Canonical Context-Free Grammars and Strong Learning: Two Approaches

Alexander Clark
Department of Philosophy,
King's College London
alexander.clark@kcl.ac.uk

Abstract

Strong learning of context-free grammars is the problem of learning a grammar which is not just weakly equivalent to a target grammar but isomorphic or structurally equivalent to it. This is closely related to the problem of defining a canonical grammar for the language. The current proposal for strong learning of a small class of CFGs uses grammars whose nonterminals correspond to congruence classes of the language, in particular to a subset of those that satisfy a *primality* condition. Here we extend this approach to larger classes of CFGs where the nonterminals correspond instead to closed sets of strings; to elements of the syntactic concept lattice. We present two different classes of canonical context-free grammars. One is based on all of the primes in the lattice: the other, more suitable for strong learning algorithms is based on a subset of primes that are irreducible in a certain sense.

1 Introduction

This paper is concerned with the problem of strong learning of context-free grammars in the distributional framework. One approach, initiated in (Clark, 2014) is to develop strong learning algorithms by defining canonical grammars based on properties of algebraic structures associated with the language: specifically the syntactic monoid of the language. In that paper a strong learning result was presented for a subclass of the class of substitutable languages, languages which have a simple language theoretic closure property.

In this paper we will extend the canonical grammar ideas to a larger class of grammars, while not presenting a full strong learning result, for reasons of space and some technical details not yet resolved. Rather than using the syntactic monoid, we use the syntactic concept lattice (SCL), (Clark, 2013), a richer structure that is suitable for modeling all context-free grammars. In the case of substitutable languages the syntactic monoid is almost identical to the syntactic concept lattice.

We want these canonical grammars to be as unambiguous as possible, and to use as few nonterminals as possible. These two obvious principles pull in the same direction: a grammar with extra nonterminals will typically have extra derivations and thus a higher degree of ambiguity. Finding some global minimum leads in general to intractable computational problems – the set covering problem, a classic NP-hard problem – and the answer may be indeterminate (in that there may be two structurally distinct minima). So rather we stipulate some technical notion which is more determinate, and can be efficiently identified (though we do not talk about the algorithmic issues here). In particular, we want the grammars defined to be compatible with efficient learning algorithms for context-free grammars (Yoshinaka, 2012a; Leiß, 2014).

In the case of the monoid, we only have one operation, concatenation, and given a derivation tree with unlabeled interior nodes, each node in the tree can only be legally labeled with the unique congruence class of the yield of the subtree. Thus given the unlabeled trees, the labeling is determined.

In the case of the SCL, we have a lattice struc-

Proceedings of the 14th Meeting on the Mathematics of Language (MoL 14), pages 99–111,
Chicago, USA, July 25–26, 2015.

ture, and so there are many different possible ways of modeling the structure, and many different ways of labeling a given tree from the very specific to the very general. We previously argued in (Clark, 2011) that the most general labelings would be optimal; that view now seems simplistic.

We argue that we should only model those unpredictable parts of the structure, that is to say those places where the structure differs from the free structure $\mathcal{P}(\Sigma^*)$. The grammar does not need to state that $\{u\} \circ \{v\} = \{uv\}$ or that $\{u\} \cup \{v\} = \{u, v\}$: these are true in the free structure. It is only when these are not equal that we need to represent the difference.

We will give definitions of the basic mathematical concepts we use in Section 2, including a brief introduction to the syntactic concept lattice in Section 2.4 to make the paper self-contained.

In Section 3 we explain the relation between strong learning and canonical grammars.

Then in Section 4 we extend the definition of primes from congruence classes to closed sets of strings. Section 5 presents our first family of canonical grammars that are based directly on all of the primes in the language.

In the case of concepts the lattice structure means that there may be many different concepts that contain a given string, and so in Section 6 we discuss how to exploit the lattice structure to select a smaller set of categories that are irreducible in some sense; and then in Section 7 we present a second family of canonical grammars based on this restricted subset of the primes. We finish with a worked example to illustrate the abstract mathematical discussion and some discussion.

2 Preliminaries

2.1 Strings, Languages and Contexts

We assume a fixed alphabet Σ and write Σ^* for the set of strings. A language is a subset of Σ^*, we write concatenation of languages L, M as $L \cdot M$, or sometimes just LM. The empty string is λ. We take a symbol $\Box \notin \Sigma$, and using this we define a context as an element of $\Sigma^* \Box \Sigma^*$, written $l \Box r$. We define $\odot$ as $l \Box r \odot w = lwr$, and extend these to sets of strings and contexts in the usual way. The empty context $\lambda \Box \lambda = \Box$ is particularly important: of course $\Box \odot w = w$.

2.2 Grammars

We define CFGs standardly as a tuple $\langle \Sigma, V, S, P \rangle$ where $S \in V$ is a single start symbol, V is the set of nonterminals and P is a finite subset of $V \times (\Sigma \cup V)^*$, written as $N \to \alpha$. The derivation process is denoted by $\Rightarrow$ and $\Rightarrow^*$. We define

$$\mathcal{L}(G, N) = \{w \in \Sigma^* \mid N \Rightarrow_G^* w\}$$

and define $\mathcal{L}(G) = \mathcal{L}(G, S)$. We also define the set of derivation contexts:

$$\mathcal{C}(G, N) = \{l \Box r \in \Sigma^* \Box \Sigma^* \mid S \Rightarrow_G^* lNr\}$$

The following property corresponds to the *context-free* property of the derivation process:

$$\mathcal{C}(G, N) \odot \mathcal{L}(G, N) \subseteq \mathcal{L}(G).$$

Two grammars $G_1 = \langle \Sigma, V_1, S_1, P_1 \rangle$, $G_2 = \langle \Sigma, V_2, S_2, P_2 \rangle$ are weakly equivalent if $\mathcal{L}(G_1) = \mathcal{L}(G_2)$. They are isomorphic if there is a bijection $\phi : V_1 \to V_2$, such that $\phi(S_1) = S_2$ and $\phi(P_1) = P_2$, extending ϕ to productions and sets of productions in the natural way. Isomorphic grammars are identical except for a relabeling of the nonterminal symbols. Clearly isomorphism implies weak equivalence.

2.3 Lattices

We assume some familiarity with lattices: see (Davey and Priestley, 2002) for basic definitions. We write $\top, \bot, \vee, \wedge$ as standard. An element x is join-irreducible iff $x = a \vee b$ implies $a = x$ or $b = x$; dually it is meet irreducible iff $x = a \wedge b$ implies $a = x$ or $b = x$. For some lattices, the set of join irreducible elements and the set of meet irreducible elements can form a "basis" for the lattice, in that every element can be represented as a finite join of join-irreducible elements and/or a finite meet of meet-irreducible elements. In the lattice of all subsets of Σ^*, $\mathcal{P}(\Sigma^*)$ the join irreducible sets are the singleton sets $\{w\}$ for any string, and the meet irreducible sets are $\Sigma^* \setminus \{w\}$.

A descending chain is a strictly descending sequence of elements of a lattice $X_0 \supset X_2 \supset \ldots X_n$. A lattice satisfies the descending chain condition (DCC) if there are no infinite descending chains. If a lattice satisfies the DCC, then every nonempty subset has at least one minimal element. We define the ascending chain condition (ACC) dually.

2.4 The Syntactic Concept Lattice

We now describe the syntactic concept lattice briefly; for fuller descriptions see e.g. (Clark, 2013; Leiß, 2014; Wurm, 2012). Given a fixed language L, we have a Galois connection between sets of strings and sets of contexts defined, where S is a set of strings and C is a set of contexts:

$$S^{\triangleright} = \{l \Box r \mid l \Box r \odot S \subseteq L\}$$

and

$$C^{\triangleleft} = \{w \in \Sigma^* \mid C \odot w \subseteq L\}$$

.

A closed set of strings is a set of strings S such that $S = S^{\triangleright\triangleleft}$, a closed set of contexts is one such that $C = C^{\triangleleft\triangleright}$. A concept is an ordered pair $\langle S, C\rangle$ such that $S = C^{\triangleleft}$ and $C = S^{\triangleright}$. In this case both S and C are closed. We will therefore often refer to a concept through the corresponding closed set of strings. Note that for any such concept, the following property holds, which corresponds to the context-free property of the CFG derivations:

$$C \odot S \subseteq L$$

Clearly $w \in L$ iff $\Box \in \{w\}^{\triangleright}$, and so L is closed.

The Syntactic Concept Lattice of L, written $\mathfrak{B}(L)$ is the collection of concepts, with the following constants, relations and operations, which we define in terms of the closed sets of strings alone: $S \vee T = (S \cup T)^{\triangleright\triangleleft}, S \wedge T = S \cap T$, $S \circ T = (S \cdot T)^{\triangleright\triangleleft}$, $S \leq T$ iff $S \subseteq T$, $\top = \Sigma^*$, $\bot = \emptyset^{\triangleright\triangleleft}$. With these operations $\mathfrak{B}(L)$ is a complete idempotent semiring, and furthermore a complete residuated lattice.

This lattice forms a hierarchy of all distributionally definable sets of strings in the language. There will be a finite number of elements iff the language is regular. Minimal grammars will have nonterminals that correspond to elements of the syntactic concept lattice as shown by (Clark, 2013). Given a context-free grammar G such that $\mathcal{L}(G) = L$, we define a universal morphism $h_L : V \to \mathfrak{B}(L)$ given by $h_L(N) = \mathcal{L}(G, N)^{\triangleright\triangleleft}$. We extend this to a CFG-morphism in the obvious way. (Clark, 2013) proved that for all CFGs $\mathcal{L}(h_L(G)) = L$. Therefore any CFG for L can be mapped to a possibly smaller grammar whose nonterminals are elements of $\mathfrak{B}(L)$. We can therefore assume that the nonterminals of the grammar are elements of $\mathfrak{B}(L)$.

3 Weak and Strong Learning

We will not present any learning algorithms here, but the work is motivated by learning considerations and so we need to make the background assumptions clear. In standard models of learning, there is a target grammar G_* and the learner, using information only about $\mathcal{L}(G_*)$, must eventually return a grammar $\hat{G}$ such that $\mathcal{L}(\hat{G}) = \mathcal{L}(G_*)$. In strong learning (Clark, 2014) in contrast, given the same information source, the learner must pick a $\hat{G}$ such that $\hat{G}$ is isomorphic to G_*.

(Clark, 2014) observes that the existence of a canonical grammar is a necessary condition for a strong learning algorithm. Any strong learning algorithm will implicitly define a canonical grammar for any language in the class of languages that it learns. Much of that paper is in fact concerned with precisely that definition. Accordingly in this paper we focus on defining a canonical grammar rather than directly presenting a learning algorithm.

The universal property of the syntactic concept lattice is an important tool. This means that rather than dealing directly with CFGs which are arbitrary and intractable we can deal with the lattices $\mathfrak{B}(L)$ which have nice mathematical properties. We can assume without loss of generality that the nonterminals of the grammar will correspond to concepts or closed sets of strings: to elements of the lattice. Given this, there is a natural notion of a production being *correct*: $X \to w$ is correct if $w \in X$, $X \to Y_1 \dots Y_n$ is correct if $X \supseteq Y_1 \cdot \dots \cdot Y_n$.

Given that a language that is not regular will have an infinite number of concepts, we need a principled way of selecting a finite number of these in an appropriate way so that we have a finite grammar. The general approach we take is to identify some elements that are irreducible in some sense with respect to the algebraic structure of the residuated lattice.

In the case of substitutable grammars, the closed sets of strings are almost exactly the congruence classes—except for $\top, \bot$ and $\{\lambda\}^{\triangleright\triangleleft}$, every closed set of strings is either equal to a congruence class or a congruence class together with λ. There seems to be only one plausible way to define a grammar, given that the mathematical structure of the congruence classes is just a monoid. Since this structure is so simple, there is only one reasonable irreducibility

property that we can use to select from the congruence classes: primality, which we define later. In the case of general CFLs things are unsurprisingly much more complicated. There seem to be two different factors to be considered. One factor concerns, as in the case of the monoid, the concatenation structure of the strings—the monoid structure of $\mathfrak{B}(L)$—and the other concerns the partial order: the lattice structure of $\mathfrak{B}(L)$.

We start by discussing the concatenation structure in Section 4 and discuss the lattice structure later in Section 6.

4 Primes

Since a monoid is a very simple algebraic structure, with a single associative binary operation, there is only one reasonable technique to define a subset of elements of the syntactic monoid in such a way that the grammar based on those elements is well behaved. (Clark, 2014) argues that we should represent only those elements where concatenation differs from the free operation of concatenation: in other words where $[uv] \supset [u][v]$. Language theoretically these represent places where the monoid has some nontrivial structure and grammatically they provide evidence for a nontrivial nonterminal: a nonterminal which occurs on the left hand side of more than one production. Congruence classes which have this desirable property are called *primes*.

For a congruence class the definition of a prime is straightforward. If a nonzero nonunit[1] congruence class X has a nontrivial decomposition into two congruence classes Y, Z such that $X = Y \cdot Z$ then it is composite. The trivial decompositions are $X = X \cdot [\lambda] = [\lambda] \cdot X$, where $[\lambda] = \{w \in \Sigma^* \mid \{w\}^{\triangleright} = \{\lambda\}^{\triangleright}\}$.

In the case of a CFL which is not substitutable, we need a different criterion, since we may use concepts that are not congruence classes but unions of congruence classes. This is complicated by the fact that the empty string may occur in many different closed sets of strings.

Definition 1. *$X \in \mathfrak{B}(L_*)$ is composite if there are $Y, Z \in \mathfrak{B}(L)$ such that $X \neq Y$ and $X \neq Z$ and $X = Y \cdot Z$. An element of $\mathfrak{B}(L_*)$ is prime if it is not composite. We write $\mathfrak{P}(L_*)$ for the set of primes of a language L_*. The unit prime is $\{\lambda\}^{\triangleright\triangleleft}$.*

We define $\mathfrak{P}(L_*)_\lambda = \mathfrak{P}(L_*) \setminus \{\{\lambda\}^{\triangleright\triangleleft}\}$, the set of nonunit primes.

Note that here we do not exclude $\{\lambda\}^{\triangleright\triangleleft}$. Clearly for any closed set of strings $X = X \cdot \{\lambda\}^{\triangleright\triangleleft} = \{\lambda\}^{\triangleright\triangleleft} \cdot X$. For the condition to be nontrivial, we clearly need to exclude such cases, but we also want to exclude cases such as $a^* = a^* \cdot a^*$ and $a^*b^* = (a^*b^*) \cdot b^*$.

Lemma 1. *For every $a \in \Sigma$, $\{a\}^{\triangleright\triangleleft}$ is prime.*

Proof. Supppose $\{a\}^{\triangleright\triangleleft} = B \cdot C$. Since $a \in \{a\}^{\triangleright\triangleleft}$ either $a \in B$ and $\lambda \in C$ or vice versa. Assume the former. Since $a \in B$, this means that $B \supseteq \{a\}^{\triangleright\triangleleft}$ and since $\lambda \in C$, this means that $B \subseteq \{a\}^{\triangleright\triangleleft}$. Therefore $B = \{a\}^{\triangleright\triangleleft}$, or, by a similar argument $C = \{a\}^{\triangleright\triangleleft}$. Therefore it is prime. □

Lemma 2. *$\{\lambda\}^{\triangleright\triangleleft}$ is prime.*

Proof. Suppose $\{\lambda\}^{\triangleright\triangleleft} = X \cdot Y$. Clearly $\lambda \in X, \lambda \in Y$. Therefore $X \subseteq \{\lambda\}^{\triangleright\triangleleft}$ and $Y \subseteq \{\lambda\}^{\triangleright\triangleleft}$. But if X, Y are in $\mathfrak{B}(L_*)$, they must both be equal to $\{\lambda\}^{\triangleright\triangleleft}$ since that is the smallest concept that contains λ. □

Definition 2. *If $X \in \mathfrak{B}(L)$ and $\alpha \in \mathfrak{P}(L)^+$, a nonempty string of primes, we write $\bar{\alpha}$ for the concatenation of the primes in α. So if $\alpha = \langle A_1, \ldots, A_n \rangle$ then $\bar{\alpha} = A_1 \cdot \cdots \cdot A_n$. We say that α is a prime decomposition of X iff $\bar{\alpha} = X$, and none of the elements of α are unit. In the special case where $X = \{\lambda\}^{\triangleright\triangleleft}$ we consider $\langle \{\lambda\}^{\triangleright\triangleleft} \rangle$ to be a prime decomposition.*

Example 1. *If $L = \{a^n b^n \mid n > 0\}$, then the primes are $\{a\}, \{b\}$ and $L' = L \cup \{\lambda\}$, together with $\top, \bot$ and $\{\lambda\}^{\triangleright\triangleleft}$. L has the prime decomposition $\langle \{a\}, L', \{b\} \rangle$.*

We need to consider two cases: one where a prime contains λ and one where it does not. If a closed set of strings contains the empty string, that means that it represents an optional category; it can be replaced by the empty string. If X is a closed set of strings that contains λ and $X = Y \cdot Z$, then clearly $\lambda \in Y \cap Z$ and $Y \subseteq X$ and $Z \subseteq X$. Therefore a decomposition of a concept that contains λ will be into

[1] The zero congruence class, if it exists, is $\emptyset^{\triangleleft}$ and the unit is $[\lambda]$.

proper subsets of that concept. Decompositions using concepts with λ may not terminate, if the lattice has infinite descending chains.

Example 2. *Let $L_1 = (ba)^*$, and $L_n = L_{n-1} \cdot (ba^n)^*$. Consider the language $L = \bigcup_n L_n$. This is a closed set of strings with an infinite descending chain $L \supset L \setminus L_1 \supset L \setminus (L_1 \cup L_2) \cdots$. For each n, $(ba^n)^*$ is closed and prime, and L has no finite prime decomposition.*

Lemma 3. *If $\mathfrak{B}(L_*)$ satisfies DCC then every element has a prime decomposition.*

Proof. If X is prime, then it has a length one decomposition, $\langle X \rangle$. Define the width of a non empty set of strings to be the minimum length of a string in the set. If it contains the empty string, then the width is zero.

Let M be the set of all non-zero non-unit concepts without prime decompositions. Suppose it is nonempty; then it has at least one minimal element, by the DCC. Take a minimal element of minimal width, X. Suppose X has width n. It is not prime by assumption and so $X = Y \cdot Z$. Case 1: width of X is zero and so both Y and Z contain the empty string: therefore Y, Z are both proper subsets of X. Therefore they are not in M (since X is minimal). Moreover they are not zero or unit, therefore they have prime decompositions such that $Y = \bar{\alpha}$ and $Z = \bar{\beta}$ and therefore $\alpha\beta$ is a decomposition of X. Case 2: the width of X is greater than zero; and the width of Y and Z are both less than width of X. Then Y and Z are both not in M and therefore both have prime decompositions and therefore so does X. Case 3: the width of X is greater than zero, and one of Y or Z had width zero. Assume that $\lambda \in Y$ (the other case is identical), then Z is a proper subset of X and therefore has a prime decomposition, and Y has width less than X and is therefore also has a prime decomposition. □

These decompositions aren't necessarily unique, but this lemma shows that the set of primes is sufficiently large to express any concept we want through finite concatenations.

It is not the case that every closed set of strings that is composite has a unique prime decomposition; even if we restrict ourselves to maximal decompositions: decompositions where no element can be replaced with a larger one. Clearly we can decompose, for example a^* into $\{\lambda, a^k\} \cdot (a^* \setminus \{a^k\})$ for any k, and for a suitable language these can be prime.

Example 3. *Consider the language $L = \{a, aa\}$ this has closed sets of strings $A = \{a\}$ and $B = \{\lambda, a\}$. $L = A \cdot B = B \cdot A$. So L is composite and it has two distinct prime factorisations, which are clearly both maximal.*

It simplifies the analysis here if we assume that all the concepts in a language have unique prime decompositions; accordingly we will restrict ourselves to that case for the moment, though we will remove this requirement in Section 7.

Definition 3. *A language has the unique factorisation property (UFP) if every closed set of strings with a nonempty distribution has a unique maximal prime factorisation; as before we stipulate that $\{\lambda\}^{\triangleright\triangleleft}$ has such a unique factorisation.*

If a language has the UFP, and P is a closed set of strings, then we can write $\Phi(P)$ for the unique prime factorisation, which is a string of primes, in the case of P, is the length 1 string. If $\alpha = \Phi(P)$ then $\bar{\alpha} = P$.

Now if we have languages which are UFP, DCC and a finite number of primes, then we can define a unique grammar.

Definition 4. *We define the class of languages $\mathfrak{L}_{\mathrm{P}}$ to be the set of languages which satisfy all of the following three conditions:*

1. *the unique prime decomposition property,*
2. *have no infinite ascending or descending chains,*
3. *and have a finite number of primes.*

All substitutable context free languages with a finite number of primes satisfy these conditions and so this is an extension of the approach in (Clark, 2014). All regular languages have a finite number of primes and therefore satisfy the chain conditions, but may not be uniquely decomposable.

Suppose we have a language $L \in \mathfrak{L}_{\mathrm{P}}$ and a prime N, we can construct a set of productions with N on the left hand side as follows.

Definition 5. *Let $\Gamma \subseteq \mathfrak{B}(L)^+$. We define a preorder on these strings by $\alpha \precsim \beta$ iff $\bar{\alpha} \subseteq \bar{\beta}$.*

The maximal elements of Γ *wrt this pre-order are the elements*

$$\max \Gamma = \{\alpha \in \Gamma \mid \forall \beta \in \Gamma, \bar{\beta} \supseteq \bar{\alpha} \Rightarrow \bar{\alpha} = \bar{\beta}\}.$$

Definition 6. *Define* $\Gamma(N) = \{\alpha \in \mathfrak{P}(L)^+_\lambda \mid N \supset \bar{\alpha}\}$. *We can take the maximal elements of this set:*
$\max(\Gamma(N)) = \{\alpha \in \Gamma(N) \mid \forall \alpha' \in \Gamma(N), \bar{\alpha'} \supseteq \bar{\alpha} \Rightarrow \alpha' = \alpha\}$

The elements of $\max(\Gamma(N))$ will be the right hand sides of productions with N on the left hand side.

Lemma 4. *If* $\alpha \in \Gamma(N)$ *then there is an* $\alpha' \in \max(\Gamma(N))$ *such that* $\bar{\alpha} \subseteq \bar{\alpha'}$.

Proof. Consider the set $\{\gamma \in \Gamma(N) \mid \bar{\gamma} \supseteq \bar{\alpha}\}$. If $\mathfrak{B}(L)$ has no infinite ascending chains, then every non empty set has a maximal element; so let α' be a maximal element of this set which clearly satisfies the condition required. □

Note that $\langle N \rangle$ is not in $\Gamma(N)$, since we require a strict superset relation; this saves the definition from vacuity. If we have infinite ascending chains then we may not have a maximal element. This motivates the use of an ascending chain condition.

5 Canonical Grammar based on primes

We are now in a position to define a set of canonical grammars for $\mathfrak{L}_{\text{P}}$. Given the nature of the problem it is inevitable that we will have to have a large number of restrictions on the sorts of grammars that we can learn. There are two ways of framing these restrictions either as restrictions on the grammars that generate the language or as language theoretic restrictions themselves. Here we stick to the latter approach; we take a language, and define criteria that define a class of languages. As the reader will see though, we can express the constraints we need in purely language theoretic terms, though those constraints will correspond naturally to finiteness constraints on the various sets of nonterminals and productions.

Definition 7. *For each language* $L \in \mathfrak{L}_{\text{P}}$ *we define the grammar* $\mathcal{G}_{\text{P}}(L)$ *as the tuple* $\langle \Sigma, V, S, P \rangle$ *where*

- S *is a distinguished symbol,*
- $V = \{S\} \cup \mathfrak{P}(L)_\lambda$. *If* $\top^\triangleright = \emptyset$ *then we remove* $\top$ *from* V*: if* $\bot = \emptyset$ *then we remove* $\bot$,
- P *is the union of the following sets of productions, for all* $N \in \mathfrak{P}(L)_\lambda$

  $N \to a$ *if* $a \in N$, *and* $a \in \Sigma$

  $N \to \lambda$ *if* $\lambda \in N$

  $N \to \alpha$ *for all* $\alpha \in \max(\Gamma(N))$

  $S \to \Phi(L)$.

Lemma 5. *For every prime* N, $\mathcal{L}(G, N) \supseteq N$.

Proof. Induction on length of w. Base case; $|w| \leq 1$. in which case if $w \in N$ then there is a rule $N \to w$. Otherwise if $w \in N$ and $w = a_1 \dots a_n$ where $a_i \in \Sigma$, then since $N \supseteq a_1^{\triangleright\triangleleft} \dots a_n^{\triangleright\triangleleft}$, and each $a_i^{\triangleright\triangleleft}$ is prime, we know that we have a correct production $\langle a_1^{\triangleright\triangleleft} \dots a_n^{\triangleright\triangleleft} \rangle \in \beta(N)$. Therefore there is some $\alpha \in \gamma(N)$ such that $w \in \bar{\alpha}$, and a production $N \to \alpha$. Since $w \in \bar{\alpha}$ we must have strings $w_1 \dots w_k = w$ such that $k = |\alpha|$, and for each $1 \leq i \leq k$ $w_i \in N_i$ and $\alpha = N_1 \dots N_k$.

By induction otherwise each w_i has length less than n and thus $N_i \stackrel{*}{\Rightarrow} w_i$, and therefore $N \stackrel{*}{\Rightarrow} w$. We should also consider the case where $k = 1$, in which case we have a unary rule, and the case where all but one of the w_i have length 0. In both cases we have one prime Q strictly less than N, and since we have a finite number of primes, and no unary cycles a simple induction on the derivation height will suffice. □

Lemma 6. *The canonical grammar for* G *is finite and generates* L.

Proof. Note that all rules are correct and thus by induction we can show that $\mathcal{L}(G, N) \subseteq N$, which combined with the previous lemma tells us that $\mathcal{L}(G, P) = P$ for all primes.

It is finite since we cannot have two productions whose right hand sides start with the same prime. Note that $\mathfrak{B}(L)$ is a residuated lattice and that for any two closed sets of strings X, Y, $X \backslash Y = \{w \mid X \cdot \{w\} \subseteq Y\}$ is closed. If $P \to N\alpha$ and $P \to N\beta$ are both productions with $|\alpha| > 0$, $|\beta| > 0$ then since $\alpha \leq N \backslash P$ we have that $\alpha = \beta$ since they are both equal to the unique prime factorisation of $N \backslash P$. Therefore if there are only n primes, we can

have at most $2n + |\Sigma| + 1$ productions with P on the left hand side.

Finally we verify that S generates the right strings, which is immediate. Either L is prime, in which case it is trivial since we have a production $S \to L$, or L has prime decomposition $N_1 \dots N_k$, in which case we can see that the production $S \to N_1 \dots N_k$ combined with the fact that $\mathcal{L}(N_i) = N_i$, gives the fact that $\mathcal{L}(S) = L$. □

This gives us a canonical grammar class for $\mathfrak{L}_{\text{P}}$ but the grammars are still very redundant and ambiguous. In the next section we consider how to select a smaller set of nonterminals.

Lemma 7. *Every language with a finite number of primes and no infinite chains is a context free language.*

Proof. Clearly $\mathfrak{L}_{\text{P}}$ is a proper subset of the context free languages, but even if we have no unique prime decomposition, we can still get a CFG by picking some shortest prime decomposition nondeterministically. □

We can weaken these conditions as we shall see as they are not necessary conditions; here is an example of a CFL with infinite chains that still receives an adequate canonical grammar.

Example 4. *Let $L = \{w \in \{a, b\}^+ : |w|_a > |w|_b\}$ The congruence classes are obviously indexed by $|w|_a - |w|_b$. Call these $E_n = \{w \mid |w|_a - |w|_b = n\}$. The closed sets of strings are $C_n = \{w \mid |w|_a - |w|_b \geq n\}$. $L = C_1$ is prime, C_{-1} is prime, C_0 is prime. C_n for $n > 1$ are composite. $C_n \circ C_m = C_{n+m}$.*

$C_2 = C_1 \circ C_1 = b^{\triangleright\triangleleft} \circ b^{\triangleright\triangleleft}$ and it is composite. Since $C_1 = E_1 \cup C_0$, $C_1 \cdot C_1 = (E_1 \cup C_0) \cdot (E_1 \cup C_0) = E_1 \cdot E_1 \cup (E_1 \cdot C_0) \cup (C_0 \cdot C_0) \cup (C_0 \cdot E_1)$. Since $\lambda \in C_0$ this means that $C_1 \cdot C_1 \supseteq C_0 \cup E_1 \cup E_2 = C_2$. Ergo this is composite. Therefore there are exactly three primes. However this still has a simple canonical grammar.

6 Lattice structure

For general context-free languages there may be very many prime concepts, and as a result grammars based on all primes may be excessively ambiguous, and unsuitable for the description of natural languages.[2] Indeed there are finite languages where the number of primes is exponential in the number of strings in the language.

Example 5. *For some large $\Sigma = \{a_1, \dots, a_n\}$ define $L = \{a_i a_j \mid i \neq j\}$. Clearly $|L| = n(n-1)$. Every nonempty proper subset X of Σ is closed; defined by the set of contexts $\{\Box a_i \mid a_i \notin X\}$. $\mathfrak{B}(L)$ therefore has $2^n + 1$ concepts, none of which are composite.*

In as case such as this the grammar defined by $\mathcal{G}_{\text{P}}(L)$ will be exponentially larger than $|L|$; which is clearly undesirable.

Moreover, the previous approach relies on the number of primes in the language being finite. Many simple languages have an infinite number of primes though, and so it is natural to try to extend this approach by considering some additional properties that might serve to pick out a finite subset of these primes. We need some additional constraints to get smaller and less ambiguous grammars. While it is natural to look to the meet and join irreducible elements of the syntactic concept lattice, it seems better to use a slightly larger set; the images of the irreducible elements of the free lattice. We call these semi-irreducible; we would like to use the terms join-prime and meet-prime, but these already have different meanings in lattice theory.

Definition 8. *Suppose $X \in \mathfrak{B}(L)$; we say that X is join-semi-irreducible (*JSI*) if $X = \{w\}^{\triangleright\triangleleft}$ for some $w \in \Sigma^*$. we say that X is meet-semi-irreducible (*MSI*) if $X = \{l \Box r\}^{\triangleleft}$ for some $l, r \in \Sigma^*$.*

Observe that if X is join-semi-irreducible then it contains some strings that are not in any lower concepts. Similarly if X is meet-semi-irreducible then it contains some contexts that do not occur in any higher concepts. Note that L is always MSI.

We will illustrate this with a simple example, using a finite language.

Example 6. *Consider the language generated by the* CFG, *$S \to AX, S \to BY, A \to a, A \to c, A \to m, B \to b, B \to c, X \to x, X \to z, X \to m, Y \to y, Y \to z$. This is a finite language which consists of 11 strings, all of length 2. The string cz receives two parses under this grammar:*

[2] We do not discuss here the difficult question of whether natural language grammars exhibit spurious ambiguity—syntactic ambiguity that does not relate to semantic ambiguity.

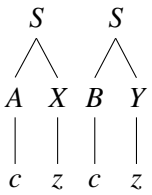

Figure 1 shows the lattice for this language. Note that the ambiguous letters/words $\{c, m, z\}$ are at the bottom of the diagram. For example $\{c\}^{\triangleright} = \mathcal{C}(A) \cup \mathcal{C}(B)$; this is clearly JSI*, since it is defined by c, but not* MSI*. At the top of the diagram are concepts that are* MSI*, but not* JSI*. In the middle, marked with boxes we have the concepts that are* MSI-JSI*.*

We can represent every element either as a meet of MSI-concepts or a join of JSI-concepts.

Lemma 8. *For any $X \in \mathfrak{B}(L)$, $X = \bigcup\{\{w\}^{\triangleright\triangleleft} \mid w \in X\}$ and $X^{\triangleright} = \bigcup\{\{l\Box r\}^{\triangleleft\triangleright} \mid l\Box r \in X^{\triangleright}\}$.*

We can now discuss the role of these concepts. Suppose we have some derivation in a grammar $S \stackrel{*}{\Rightarrow} lNr \stackrel{*}{\Rightarrow} lwr$, where N is the nonterminal corresponds to some concept.

This places two constraints on N. On the one hand $l\Box r \in N^{\triangleright}$, in other words $N \leq \{l\Box r\}^{\triangleleft}$, which is an MSI-concept. On the other hand $w \in N$ in other words $N \geq w^{\triangleright\triangleleft}$, a JSI-concept. Clearly $\{l\Box r\}^{\triangleleft} \geq w^{\triangleright\triangleleft}$ since $lwr \in L$. In the special case where $\{l\Box r\}^{\triangleleft} = w^{\triangleright\triangleleft}$, we know then that this must be the value of N. Therefore the elements that are MSI, JSI and primes are very special.

Definition 9. *A closed set of strings is an* MSI-JSI-*prime, if it is* MSI*,* JSI *and prime.*

7 Grammar based on MSI-JSI-primes

We now consider how to define a class of grammars where the nonterminals consist only of the set of MSI-JSI-primes. Rather than requiring that the lattice contains no infinite chains, and using the UFP, we define a weaker property, which more directly determines the finiteness of the relevant sets of productions. We define the non-standard term *finitely Noetherian*.

Definition 10. *We say that a set $\Gamma \subseteq \mathfrak{B}(L)^+$, with the pre-order $\precsim$, is finitely Noetherian if $|\max \Gamma|$ is finite and every element of Γ is less than some maximal element.*

A set can fail to be finitely Noetherian either because it has infinite ascending chains with no maximal elements, or because it has an infinite number of maximal elements.

Definition 11. *$\mathfrak{L}_{\text{MJ}}$ is the set of all languages L such that*

1. *L is nonempty and does not contain λ.*
2. *There are a finite number of* MSI-JSI-*primes; we let V denote the set of* MSI-JSI-*primes, except for $\{\lambda\}^{\triangleright\triangleleft}$, $\top$ if $\top_{\triangleright} = \emptyset$, and $\bot$ if $\bot = \emptyset$.*
3. *For each $a \in \Sigma$ and each $l\Box r \in a^{\triangleright}$ there is some $X \in V$ such that $a \in X$ and $l\Box r \in X^{\triangleright}$.*
4. *For every $X \in V$, $\{\alpha \in V^+ \mid \bar{\alpha} \subset X\}$ is finitely Noetherian.*
5. *$\{\alpha \in V^+ \mid \bar{\alpha} \subseteq L\}$ is finitely Noetherian.*

$\mathfrak{L}_{\text{MJ}}$ is incomparable with $\mathfrak{L}_{\text{P}}$, as the following two examples show.

Example 7. *$L = \{ax, bx, ay, cy, bz, cz\}$. This language in is $\mathfrak{L}_{\text{P}}$ but not $\mathfrak{L}_{\text{MJ}}$ as there are no* MSI-JSI-*primes that contain for example a. $\{a, b\}$ is* MSI *as it is defined by $\Box x$, but not* JSI*. $\{a\}$ on the other hand is* JSI *but not* MSI*.*

Example 8. *$L = \{a^n x b^n \mid x \geq 0\} \cup \{a^n x b^n \mid x \geq 0\}$. This is in $\mathfrak{L}_{\text{MJ}}$ but not $\mathfrak{L}_{\text{P}}$. Note that for all n, $\{b^n, c^n\}$ is closed and* MSI *as it is defined by the single context $a^n x\Box$, and is clearly prime, but not* JSI*. Therefore there are infinitely many primes. The* MSI-JSI-*primes are only $\{a^n x b^n \mid x \geq 0\}$, $\{a^n x c^n \mid x \geq 0\}$, $\{a\}$, $\{b\}$, $\{c\}$ and $\{x\}$.*

Definition 12. *If $L \in \mathfrak{L}_{\text{MJ}}$ and V is the set of* MSI-JSI-*primes, then for a set of strings X we write $\Gamma(X) = \{\alpha \in V^+ \mid \bar{\alpha} \subset X\}$, and $\Delta(X) = \{\alpha \in V^+ \mid \bar{\alpha} \subseteq X\}$.*

Definition 13. *For every language L in $\mathfrak{L}_{\text{MJ}}$, we define a grammar*

$$\mathcal{G}_{\text{MJ}}(L) = \langle V \cup \{S\}, S, P_L \cup P_B \cup P_S \rangle$$

where

- *V is the set of* MSI-JSI-*primes of L.*
- *S is a distinguished symbol.*

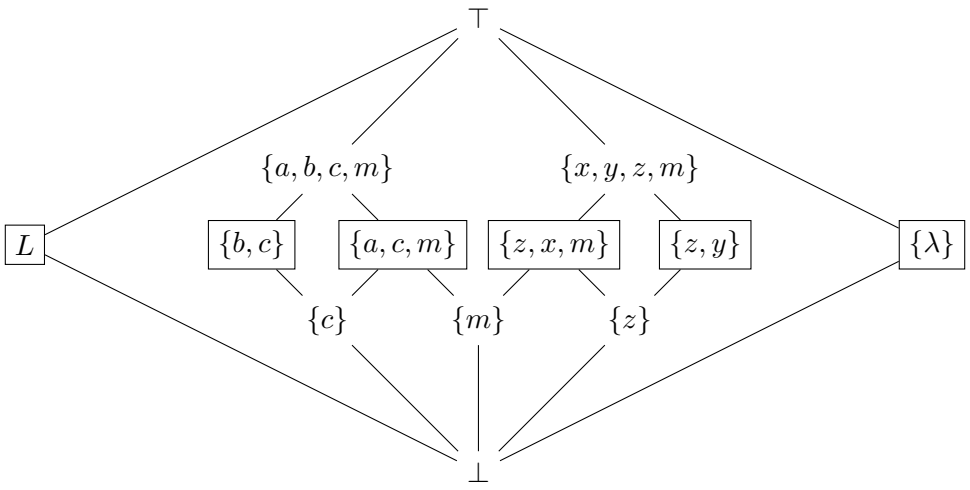

Figure 1: Hasse diagram of the syntactic concept lattice of language in Example 6. All of these elements are prime. The boxed elements are those which are both MSI and JSI, which, apart from the empty string concept, correspond to the nonterminals of the original grammar. $\{a, b, c, m\}$ is MSI, since it is equal to $\{\Box z\}^\triangleleft$, but it is not JSI.

- *P_L is the set of all productions $X \to a$ such that $X \in V$, $a \in \Sigma \cup \lambda$ and $a \in X$, and there is no $Y \in V$ such that $a \in Y$ and $Y < X$.*
- *P_B is the set of productions $X \to \alpha$ for every $X \in V$ and for every $\alpha \in \max \Gamma(X)$*
- *P_S is the set of productions $S \to \alpha$ for every $\alpha \in \max \Delta(L)$.*

Note that by Definition 11, this is finite and a CFG.

We now show that this grammar will generate all of the strings in the language. We do this by a joint induction on the length of the strings and the height of the nonterminals in the lattice; it seems easier to write these proofs as *reductio*s.

Lemma 9. *For any $X \in V$, If $\lambda \in X$ then*

$$X \stackrel{*}{\Rightarrow}_{G_*} \lambda.$$

Proof. Suppose this is false. Take a minimal $X \in V$ such that $\lambda \in X$ but it is not the case that $X \stackrel{*}{\Rightarrow}_{G_*} \lambda$. If X is a minimal element of the set of nonterminals that contain λ then there would be a production $X \to \lambda$ in G_*; therefore X is not minimal. Let Y be some nonterminal less than X such that $\lambda \in Y$. Since X is minimal we have $Y \stackrel{*}{\Rightarrow} \lambda$. Now $\langle Y \rangle \in \Gamma(X)$ so there is some production $X \to \alpha$ such that $\bar{\alpha} \supseteq Y$. Now if $\alpha = Z_1 \dots Z_k$ then each of the Z_i must be a proper subset of X that contains λ, (since $\lambda \in Y \subseteq \bar{\alpha}$). Since they are proper subsets we have $Z_i \stackrel{*}{\Rightarrow} \lambda$ and thus $X \stackrel{*}{\Rightarrow} \lambda$ which is a contradiction. □

Lemma 10. *For any $a \in \Sigma$ and $X \in V$, if $a \in X$ then*

$$X \stackrel{*}{\Rightarrow}_{G_*} a.$$

Proof. We use just the same argument as the previous proof, except that when we consider $\alpha = Z_1 \dots Z_k$, there must be at least one i such that $a \in Z_i$ and $\lambda \in Z_j$ for all $j \neq i$. By Lemma 9, $Z_j \stackrel{*}{\Rightarrow} \lambda$, by minimality of the counterexample $Z_i \stackrel{*}{\Rightarrow} a$, and therefore $X \stackrel{*}{\Rightarrow} a$. □

Lemma 11. *For any $w = a_1 \dots a_n \in \{l\Box r\}^\triangleleft$ for some $l\Box r$, there are $A_1, \dots, A_n \in V$ such that $a_i \in A_i$, and*

$$\overline{\langle A_1 \dots A_n \rangle} \subseteq \{l\Box r\}^\triangleleft.$$

Proof. By induction on n. Base case, $n = 1$ is trivial by Part 3 of Definition 11.

Clearly $a_1 \in \{l\Box a_2 \dots a_n r\}^\triangleleft$. Pick some $A_1 \in V$ such that $a_1 \in A_1$ and $l\Box a_2 \dots a_n r \in A_1^\triangleright$. Since $A_1 \in V$ there is some v_1 such that $A_1 = v_1^{\triangleright\triangleleft}$. Since $a_2 \dots a_n \in \{lv_1\Box r\}^\triangleleft$ then by the inductive hypothesis there are $A_2, \dots, A_n$ such that $\overline{\langle A_2 \dots A_n \rangle} \subseteq \{lv_1\Box r\}^\triangleleft$, and therefore

$$\overline{\langle A_1 \dots A_n \rangle} \subseteq \{l\Box r\}^\triangleleft.$$

The result then follows by induction. □

Lemma 12. *For any $X \in V$, $w \in X$ if $|w| > 1$ then*

$$X \stackrel{*}{\Rightarrow} w.$$

Proof. Suppose this is false for some w and X; pick a shortest w and a minimal X.

By Lemma 10, $|w| > 1$. Let $w = a_1 \dots a_n$. Let $l\Box r$ be some context such that $X = \{l\Box r\}^{\triangleleft}$. By Lemma 11, we know that we have some $A_1 \dots A_n$ such that $\overline{\langle A_1 \dots A_n \rangle} \subseteq X$, $a_i \in A_i$, for $1 \leq i \leq n$. Since X is a prime, we know that $\overline{\langle A_1 \dots A_n \rangle} \subset X$. Therefore there is some $X \to \beta$ such that $\overline{\langle A_1 \dots A_n \rangle} \subseteq \bar{\beta}$. Let $\beta = B_1 \dots B_k$ for some $k \geq 1$. Now $w \in \bar{\beta}$. Therefore there are strings $v_1 \dots v_k = w$ such that $v_j \in B_j$, $1 \leq j \leq k$. Now for each v_j, B_j either $|v_j| < |w|$ or $B_j \subset X$, and so by the assumption at the beginning of the proof, $B_j \stackrel{*}{\Rightarrow} v_j$. Therefore $X \Rightarrow B_1 \dots B_k \Rightarrow v_1 \dots v_k = w$ which is a contradiction. □

Theorem 1. *For every $L_* \in \mathfrak{L}_{\text{MJ}}$, $\mathcal{L}(\mathcal{G}_{\text{MJ}}(L_*)) = L_*$.*

Proof. Write G_* for $\mathcal{G}_{\text{MJ}}(L_*)$. It is easy to see that each production $X \to \alpha$ is correct. A simple induction establishes that $\mathcal{L}(G_*) \subseteq L_*$. The nontrivial part of the proof is to show that every string in L_* is generated from S. Given the previous lemmas, this is straightforward; the proofs are a bit repetitive, following the earlier lemmas with minor variations. Suppose $w \in L_*$.

- If $w = a$ for some $a \in \Sigma$ then pick some $X \in V$ such that $\Box \in X^{\triangleright}$ and $a \in X$. By Lemma 10, $X \stackrel{*}{\Rightarrow} a$ and $X \subseteq L$; If $X = L$ then there is a unary rule $S \to X$. Otherwise $\langle X \rangle \in \Delta(L)$ and so there is some production $S \to B_1 \dots B_k$ such that $a \in \overline{\langle B_1 \dots B_k \rangle}$. So there must be some B_i such that $a \in B_i$ and $\lambda \in B_j$ for $j \neq i$, and the result follows by Lemmas 10 and 9.

- If $|w| > 1$, then there are two cases. First it might be that $L \in V$, in which case there is a single rule $S \to L$, and it is immediate by Lemma 12. If not, then we can use the same argument as in Lemma 12, which we take rapidly here. If $w = a_1 \dots a_n \in L$, then we have some $A_1, \dots, A_n$ such that $\langle A_1 \dots A_n \rangle \in \Delta(L)$ and $A_i \stackrel{*}{\Rightarrow} a_i$ for $1 \leq i \leq n$. Therefore we have some rule $S \to B_1 \dots B_k$ where $w \in \overline{\langle B_1, \dots B_k \rangle}$ and therefore strings $v_1, \dots v_k$ such that $w = v_1 \dots v_k$ and $v_i \in B_i$; therefore $S \Rightarrow B_1 \dots B_k \stackrel{*}{\Rightarrow} v_1 \dots v_k = w$.

Label	Strings	Context
OAUX	can, λ	eagles □ eat.
AUX	can	□ eagles eat?
N	eagles	□ that eat fly.
NP	eagles that eat	□ can eat.
RC	that can eat, λ, ...	eagles □ can fly.
VI	eat, fly	eagles that eat can □.
VP	can eat, died	eagles that □ can fly.
C	that	eagles □ can fly fly.
STOP	.	eagles fly □
Q	?	can eagles fly □

Table 1: Prime concepts of the example; note that they are all MSI-JSI-primes except for OAUX.

Therefore $\mathcal{L}(G_*) \supseteq L_*$ which establishes the theorem. □

8 Example

We will illustrate some properties of our proposed solution with a simple toy example, based approximately on an example in (Berwick et al., 2011). We use a finite language as this shows most sharply the distinction between weak learning, which is trivial, and strong learning, which even in the case of acyclic CFGs is either impossible or intractable in the general case, depending on how it is formalised. The language is generated by the following grammar; optional elements are in brackets.

S → NP VP ., S → CAN NP VI ?
NP → EAGLES (RC), NP → THEY
RC → THAT VP
VP → (CAN) VI
VP → DIED
VI → EAT, VI → FLY

It contains examples like "can eagles that fly eat?" and "eagles that can fly can eat." "can eagles that died eat?". The language contains some optional elements and as a result is not substitutable. This is not the minimal example as for technical reasons the example needs to be sufficiently complex. Very simple examples may not contain enough informa-

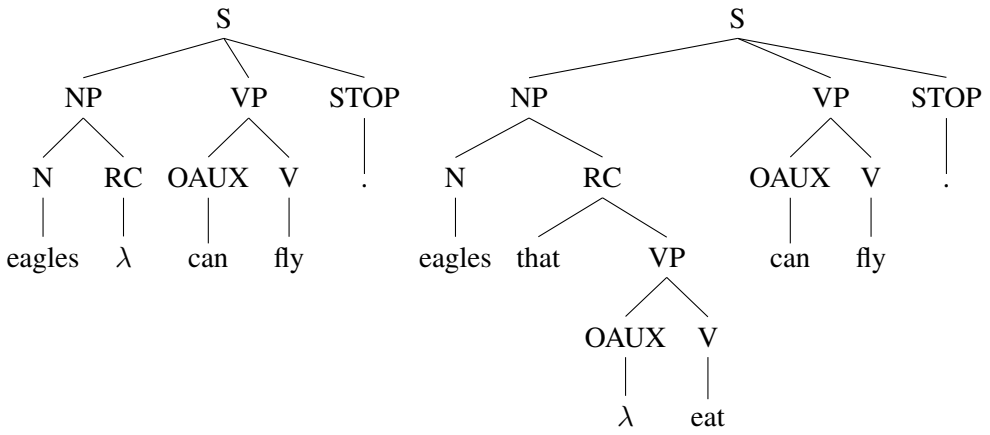

Figure 2: Example parse trees using $\mathcal{G}_{\text{P}}(\cdot)$. Note that where optional elements do not appear, we have an empty (phonologically null) constituent. This is a notational variant of having a unary rule.

tion to indicate the structure unequivocally. So for example, we added a word "died". This alters the structure of the lattice and means that the class VP is then a prime. Without this addition, the concept corresponding to VP would be decomposable as $\{\text{CAN}, \lambda\} \cdot \{\text{EAT}, \text{FLY}\}$. Similarly we added the word THEY to the class of NP. Without these additions, the language would not contain enough information to distinguish whether, for example, the relative clause attaches to the preceding noun or the following verb. Consider the language $L = \{ab, acb\}$. There is no motive, based on the strings, for the claim that c attaches to the left or the right. But if we enlarge it slightly to $\{ab, acb, ad\}$ it is more natural to attach the c optionally to the b.

The lattice contains 43 elements of which 10 are prime, which are listed in Table 1; we label each prime with a mnemonic label, reusing the nonterminal symbols from the target grammar for ease of reading. The composite elements contain for example the set of four strings $\{\text{EAT EAT}, \text{EAT FLY}, \text{FLY EAT}, \text{FLY FLY}\}$ which is clearly not prime.

Figure 2 and 3 show some trees of some of the sentences that illustrate the structure of the derived grammars.

9 Discussion and Conclusion

The two methods we present here do not exhaust the possibilities: rather they present two extremes. One method uses all primes, the other uses what

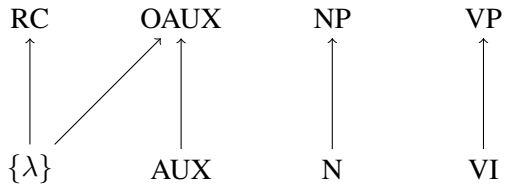

Figure 4: Lattice fragments showing some of the primes. The word CAN, forms a concept on its own, AUX, since when it occurs at the beginning of a sentence, it is obligatory. The concept RC which contains the sequences THAT EAT, THAT CAN FLY and so on, is always optional, and so there is no corresponding concept that does not contain λ.

seems to be the smallest possible set of primes that we can define using these techniques. Our reliance on individual contexts and substrings in the definition of MSI-JSI-primes is both natural but inadequate. In terms of earlier traditions of analytical linguistics, we are following Sestier-Kunze domination rather than Dobrushin-domination (Marcus, 1994). (Kunze, 1967 1968) argues that for the adequate description of German lexical categories simple Dobrushin domination is inadequate. Note also the similar notions in (Adriaans, 1999) of context-separability and expression-separability. The set of MSI-JSI-primes may well be too restricted; it seems necessary to have nonterminals that correspond to cases where $\{l \square r\}^{\triangleleft} \supsetneq \{w\}^{\triangleright\triangleleft}$. We leave this problem for future work.

It also seems quite natural to define a dual of primality. For a closed set of strings X, we can de-

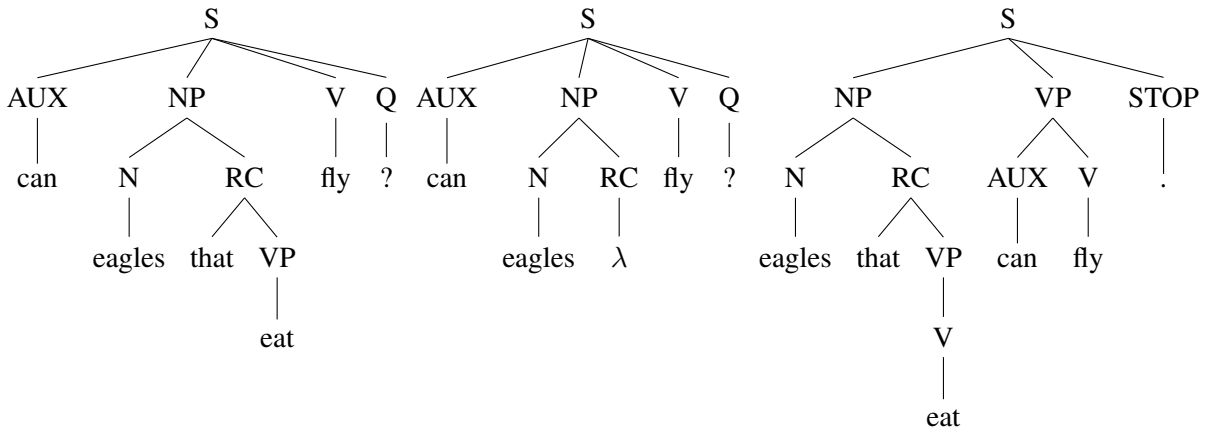

Figure 3: Example parse trees using the second approach, $\mathcal{G}_{\text{MJ}}(\cdot)$. Here we do not have OAUX, and so we have unary rules VP $\rightarrow$ V.

fine conditions $X^{\triangleright} = Y \square Z$, for some closed sets of strings Y, Z. There are also weaker variants of this. These correspond to a condition that the non-terminal occur on the left hand side of more than one production.

From a linguistic point of view, it is interesting that these approaches give local trees of potentially unbounded ranks as well as empty constituents and unary rules. This therefore means that the claim that natural languages only use a binary syntactic operation (MERGE) becomes a contentful empirical claim.

There is a close relation between the models used here and the primal and dual weak learning algorithms presented in (Yoshinaka, 2012b); in particular, the categories that are MSI, have the 1-finite-context-property (FCP) and the JSI-concepts have the 1-finite-kernel-property (FKP). The languages in $\mathfrak{L}_{\text{MJ}}$ therefore will be weakly learnable under certain paradigms. In order to turn the results here into a full strong learning result requires then the efficient computation of the canonical grammar given a sufficiently large weakly correct grammar. There do appear to be some technical problems to overcome: for example showing that the number of errors made in selecting the MSI-JSI-primes will only ever be finite.

Given the extension of distributional learning to multiple context-free grammars (MCFGs) (Seki et al., 1991) by (Yoshinaka, 2011), and the extension of the syntactic concept lattice in (Clark and Yoshinaka, 2014), it seems possible to straightforwardly extend these methods to at least some MCFGs. In particular the notion of a closed set of strings being composite is naturally generalised by replacing the single concatenation operation $\cdot$, with the family of all non-deleting non-permuting linear regular functions of appropriate arities.

The existence of these canonical grammars seems to be related in an interesting way to algebraic properties of the syntactic concept lattice. Indeed the finite cardinality of the lattice is exactly equivalent to the regularity of the language. It seems that other finiteness properties of the lattice—for example, compactness, the chain conditions etc.—may be crucial. More generally, the results presented here show that it may be possible to have strong learning algorithms for some quite large classes of languages. This suggests that the orthodox view that semantic information is required to learn syntactic structure may be mistaken; the set of strings of the language may define an intrinsic structure that can be learned purely distributionally. If the structures so defined can support compositional interpretation of the semantics, then this would provide strong empirical support for this approach.

Acknowledgments

I am grateful to Ryo Yoshinaka, Makoto Kanazawa and Greg Kobele, for technical comments; and to Bob Berwick, Paul Pietroski and George Walkden for discussions that have helped to motivate this approach.

References

Pieter Adriaans. 1999. Learning shallow context-free languages under simple distributions. Technical Report ILLC Report PP-1999-13, Institute for Logic, Language and Computation, Amsterdam.

R.C. Berwick, P. Pietroski, B. Yankama, and N. Chomsky. 2011. Poverty of the stimulus revisited. *Cognitive Science*, 35:1207–1242.

Alexander Clark and Ryo Yoshinaka. 2014. An algebraic approach to multiple context-free grammars. In Sergei Soloviev and Nicholas Asher, editors, *Logical Aspects of Computational Linguistics*. Springer.

Alexander Clark. 2011. A language theoretic approach to syntactic structure. In Makoto Kanazawa, András Kornai, Marcus Kracht, and Hiroyuki Seki, editors, *The Mathematics of Language*, pages 39–56. Springer Berlin Heidelberg.

Alexander Clark. 2013. The syntactic concept lattice: Another algebraic theory of the context-free languages? *Journal of Logic and Computation*.

Alexander Clark. 2014. Learning trees from strings: A strong learning algorithm for some context-free grammars. *Journal of Machine Learning Research*, 14:3537–3559.

B. A. Davey and H. A. Priestley. 2002. *Introduction to Lattices and Order*. Cambridge University Press.

J. Kunze. 1967–1968. Versuch eines objektivierten Grammatikmodells I, II. Z. *Zeitschriff Phonetik Sprachwiss. Kommunikat*, 20-21.

Hans Leiß. 2014. Learning context free grammars with the finite context property: A correction of A.Clark's algorithm. In Glyn Morrill, Reinhard Muskens, Rainer Osswald, and Frank Richter, editors, *Formal Grammar*, volume 8612 of *Lecture Notes in Computer Science*, pages 121–137. Springer Berlin Heidelberg.

S Marcus. 1994. The status of research in the field of analytical algebraic models of language. In Carlos Martín-Vide, editor, *Current Issues in Mathematical Linguistics*, pages 3–23. Elsevier.

H. Seki, T. Matsumura, M. Fujii, and T. Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):229.

Christian Wurm. 2012. Completeness of full Lambek calculus for syntactic concept lattices. In *Proceedings of the 17th conference on Formal Grammar 2012 (FG)*.

R. Yoshinaka. 2011. Efficient learning of multiple context-free languages with multidimensional substitutability from positive data. *Theoretical Computer Science*, 412(19):1821 – 1831.

R. Yoshinaka. 2012a. Integration of the dual approaches in the distributional learning of context-free grammars. In Adrian-Horia Dediu and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications*, volume 7183 of *Lecture Notes in Computer Science*, pages 538–550. Springer Berlin Heidelberg.

Ryo Yoshinaka. 2012b. Integration of the dual approaches in the distributional learning of context-free grammars. In Adrian-Horia Dediu and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications*, volume 7183 of *Lecture Notes in Computer Science*, pages 538–550. Springer Berlin Heidelberg.