A Learnable Constraint-based Grammar Formalism

Smaranda Muresan
School of Communication and Information
Rutgers University
smuresan@rci.rutgers.edu

Abstract

Lexicalized Well-Founded Grammar (LWFG) is a recently developed syntactic-semantic grammar formalism for deep language understanding, which balances expressiveness with provable learnability results. The learnability result for LWFGs assumes that the semantic composition constraints are learnable. In this paper, we show what are the properties and principles the semantic representation and grammar formalism require, in order to be able to learn these constraints from examples, and give a learning algorithm. We also introduce a LWFG parser as a deductive system, used as an inference engine during LWFG induction. An example for learning a grammar for noun compounds is given.

1 Introduction

Recently, several machine learning approaches have been proposed for mapping sentences to their formal meaning representations (Ge and Mooney, 2005; Zettlemoyer and Collins, 2005; Muresan, 2008; Wong and Mooney, 2007; Zettlemoyer and Collins, 2009). However, only few of them integrate the semantic representation with a grammar formalism: λ-expressions and Combinatory Categorial Grammars (CCGs) (Steedman, 1996) are used by Zettlemoyer and Collins (2005;2009), and ontology-based representations and Lexicalized Well-Founded Grammars (LWFGs) (Muresan and Rambow, 2007) are used by Muresan (2008).

An advantage of the LWFG formalism, compared to most constraint-based grammar formalisms developed for deep language understanding, is that it is accompanied by a learnability guarantee, the search space for LWFG induction being a complete grammar lattice (Muresan and Rambow, 2007). Like other constraint-based grammar formalisms, the semantic structures in LWFG are composed by constraint solving, semantic composition being realized through constraints at the grammar rule level. Moreover, semantic interpretation is also realized through constraints at the grammar rule level, providing access to meaning during parsing.

However, the learnability result given by Muresan and Rambow (2007) assumed that the grammar constraints were learnable. In this paper we present the properties and principles of the semantic representation and grammar formalism that allow us to learn the semantic composition constraints. These constraints are a simplified version of "path equations" (Shieber et al., 1983), and we present an algorithm for learning these constraints from examples (Section 5). We also present a LWFG parser as a deductive system (Shieber et al., 1995) (Section 3). The LWFG parser is used as an innate inference engine during LWFG learning, and we present an algorithm for learning LWFGs from examples (Section 4). A discussion and an example of learning a grammar for noun compounds are given is Section 6.

2 Lexicalized Well-Founded Grammars

Lexicalized Well-Founded Grammar (LWFG) is a recently developed formalism that balances expressiveness with provable learnability results (Muresan and Rambow, 2007). LWFGs are a type of Definite Clause Grammars (Pereira and Warren, 1980) in which (1) the context-free backbone is extended by introducing a partial ordering relation among nonterminals, 2) grammar nonterminals are augmented with strings and their syntactic-semantic representations, called *semantic molecules*, and (3) grammar rules can have

Coling 2008: Poster Volume, pages 885–893,
Beijing, August 2010

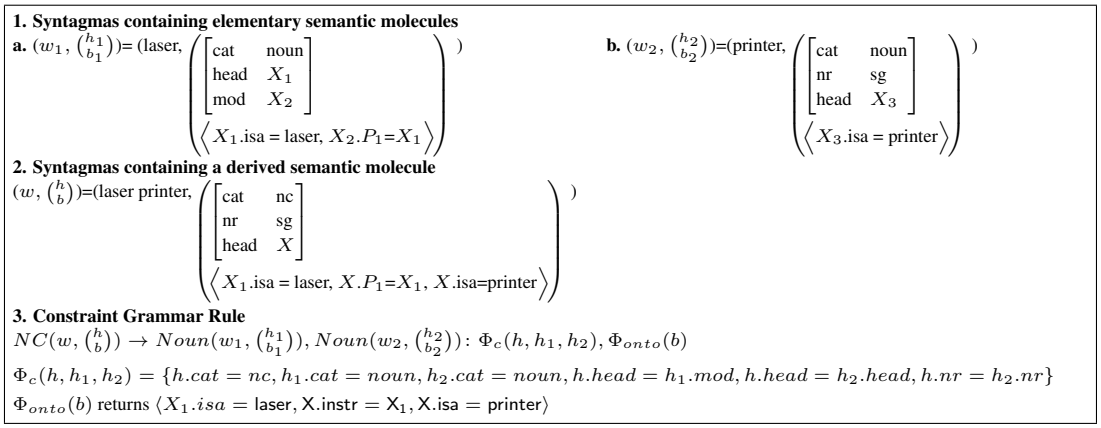

Figure 1: Syntagmas containing elementary semantic molecules (1) and a derived semantic molecule (2); A constraint grammar rule together with the semantic composition and ontology-based interpretation constraints, Φ_c and Φ_{onto} (3)

two types of constraints, one for semantic composition and one for semantic interpretation. The first property allows LWFG learning from a small set of examples. The last two properties make LWFGs a type of syntactic-semantic grammars.

Definition 1. *A semantic molecule associated with a natural language string w, is a syntactic-semantic representation, $w' = \binom{h}{b}$, where h (head) encodes compositional information, while b (body) is the actual semantic representation of the string w.*

Grammar nonterminals are augmented with pairs of strings and their semantic molecules. These pairs are called *syntagmas*, and are denoted by $\sigma = (w, w') = (w, \binom{h}{b})$.

Examples of semantic molecules for the nouns *laser* and *printer* and the noun-noun compound *laser printer* are given in Figure 1. When associated with lexical items, semantic molecules are called *elementary semantic molecules*. When semantic molecules are built by the combination of others, they are called *derived semantic molecules*. Formally, the semantic molecule head, h, is a one-level feature structure (i.e., values are atomic), while the semantic molecule body, b, is a logical form built as a conjunction of atomic predicates $\langle concept\rangle.\langle attr\rangle = \langle concept\rangle$, where variables are either concept or slot identifiers in an ontology.[1]

Muresan and Rambow (2007) formally defined LWFGs, and we present here a slight modification of their definition.

Definition 2. *A Lexicalized Well-Founded Grammar (LWFG) is a 7-tuple, $G = \langle \Sigma, \Sigma', N_G, \succeq, P_G, P_\Sigma, S\rangle$, where:*

1. *Σ is a finite set of terminal symbols.*
2. *Σ' is a finite set of elementary semantic molecules corresponding to the terminal symbols.*
3. *N_G is a finite set of nonterminal symbols. $N_G \cap \Sigma = \emptyset$. We denote $pre(N_G) \subseteq N_G$, the set of pre-terminals (a.k.a, parts of speech)*
4. *$\succeq$ is a partial ordering relation among nonterminals.*
5. *P_G is the set of constraint grammar rules. A constraint grammar rule is written $A(\sigma) \rightarrow B_1(\sigma_1), \ldots, B_n(\sigma_n)$: $\Phi(\bar{\sigma})$, where $A, B_i \in N_G$, $\bar{\sigma} = (\sigma, \sigma_1, ..., \sigma_n)$ such that $\sigma = (w, w'), \sigma_i = (w_i, w_i'), 1 \leq i \leq n, w = w_1 \cdots w_n, w' = w_1' \circ \cdots \circ w_n'$, and $\circ$ is the composition operator for semantic molecules (more details about the composition operator are given in Section 5). For brevity, we denote a rule by $A \rightarrow \beta$: Φ, where $A \in N_G, \beta \in N_G^+$. P_Σ is the set of constraint grammar rules whose left-hand side are pre-terminals, $A(\sigma) \rightarrow$, $A \in pre(N_G)$.*

[1]The body of a semantic molecule is called OntoSeR and is a flat ontology-based semantic representation.

We use the notation $A \rightarrow \sigma$ for this grammar rules. In LWFG due to partial ordering among nonterminals we can have ordered constraint grammar rules and non-ordered constraint grammar rules (both types can be recursive or non-recursive). A grammar rule $A(\sigma) \rightarrow B_1(\sigma_1), \ldots, B_n(\sigma_n)\colon \Phi(\bar{\sigma})$, is an ordered rule, if for all B_i, we have $A \succeq B_i$. In LWFGs, each nonterminal symbol is a left-hand side in at least one ordered non-recursive rule and the empty string cannot be derived from any nonterminal symbol.

6. *$S \in N_G$ is the start nonterminal symbol, and $\forall A \in N_G, S \succeq A$ (we use the same notation for the reflexive, transitive closure of $\succeq$).*

The partial ordering relation $\succeq$ makes the set of nonterminals well-founded[2] , which allows the ordering of the grammar rules, as well as the ordering of the syntagmas generated by LWFGs. This ordering allow LWFG learning from a small set of representative examples (Muresan and Rambow, 2007) (P_Σ is not learned).

An example of a LWFG rule is given in Figure 1(3). Nonterminals are augmented with syntagmas. Moreover, in LWFG the semantic composition and interpretation are realized via constraints at the grammar rule level ($\Phi(\bar{\sigma})$ in Definition 2). More precisely, syntagma composition means string concatenation ($w = w_1 w_2$) and semantic molecule composition ($\binom{h}{b} = \binom{h_1}{b_1} \circ \binom{h_2}{b_2}$) —- where the bodies of semantic molecules are concatenated through logical conjunction ($b = (b_1, b_2)\nu$, where ν is a variable substitution $\nu = \{X_2/X, X_3/X\}$), while the semantic molecules heads are composed through compositional constraints $\Phi_c(h, h_1, h_2)$, which are a simplified version of "path equations" (Shieber et al., 1983) (see Figure 1(3)). During LWFG learning, *compositional constraints Φ_c are learned* together with the grammar rules. Semantic interpretation, which is ontology-based in LWFG, is also encoded as constraints at the grammar rule level — Φ_{onto} — providing access to meaning during parsing. $\Phi_{onto}(b)$ constraints are applied to the body of the semantic molecule corresponding to the syntagma associated with the left-hand side nonterminal. *The ontology-based constraints are not learned*; rather, Φ_{onto} is a general predicate that succeed or fail as a result of querying an ontology — when it succeeds, it instantiates the variables of the semantic representation with concepts/slots in the ontology (see the example in Figure 1(3)).

[2] $\succeq$ should not be confused with information ordering derived from flat feature structures

2.1 Derivation in LWFG

The derivation in LWFG is called ground syntagma derivation, and it can be seen as the bottom up counterpart of the usual derivation. Given a LWFG, G, the *ground syntagma derivation* relation, $\stackrel{*G}{\Rightarrow}$, is defined as: $\frac{A \rightarrow \sigma}{A \stackrel{*G}{\Rightarrow} \sigma}$ (if $\sigma = (w, w'), w \in \Sigma, w' \in \Sigma'$, i.e., $A \in pre(N_G,)$, and $\frac{B_i \stackrel{*G}{\Rightarrow} \sigma_i,\ i=1,\ldots,n, \quad A(\sigma) \rightarrow B_1(\sigma_1),\ldots,B_n(\sigma_n)\colon \Phi(\bar{\sigma})}{A \stackrel{*G}{\Rightarrow} \sigma}$.

The *set of all syntagmas* generated by *a grammar G* is $L_\sigma(G) = \{\sigma | \sigma = (w, w'), w \in \Sigma^+, \exists A \in N_G, A \stackrel{*G}{\Rightarrow} \sigma\}$. Given a LWFG G, $E_\sigma \subseteq L_\sigma(G)$ is called a sublanguage of G. Extending the notation, given a LWFG G, the set of syntagmas generated by a *rule* $(A \rightarrow \beta\colon \Phi) \in P_G$ is $L_\sigma(A \rightarrow \beta\colon \Phi) = \{\sigma | \sigma = (w, w'), w \in \Sigma^+, (A \rightarrow \beta\colon \Phi) \stackrel{*G}{\Rightarrow} \sigma\}$, where $(A \rightarrow \beta\colon \Phi) \stackrel{*G}{\Rightarrow} \sigma$ denotes the ground derivation $A \stackrel{*G}{\Rightarrow} \sigma$ obtained using the rule $A \rightarrow \beta\colon \Phi$ in the last derivation step.

3 LWFG Parsing as Deduction

Following Shieber (1995), we present the Lexicalized Well-Founded Grammar parser as a deductive proof system in Table 1. The *items* of the logic are of the form $[i, j, \sigma_{ij}, A \rightarrow \alpha \bullet \beta \Phi^A]$, where $A \rightarrow \alpha\beta\colon \Phi^A$ is a grammar rule, Φ^A — the constraints corresponding to the grammar rule whose left-hand side nonterminal is A— can be true, $\bullet$ shows how much of the right-hand side of the rule has been recognized so far, i points to the parent node where the rule was invoked, and j points to the position in the input that the recognition has reached. We use the following notations: $\sigma^R_{ij} = (w^R_{ij}, \binom{h^R_{ij}}{b^R_{ij}})$ are syntagmas corresponding to the partially parsed right-hand side of a rule; $\sigma^L_{ij} = (w^L_{ij}, \binom{h^L_{ij}}{b^L_{ij}})$ are ground-derived syntagmas (i.e., they are augmenting the left-hand side non-

Item form	$[i, j, \sigma_{ij}, A \rightarrow \alpha \bullet \beta \Phi^A]$	$1 \leq i, j \leq n+1, A \in N_G, \alpha\beta \in N_G^*$ the Φ^A constraint can be true
Axioms	$[i, i+1, \sigma^L_{ii+1}, B_i \rightarrow \bullet]$	$1 \leq i \leq n, B_i \in pre(N_G), B_i \rightarrow \sigma^L_{ii+1} \in P_\Sigma$
Goals	$[i, j, \sigma^L_{ij}, A \rightarrow \alpha\Phi^A \bullet]$	$1 \leq i, j \leq n+1, A \in N_G, \alpha \in N_G^+$
Inference Rules		
Prediction	$\frac{[i,j,\sigma^L_{ij}, B \rightarrow \beta \Phi^B \bullet]}{[i,i,\sigma^R_{ii}, A \rightarrow \bullet B\gamma\Phi^A]}$ $\langle A \rightarrow B\gamma \colon \Phi^A \rangle$	$(A \rightarrow B\gamma \colon \Phi^A) \in P_G$ $\sigma^R_{ii} = \sigma_\emptyset$ (i.e., $w^R_{ii} = \epsilon$, $b^R_{ii} = true$ and $h^R_{ii} = \emptyset$)
Completion	$\frac{[i,j,\sigma^R_{ij}, A \rightarrow \alpha \bullet B\ \gamma\ \Phi^A] \quad [j,k,\sigma^L_{jk}, B \rightarrow \beta\ \Phi^B\ \bullet]}{[i,k,\sigma^R_{ik}, A \rightarrow \alpha\ B \bullet \gamma\ \Phi^A]}$	$\sigma^R_{ik} = \sigma^R_{ij} \circ \sigma^L_{jk}$, where $w^R_{ik} = w^R_{ij} w^L_{jk}$, $b^R_{ik} = b^R_{ij} b^L_{jk}$, $h^R_{ik} = h^R_{ij} \cup h^L_{jk}$
Constraint	$\frac{[i,j,\sigma^R_{ij}, A \rightarrow \alpha \bullet \Phi^A]}{[i,j,\sigma^L_{ij}, A \rightarrow \alpha\Phi^A \bullet]}$ $\langle \Phi^A$ is satisfiable $\rangle$	$\sigma^L_{ij} = \phi(\sigma^R_{ij})$

Table 1: LWFG parsing as deductive system

terminal of a LWFG rule). The goal items are of the form $[i, j, \sigma^L_{ij}, A \rightarrow \alpha\Phi^A \bullet]$, where σ^L_{ij} is ground-derived from the rule $A \rightarrow \alpha \colon \Phi^A$.

Compared to the deductive system in (Shieber et al., 1995), the LWFG parser has the following characteristics: each item is augmented with a syntagma; the *Constraint rule* is a *new inference rule*, and the goal items are associated to *every nonterminal in the grammar*, not only to the start symbol (i.e., LWFG parser is a robust parser). The **Constraint** inference rule is the only one that obtains an inactive edge[3], from an active edge by executing the grammar constraint Φ^A (the $\bullet$ is shifted across the constraint). By applying the Constraint rule as the last inference rule we obtain the ground-derived syntagmas σ^L_{ij}. Thus, the goal items are obtained only after the Constraint rule is applied. During this inference rule we have that $\sigma^L_{ij} = \phi(\sigma^R_{ij})$, where ϕ is defined by: $w^L_{ij} = w^R_{ij}$, $b^L_{ij} = b^R_{ij}\nu_{ij}$, and $h^L_{ij} = \varphi(h^R_{ij})$. The substitution ν_{ij} and the function φ are implicitly contained in the grammar constraint $\Phi^A_c(h^L_{ij}, h^R_{ij})$ (see Section 5 for details)

Definition 3 (Robust parsing provability)**.** *Robust parsing provability corresponds to reaching the goal item:* $\vdash_{rp} A(\sigma^L_{ij})$ *iff* $[i, j, \sigma^L_{ij}, A \rightarrow \alpha\Phi^A \bullet]$.

Thus, we can notice that the ground syntagma derivation is equivalent to robust parsing provability, i.e., $A \stackrel{*G}{\Rightarrow} \sigma$ iff $G \vdash_{rp} A(\sigma)$.

[3]We use Kay's terminology: items are edges, where the axioms and goals are inactive edges having $\bullet$ at the end, while the rest are active edges (Kay, 1986).

4 Learning LWFGs

The theoretical learning model for LWFG induction, Grammar Approximation by Representative Sublanguage (GARS), together with a learnability theorem was introduced in (Muresan and Rambow, 2007). LWFG's learning framework characterizes the "importance" of substructures in the model not simply by frequency, but rather linguistically, by defining a notion of "representative examples" that drives the acquisition process. Informally, representative examples are "building blocks" from which larger structures can be inferred via reference to a larger generalization corpus referred to as representative sublanguage in (Muresan and Rambow, 2007). The GARS model uses a polynomial algorithm for LWFG learning that take advantage of the building blocks nature of representative examples.

The LWFG induction algorithm belongs to the class of Inductive Logic Programming methods (ILP), based on entailment (Muggleton, 1995; Dzeroski, 2007). At each step a new constraint grammar rule is learned from the current representative example, σ. Then this rule is added to the grammar rule set. The process continues until all the representative examples are covered. We describe below the process of learning a grammar rule from the current representative example:

1. **Most Specific Grammar Rule Generation**. In the first step, the most specific grammar rule is generated from the current representative example σ. The category annotated

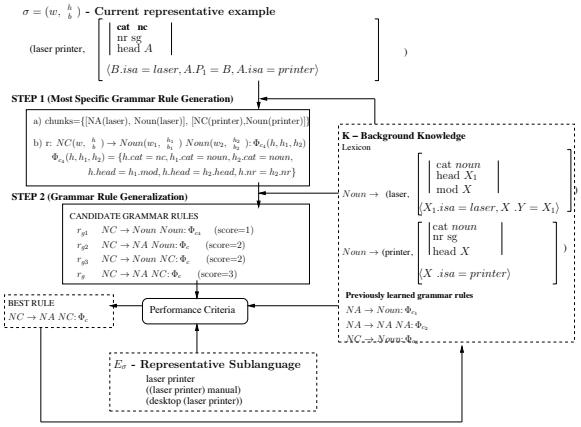

Figure 2: Example of Grammar Rule Learning

in the representative example gives the left-hand-side nonterminal, while a robust parser returns the minimum number of chunks covering the representative example. The categories of the chunks give the nonterminals of the right-hand side of the most specific rule. For example, in Figure 2, given the representative example *laser printer* annotated with its semantic molecule, and the background knowledge containing the already learned rules $NA \rightarrow Noun$: Φ_{c_1}, $NA \rightarrow NA\ NA$: Φ_{c_2}, $NC \rightarrow Noun$: Φ_{c_3} the robust parser generates the chunks corresponding to the noun *laser* and the noun *printer*: [NA(laser),Noun(laser)] and [NC(printer),Noun(printer)], respectively. The most specific rule is $NC \rightarrow Noun\ \ Noun$: Φ_{c_4}, where the left-hand side nonterminal is given by the category of the representative example, in this case nc. Compositional constraints Φ_{c_4} are learned as well. In section 5 we give the algorithm for learning these constraints, and several properties and principles that are needed in order for these constraints to be learnable.

2. **Grammar Rule Generalization**. In the second step, this most specific rule is generalized, obtaining a set of candidate grammar rules (the generalization step is the inverse of the derivation step used to define the complete grammar lattice search space in (Muresan and Rambow, 2007)). The performance criterion in choosing the best grammar rule among these candidate hypotheses is the number of examples in the representative sublanguage E_σ (generalization corpus) that can be parsed using the candidate grammar rule, r_{gi} in the last ground derivation step, together with the previous learned rules, i.e., $|E_\sigma \cap L_\sigma(r_{gi})|$. In Figure 2 given the representative sublanguage E_σ={ *laser printer, laser printer manual, desktop laser printer*} the learner will generalize to the recursive rule $NC \rightarrow NA\ NC$: Φ_7, since only this rule can parse all the examples in E_σ.

5 Learnable Composition Constraints

In LWFG, the semantic structures are composed by constraint solving, rather than functional application (with lambda expressions and lambda reduction). This section presents the properties and principles that guarantee the learnability of the compositional constraints,Φ_c, and presents an algorithm to generate these constraints from examples, which is a key result for LWFG learnability.

The information for semantic composition is encoded in the head of semantic molecules. There are three types of attributes that belong to the semantic molecule head h: category attributes $\mathcal{A}_h^c$, variable attributes $\mathcal{A}_h^v$, and feature attributes $\mathcal{A}_h^f$. Thus, $\mathcal{A}_h = \mathcal{A}_h^c \cup \mathcal{A}_h^v \cup \mathcal{A}_h^f$ and $\mathcal{A}_h^c, \mathcal{A}_h^v, \mathcal{A}_h^f$ are pairwise disjoint. For example, in Figure 1 for the noun-noun compound *laser printer*, we have that $\mathcal{A}_h^c = \{cat\}$, $\mathcal{A}_h^f = \{nr\}$, and $\mathcal{A}_h^v = \{head\}$, while for the noun *laser* we have that $\mathcal{A}_{h_1}^c = \{cat\}$, $\mathcal{A}_{h_1}^f = \emptyset$, and $\mathcal{A}_{h_1}^v = \{head, mod\}$ (nouns can be modifiers of other nouns, so their representation is similar to that of an adjective).

We describe in turn each of these types of attributes and their corresponding principles. All principles, except the first and the last mirror principles in other constraint-based linguistic formalisms, such as HPSG (Pollard and Sag, 1994).

The category attributes $\mathcal{A}_h^c$ are state attributes, and their value set gives the category of the semantic molecule. There is one attribute, cat $\in \mathcal{A}_h^c$, which is mandatory and whose value is the name of the category (e.g., $h.cat = nc$ in Figure

1). The category of a semantic molecule can be given by: 1) the cat attribute alone, or 2) the cat attribute together with other state attributes in $\mathcal{A}_h^c$ which are syntactic-semantic markers.

Principle 1 (Category Name Principle)**.** *The category name $h.cat$ of a syntagma $\sigma = (w, \binom{h}{b})$ is the same as the grammar nonterminal augmented with syntagma σ.*

When learning a LWFG rule from an example σ, the above principle allows us to determine the nonterminal in the left-hand side of the grammar rule. For example, when learning the LWFG rule from the syntagma corresponding to *laser printer* in Figure 2, the nonterminal in the left-hand side of the LWFG rule is NC since $h.cat = nc$.

The variable attributes $\mathcal{A}_h^v$ are attributes whose values are logical variables and represent the semantic valence of the molecule, which allows the binding of the semantic representations. These logical variables appear in the semantic molecule body as well. For example, in Figure 1(2) for the noun-noun compound *laser printer*, the value of the variable attribute $head \in \mathcal{A}_h^v$ is a variable X, which appears also in the body of the semantic molecule $\langle X_1.isa = laser, X.P_1 = X_1, X.isa = printer\rangle$. It can be noticed that the semantic molecule body contains other variables as well (X_1, P_1). However, only the variables present in the semantic molecule head as well (X) will participate in further composition.

Principle 2 (Semantic Representation Binding Principle)**.** *All the logical variables that the body b of a semantic molecule corresponding to a syntagma $\sigma = (w, \binom{h}{b})$, share with other syntagmas, are at the same time values of the variable attributes ($\mathcal{A}_h^v$) of the semantic molecule head.*

There is one variable attribute, $head \in \mathcal{A}_h^v$ that represents the head of a syntagma, giving the following principle:

Principle 3 (Semantic Head Principle)**.** *Given a syntagma $\sigma = (w, \binom{h}{b})$ ground derived from a grammar rule, r, there exists one and only one syntagma $\sigma_i = (w_i, \binom{h_i}{b_i})$ corresponding to a nonterminal B_i in rule r's right-hand side, which has the same value of the attribute $head$, i.e., $h.head = h_i.head$.*

The feature attributes $\mathcal{A}_h^f$ are the attributes whose values express the specific properties of the semantic molecules (e.g., number, person).

Principle 4 (Feature Inheritance Principle)**.** *If $\sigma_i = (w_i, \binom{h_i}{b_i})$ is the semantic head of a ground-derived syntagma $\sigma = (w, \binom{h}{b})$, then all feature attributes of σ inherit the values of the corresponding attributes that belong to the semantic head σ_i. That is, if $h.head = h_i.head$, then $h.f = h_i.f$, $\forall f \in \mathcal{A}_h^f \cap A_{h_i}^f$.*

Besides this principle, the feature attributes are used for category agreement. The categories that enter in agreement are maximum projection categories. This linguistic knowledge about agreement is used in the form of the following principle:

Principle 5 (Feature Agreement Principle)**.** *The agreeing categories and the agreement features are a-priori given based on linguistic knowledge, and are applied only at the semantic head level.*

Given all the above principles, we can now formulate the general Composition Principle:

Principle 6 (Composition Principle)**.** *A syntagma $\sigma = (w, w')$ corresponding to the left-hand side nonterminal of a grammar rule is obtained by string concatenation ($w = w_1 \dots w_n$) and the composition of semantic molecules corresponding to the nonterminals from the rule right-hand side:*

$$w' = \binom{h}{b} = (w_1 \cdots w_n)' = w_1' \circ \cdots \circ w_n'$$
$$= \binom{h_1}{b_1} \circ \cdots \circ \binom{h_n}{b_n} = \binom{h_1 \circ \cdots \circ h_n}{\langle b_1, \dots, b_n\rangle\nu}$$

The composition of the semantic molecule bodies is realized through conjunction after the application of a variable substitution ν. The body variable specialization substitution ν is the most general unifier (mgu) of b and $b_1, \dots, b_n$, s.t $b = (b_1, \dots, b_n)\nu$. It is a particular form of the commonly used substitution (Lloyd, 2003), i.e., a finite set of the form $\{X_1/Y_1, \dots, X_m/Y_m\}$, where $X_1, \dots, X_m, Y_1, \dots, Y_m$ are variables, and $X_1, \dots, X_m$ are distinct.

The composition of the semantic molecule heads is realized by a set of constraints $\Phi_c(h, h_1 ..., h_n)$, which is a system of equations

similar to "path equations" (Shieber et al., 1983; van Noord, 1993), but applied to flat feature structures:

$$\left\{\begin{array}{l} h_i.c = ct \\ h_i.v_i = h_j.v_j \\ h_i.f = ct\ or \\ h_i.f = h_j.f \end{array}\right\} \text{ where } \begin{array}{l} 0 \le i,j \le n, i \ne j \\ c \in \mathcal{A}^c_{h_i} \\ v_i \in \mathcal{A}^v_{h_i}, v_j \in \mathcal{A}^v_{h_j} \\ f \in \mathcal{A}^f_{h_i}, f \in \mathcal{A}^f_{h_j} \end{array}$$

When learning a LWFG rule from a representative example σ as in Figure 2, the robust parser returns the minimum number of chunks, n, covering σ. The body variable substitution ν is fully determined by the representative example as mgu of b and $b_1, \ldots, b_n$, and the compositional constraints $\Phi_c(h, h_1, \ldots, h_n)$ are learned using Alg 1. For example, in Figure 2, when learning from the representative example corresponding to the string *laser printer*, we have that $\nu = \{X_1/B, X_2/A, X_3/A, Y/P_1\}$.

In Alg 1 we use the notation $\sigma_0 = (w_0, \binom{h_0}{b_0})$ to denote the representative example σ.

Alg 1: Learn_Constraints($\sigma_0, \sigma_1, \ldots, \sigma_n$)

$\sigma_i = (w_i, \binom{h_i}{b_i}), 0 \le i \le n$
$\Phi_c \leftarrow \emptyset$
$\nu \leftarrow mgu(b_0, (b_1, \ldots, b_n))$
1 **foreach** $0 \le i \le n \wedge c \in \mathcal{A}^c_{h_i}$ **do**
  if $h_i.c = c1$ **then**
    $\Phi_c \leftarrow \Phi_c \cup \{h_i.c = c1\}$
2 **foreach** $0 \le i,j \le n \wedge i \ne j \wedge X/Y \in \nu \wedge$ $v_i \in \mathcal{A}^v_{h_i} \wedge v_j \in \mathcal{A}^v_{h_j}$ **do**
  if $h_i.v_i = X \wedge h_j.v_j = Y$ **then**
    $\Phi_c \leftarrow \Phi_c \cup \{h_i.v_i = h_j.v_j\}$
3 **if** $h_s.head = h_0.head, 1 \le s \le n$ **then**
  foreach $f \in \mathcal{A}^f_{h_0} \cap \mathcal{A}^f_{h_s}$ **do**
    if $h_0.f = c1 \wedge h_s.f = c1$ **then**
      $\Phi_c \leftarrow \Phi_c \cup \{h_0.f = h_s.f\}$
  if $h_s.cat = c_s \wedge h_i.cat = c_i \wedge agr(c_s, c_i)$, $1 \le i \le n$ **then**
    foreach $f \in agrFeatures(c_s, c_i)$ **do**
      if $h_s.f = c1 \wedge h_i.f = c1$ **then**
        $\Phi_c \leftarrow \Phi_c \cup \{h_s.f = h_i.f\}$
4 **for** *all other* $f \in \mathcal{A}^f_{h_i}, 0 \le i \le n$ **do**
  /*i.e., if we are not in case 3 */
  if $h_i.f = c1$ **then**
    $\Phi_c \leftarrow \Phi_c \cup \{h_i.f = c1\}$
return Φ_c /*i.e., $\Phi_c(h_0, h_1, \ldots, h_n)$ */

In the first step, the constraints corresponding to category attributes are fully determined by the values of these attributes that appear in the semantic molecule heads of $\sigma_0, \ldots \sigma_n$. In Figure 2, when learning the most specific rule r from the representative example *laser printer*, the set of constraints $\{h.cat = nc, h_1.cat = noun, h_2 = noun\} \subset \Phi_{c_4}$ are the constraints corresponding to category attributes. In the second step, the constraints corresponding to variable attributes are fully determined by the variables in the substitution ν that also appear as values of variable attributes $h_i.v_i, h_j.v_j$, where $0 \le i,j \le n$ and $i \ne j$. In Figure 2, only $\{X_2/A, X_3/A\} \subset \nu$ will be used, generating the set of constraints $\{h.head = h_1.mod, h.head = h_2.head\} \subset \Phi_{c_4}$. In the third step, the values of the feature attributes which obey Principles 4 and 5 are generalized — agr(c_s, c_i) is the predicate which gives us the agreement between the categories c_s and c_i (e.g., the subject agrees with the verb), and agrFeatures(c_s, c_i) gives us the set of feature attributes that participate in agreement (e.g., nr, pers, case). In Figure 2, the set of constraints $\{h.nr = h_2.nr\} \subset \Phi_{c_4}$ represents the generalization of the feature attribute values for nr, using Principle 4 . For all features attributes besides the ones that obey the above two principles, the generated constraints keep the particular values of these attributes (step 4 of Alg 1).

6 Examples

The LWFG formalism allows us to learn grammars for deep language understanding from examples. Instead of writing syntactic-semantic grammar by hand (both rules and constraints), we need to provide only a small set of representative examples — strings and their semantic molecules. Qualitative experiments on learning LWFGs showed that complex linguistic constructions can be learned and covered, such as complex noun phrases, relative clauses and reduced relative clauses, finite and non-finite verbal constructions (including, tense, aspect, negation, and subject-verb agreement), and raising and control constructions (Muresan and Rambow, 2007). In Figure 3 we show an example of learning a LWFG grammar for noun-noun compounds. The first four examples (1-4) are representative examples, while the last four examples are used for gener-

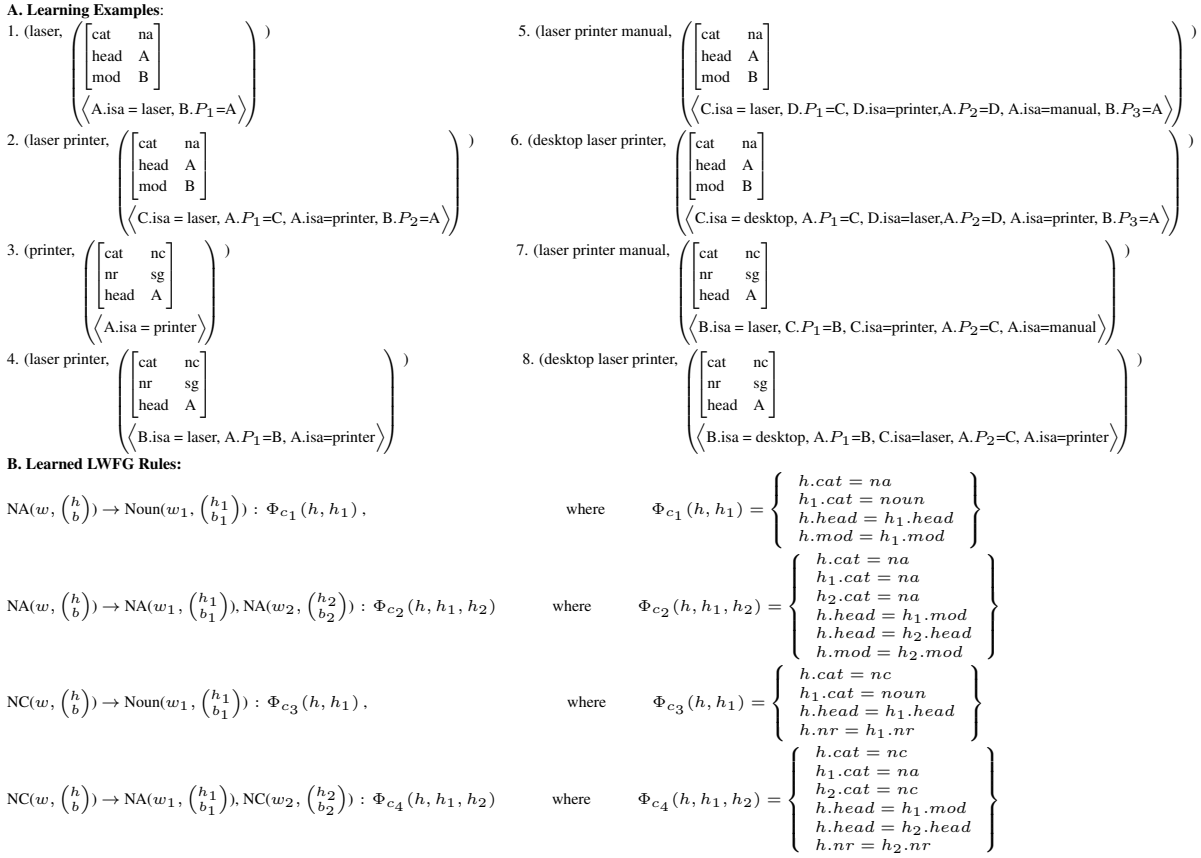

Figure 3: Learning LWFG Rules for Noun-Noun Compounds

alization (5-8). The learned grammar rules, including the learned composition constraints are also shown. The first two LWFG rules ground derive syntagmas for noun adjuncts, while the last two rules ground derive syntagmas for noun compounds. For example, "desktop laser printer" can be either a fully-formed noun compound (category nc), or it can be further combined with the noun "invoice" to obtain "desktop laser printer invoice", case in which it is a noun adjunct (category na). The learned rule for noun adjuncts is both left and right recursive, accounting for both left and right-branching noun compounds. Even though we can obtain overgeneralization in syntax, the ontology-based interpretation constraint at the rule level will prune some erroneous parses. Preliminary results in the medical domain show that Φ_{onto} can help remove erroneous parses even when using just a weak ontological model (semantic roles of verbs, prepositions, attributes of adjectives and adverbs, but no synonymy, or hierarchy of concepts or roles). However, more experiments need to be run for reporting quantitative results.

7 Conclusions

We have presented the properties and principles that the semantic representation integrated in LWFG requires so that the semantic compositional constraints are learnable from examples. These properties together with Alg 1 give a theoretical result that in conjunction with the learnability result of Muresan and Rambow (2007) show that LWFG is a learnable constraint-based grammar formalism that can be used for deep language understanding. Instead of writing grammar rules and constraints by hand, one needs to provide only a small set of annotated examples.[4]

[4]The author acknowledges the support of the NSF (SGER grant IIS-0838801). Any opinions, findings, or conclusions are those of the author, and do not necessarily reflect the views of the funding organization.

References

Dzeroski, Saso. 2007. Inductive logic programming in a nutshell. In Getoor, Lise and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. The MIT Press.

Ge, Ruifang and Raymond J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of CoNLL-2005*.

Kay, M. 1986. Algorithm schemata and data structures in syntactic processing. In *Readings in natural language processing*, pages 35–70. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Lloyd, John W. 2003. *Logic for Learning: Learning Comprehensible Theories from Structured Data*. Springer, Cognitive Technologies Series.

Muggleton, Stephen. 1995. Inverse Entailment and Progol. *New Generation Computing, Special Issue on Inductive Logic Programming*, 13(3-4):245–286.

Muresan, Smaranda and Owen Rambow. 2007. Grammar approximation by representative sublanguage: A new model for language learning. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Muresan, Smaranda. 2008. Learning to map text to graph-based meaning representations via grammar induction. In *Coling 2008: Proceedings of the 3rd Textgraphs workshop on Graph-based Algorithms for Natural Language Processing*, pages 9–16, Manchester, UK, August. Coling 2008 Organizing Committee.

Neumann, Günter and Gertjan van Noord. 1994. Reversibility and self-monitoring in natural language generation. In Strzalkowski, Tomek, editor, *Reversible Grammar in Natural Language Processing*, pages 59–96. Kluwer Academic Publishers, Boston.

Pollard, Carl and Ivan Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, Illinois.

Shieber, Stuart, Hans Uszkoreit, Fernando Pereira, Jane Robinson, and Mabry Tyson. 1983. The formalism and implementation of PATR-II. In Grosz, Barbara J. and Mark Stickel, editors, *Research on Interactive Acquisition and Use of Knowledge*, pages 39–79. SRI International, Menlo Park, CA, November.

Shieber, Stuart, Yves Schabes, and Fernando Pereira. 1995. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24(1-2):3–36.

Steedman, Mark. 1996. *Surface Structure and Interpretation*. The MIT Press.

van Noord, Gertjan. 1993. *Reversibility in Natural Language Processing*. Ph.D. thesis, University of Utrecht.

Wong, Yuk Wah and Raymond Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007)*.

Zettlemoyer, Luke S. and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of UAI-05*.

Zettlemoyer, Luke and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. In *Proceedings of the Association for Computational Linguistics (ACL'09)*.