that $a \in C^K$, $a \in \Sigma$,

$C \to \lambda$ such that $\lambda \in C^K$.

This definition of $G(K, L, F)$ is due to Leiss (2012).

Proposition 8. *If* $F \subseteq F'$, *then* $L(G(K, L, F)) \subseteq L(G(K, L, F'))$.

Proposition 9. *If* $K \subseteq K'$, *then* $L(G(K', L, F)) \subseteq L(G(K, L, F))$.

Let $G = (V, \Sigma, S, P)$ be a context-free grammar. For any $A \in V$, we define:

$L(G, A) = \{w \in \Sigma^* : A \Rightarrow_G^* w\}$

Clearly $L(G) = L(G, S)$.

A grammar G is said to have finite context property (FCP), if, for any $A \in V$, there exists a finite nonempty set $C \subseteq \Sigma^* \times \Sigma^*$ such that $A = C^{\triangleleft}$ in $\mathcal{B}(L(G))$.

A set $F \subseteq \Sigma^* \times \Sigma^*$ is said to be adequate for a language $L \subseteq \Sigma^*$, if for every finite set $K \subseteq \Sigma^*$ such that $\Sigma \subseteq K$, $\lambda \in K$, we have $L \subseteq L(G(K, L, F))$.

Theorem 5. *If* G *is a context-free grammar in Chomsky Normal Form,* $L(G) \neq \emptyset$ *and* G *has FCP, then there exist finite sets* K, F *such that* $L(G) = L(G(K, L, F))$ *and* F *is adequate for* $L(G)$.