Syntax Error Propagation in Context-Free Languages

Anonymous Author(s)

Abstract

Brzozowski (1964) defines a regular expression derivative as the suffixes which complete a known prefix. In this work, we establish a Galois connection with Valiant's (1975) fixpoint construction in the context-free setting, and further extend their work into the hierarchy of bounded context-sensitive languages realizable by finite CFL intersection. We illustrate how to lower context-free language recognition onto a tensor algebra over finite fields, loosely inspired by partial differentiation in Euclidean spaces. In addition to its theoretical value, this connection has yielded surprisingly useful applications in incremental parsing, code completion and program repair. For example, we use it to repair syntax errors, perform sketch-based program synthesis, and decide various language induction and membership queries.

1 Introduction

Recall that a CFG is a quadruple consisting of terminals (Σ), nonterminals (V), productions ($P\colon V \rightarrow (V \mid \Sigma)^*$), and a start symbol, ($S$). It is a well-known fact that every CFG is reducible to *Chomsky Normal Form*, $P'\colon V \rightarrow (V^2 \mid \Sigma)$, in which every production takes one of two forms, either $w \rightarrow xz$, or $w \rightarrow t$, where $w, x, z : V$ and $t : \Sigma$. For example, the CFG, $P := \{S \rightarrow SS \mid (S) \mid ()\}$, corresponds to the CNF:

$$P' = \left\{ S \rightarrow QR \mid SS \mid LR, \quad L \rightarrow (, \quad R \rightarrow), \quad Q \rightarrow LS \right\}$$

Given a CFG, $\mathcal{G}' : \langle \Sigma, V, P, S\rangle$ in CNF, we can construct a recognizer $R : \mathcal{G}' \rightarrow \Sigma^n \rightarrow \mathbb{B}$ for strings $\sigma : \Sigma^n$ as follows. Let 2^V be our domain, 0 be $\varnothing$, $\oplus$ be $\cup$, and $\otimes$ be defined as:

$$X \otimes Z := \left\{ w \mid \langle x, z\rangle \in X \times Z, (w \rightarrow xz) \in P \right\} \tag{1}$$

If we define $\sigma_r^{\uparrow\!\!\rightarrow} := \{w \mid (w \rightarrow \sigma_r) \in P\}$, then initialize $M^0_{r+1=c}(\mathcal{G}', e) := \sigma_r^{\uparrow\!\!\rightarrow}$ and solve for the fixpoint $M^* = M + M^2$,

$$M^0 := \begin{pmatrix} \varnothing & \sigma_1^{\rightarrow} & \varnothing & \cdots & \varnothing \\ \vdots & \ddots & & \ddots & \vdots \\ & & & \ddots & \varnothing \\ & & & & \sigma_n^{\uparrow} \\ \varnothing & \cdots & & \cdots & \varnothing \end{pmatrix} \Rightarrow M^* = \begin{pmatrix} \varnothing & \sigma_1^{\rightarrow} & \Lambda & \cdots & \Lambda_\sigma^* \\ \vdots & \ddots & & \ddots & \vdots \\ & & & \ddots & \Lambda \\ & & & & \sigma_n^{\uparrow} \\ \varnothing & \cdots & & \cdots & \varnothing \end{pmatrix}$$

we obtain the recognizer, $R(\mathcal{G}', \sigma) := S \in \Lambda_\sigma^*? \Leftrightarrow \sigma \in \mathcal{L}(\mathcal{G})?$ Full details of the bisimilarity between parsing and matrix multiplication can be found in Valiant [4] and Lee [3], who shows its time complexity to be $\mathcal{O}(n^\omega)$ where ω is the least matrix multiplication upper bound (currently, $\omega < 2.77$).

2 Method

Note that $\bigoplus_{c=1}^n M_{r,c} \otimes M_{c,r}$ has cardinality bounded by $|V|$ and is thus representable as a fixed-length vector using the characteristic function, $\mathbb{1}$. In particular, $\oplus, \otimes$ are redefined as $\boxplus, \boxtimes$ over bitvectors so the following diagram commutes,[1]

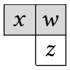
$$\begin{tikzcd}
\langle \mathcal{G}', \Sigma^{n-1} \rangle \arrow[dr, leftrightarrow] & \text{Set} & \text{Bit} & \langle \mathcal{G}', \Sigma^{n-1} \rangle \arrow[dr] & \text{SAT} \\
\text{Rubix} & M \times M \arrow[r, leftrightarrow, "\mathbb{1}^{2^{n\times n}}", "\mathbb{1}^{-2^{n\times n}}"'] \arrow[d, "\oplus \mid \otimes"] & \mathbb{Z}_2^{|V|^{n\times n}} \times \mathbb{Z}_2^{|V|^{n\times n}} \arrow[rr, tail, "\varphi^{2^{n\times n}}", "\varphi^{-2^{n\times n}}"'] \arrow[d, "+ \mid *"] & & \mathcal{M} \times \mathcal{M} \arrow[d, leftrightarrow, "+ \mid *"] \\
\text{Matrix} & 2^V \times 2^V \arrow[r, leftrightarrow, "\mathbb{1}^2", "\mathbb{1}^{-2}"'] \arrow[d, "\oplus \mid \otimes"] & \mathbb{Z}_2^{|V|} \times \mathbb{Z}_2^{|V|} \arrow[rr, tail, "\varphi^2", "\varphi^{-2}"'] \arrow[d, "\boxplus \mid \boxtimes"] & & \mathcal{V} \times \mathcal{V} \arrow[d, leftrightarrow, "\boxplus \mid \boxtimes"] \\
\text{Vector} & 2^V \arrow[r, leftrightarrow, "\mathbb{1}", "\mathbb{1}^{-1}"'] & \mathbb{Z}_2^{|V|} \arrow[rr, tail, "\varphi", "\varphi^{-1}"'] & & \mathcal{V}
\end{tikzcd}$$

where $\mathcal{V}$ is a function $\mathbb{Z}_2^{|V|} \rightarrow \mathbb{Z}_2$. Note that while always possible to encode $\mathbb{Z}_2^{|V|} \rightarrow \mathcal{V}$ using the identity function, φ^{-1} may not exist, as an arbitrary $\mathcal{V}$ might have zero, one, or in general, multiple solutions in $\mathbb{Z}_2^{|V|}$. Although holes may occur anywhere, let us consider two cases in which Σ^+ is strictly left- or right-constrained, i.e., $x\,z, x\,z : \Sigma^{|x|+|z|}$.

Valiant's $\otimes$ operator, which yields the set of productions unifying known factors in a binary CFG, naturally implies the existence of a left- and right-quotient, which yield the set of nonterminals that may appear the right or left side of a known factor and its corresponding root. In other words, a known factor not only implicates subsequent expressions that can be derived from it, but also adjacent factors that may be composed with it to form a given derivation.

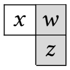
Left Quotient

$$\frac{\partial}{\partial \overleftarrow{x}} = \left\{ z \mid (w \rightarrow xz) \in P \right\}$$



Right Quotient

$$\frac{\partial}{\partial \overrightarrow{z}} = \left\{ x \mid (w \rightarrow xz) \in P \right\}$$



The left quotient coincides with the derivative operator first proposed by Brzozowski [2] and Antimirov [1] over regular languages, lifted into the context-free setting (our work). When the root and LHS are fixed, e.g., $\frac{\partial S}{\partial \overleftarrow{x}} : (\overleftarrow{V} \rightarrow S) \rightarrow \overrightarrow{\mathcal{V}}$ returns the set of admissible nonterminals to the RHS. One may also consider a gradient operator, $\overleftarrow{\nabla} S : (\overleftarrow{\mathcal{V}} \rightarrow S) \rightarrow \overrightarrow{\mathcal{V}}$, which simultaneously tracks the partials with respect to a set of multiple LHS nonterminals produced by a fixed root.

[1] Hereinafter, we use gray highlighting to distinguish between expressions containing only constants from those which may contain free variables.

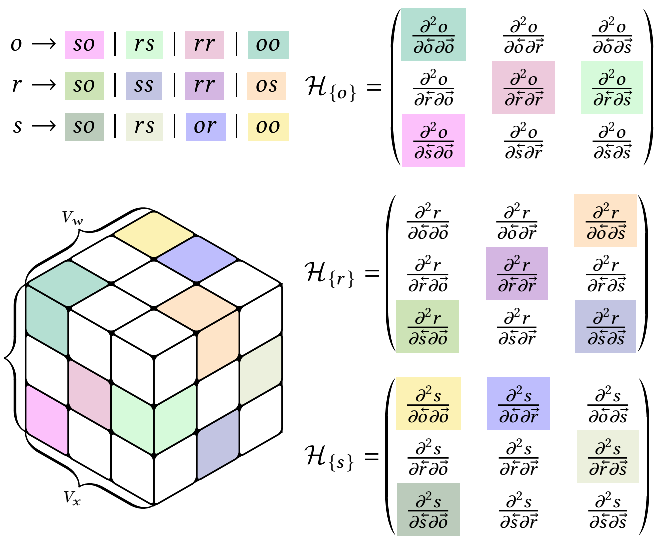

$o \to so \mid rs \mid rr \mid oo$
$r \to so \mid ss \mid rr \mid os$
$s \to so \mid rs \mid or \mid oo$
$\mathcal{H}_{\{o\}} = \begin{pmatrix} \frac{\partial^2 o}{\partial\overleftarrow{o}\partial\overrightarrow{o}} & \frac{\partial^2 o}{\partial\overleftarrow{o}\partial\overrightarrow{r}} & \frac{\partial^2 o}{\partial\overleftarrow{o}\partial\overrightarrow{s}} \\ \frac{\partial^2 o}{\partial\overleftarrow{r}\partial\overrightarrow{o}} & \frac{\partial^2 o}{\partial\overleftarrow{r}\partial\overrightarrow{r}} & \frac{\partial^2 o}{\partial\overleftarrow{r}\partial\overrightarrow{s}} \\ \frac{\partial^2 o}{\partial\overleftarrow{s}\partial\overrightarrow{o}} & \frac{\partial^2 o}{\partial\overleftarrow{s}\partial\overrightarrow{r}} & \frac{\partial^2 o}{\partial\overleftarrow{s}\partial\overrightarrow{s}} \end{pmatrix}$
$\mathcal{H}_{\{r\}} = \begin{pmatrix} \frac{\partial^2 r}{\partial\overleftarrow{o}\partial\overrightarrow{o}} & \frac{\partial^2 r}{\partial\overleftarrow{o}\partial\overrightarrow{r}} & \frac{\partial^2 r}{\partial\overleftarrow{o}\partial\overrightarrow{s}} \\ \frac{\partial^2 r}{\partial\overleftarrow{r}\partial\overrightarrow{o}} & \frac{\partial^2 r}{\partial\overleftarrow{r}\partial\overrightarrow{r}} & \frac{\partial^2 r}{\partial\overleftarrow{r}\partial\overrightarrow{s}} \\ \frac{\partial^2 r}{\partial\overleftarrow{s}\partial\overrightarrow{o}} & \frac{\partial^2 r}{\partial\overleftarrow{s}\partial\overrightarrow{r}} & \frac{\partial^2 r}{\partial\overleftarrow{s}\partial\overrightarrow{s}} \end{pmatrix}$
$\mathcal{H}_{\{s\}} = \begin{pmatrix} \frac{\partial^2 s}{\partial\overleftarrow{o}\partial\overrightarrow{o}} & \frac{\partial^2 s}{\partial\overleftarrow{o}\partial\overrightarrow{r}} & \frac{\partial^2 s}{\partial\overleftarrow{o}\partial\overrightarrow{s}} \\ \frac{\partial^2 s}{\partial\overleftarrow{r}\partial\overrightarrow{o}} & \frac{\partial^2 s}{\partial\overleftarrow{r}\partial\overrightarrow{r}} & \frac{\partial^2 s}{\partial\overleftarrow{r}\partial\overrightarrow{s}} \\ \frac{\partial^2 s}{\partial\overleftarrow{s}\partial\overrightarrow{o}} & \frac{\partial^2 s}{\partial\overleftarrow{s}\partial\overrightarrow{r}} & \frac{\partial^2 s}{\partial\overleftarrow{s}\partial\overrightarrow{s}} \end{pmatrix}$
V_w
V_z
V_x

Figure 1. CFGs are witnessed by a rank-3 tensor, whose nonempty inhabitants indicate CNF productions. Gradients in this setting effectively condition the parse tensor M by constraining the superposition of admissible parse forests.

2.1 Gradient estimation

Given some unparseable string, i.e., $\sigma_1 \ldots \sigma_n : \boxed{\Sigma}^n \cap \mathcal{L}(\mathcal{G})^c$, where should we put holes to obtain a parseable $\sigma' \in \mathcal{L}(\mathcal{G})$? To estimate the effect of perturbing σ on Λ^*_σ, one can either (1) backpropagate ∇S across upper-triangular entries of $\mathcal{M}^*$, or (2) stochastically sample *minibatches* $\boldsymbol{\sigma} : \Sigma^{n \pm q} \sim \Delta_q(\sigma)$ from the Levenshtein q-ball centered on σ, i.e., the space of all edits with Levenshtein distance $\leq q$, loosely analogous to a finite difference approximation. Let us consider (2) and suppose $U : \mathbb{Z}_2^{m \times m}$ is a matrix whose structure is shown in Eq. 2, wherein C is a primitive polynomial over $\mathbb{Z}_2^m$ with coefficients $C_{1 \ldots m}$ and semiring operators $\oplus := \veebar, \otimes := \land$:

$$U^t V = \begin{pmatrix} C_1 & \cdots & \cdots & \cdots & C_m \\ \top & \circ & \cdots & \cdots & \circ \\ \circ & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ \circ & \cdots & \circ & \top & \circ \end{pmatrix}^t \begin{pmatrix} V_1 \\ \vdots \\ \vdots \\ \vdots \\ V_m \end{pmatrix} \tag{2}$$

Since C is primitive, the sequence $\mathbf{S} = (U^{0 \ldots 2^m - 1} V)$ must be *fully periodic*, i.e., for all $i, j \in [0, 2^m)$, $\mathbf{S}_i = \mathbf{S}_j \Rightarrow i = j$. To uniformly sample $\boldsymbol{\sigma} \sim \Sigma^d$ without replacement, we form an injection $\mathbb{Z}_2^m \rightharpoonup \Sigma^d$, cycle over $\mathbf{S}$, then discard samples which index any nonexistent element(s) of Σ. This method will reject $(1 - |\Sigma| 2^{-\lceil \log_2 |\Sigma| \rceil})^d$ samples overall, and requires $\mathcal{O}(1)$ per sample and $\mathcal{O}(|\Sigma|^d)$ to cover Σ^d. Next, to admit deletion, we augment P with $(\varepsilon^+ \to \varepsilon \mid \varepsilon^+ \, \varepsilon^+)$ and replace each production $(w \to t)$ with $(w \to t \, \varepsilon^+ \mid \varepsilon^+ \, t \mid t)$. Finally, to generate $\boldsymbol{\sigma} \sim \Delta_q(\sigma)$, we enumerate hole configurations $H(\sigma, i) = \sigma_{1 \ldots i-1} \, \_ \, \_ \, \sigma_{i+1 \ldots n}$ for each $i \in \left\{ n \atop d \right\}$ and $d \in 1 \ldots q$, then solve for $\mathcal{M}^*$. If $S \in \Lambda^*_{\boldsymbol{\sigma}}$? has at least one solution, each edit in each $\sigma' \in \boldsymbol{\sigma}$ will match exactly one of seven patterns:

$$\text{Deletion} = \left\{ \sigma_1 \; \ldots \; \gamma_1 \, \gamma_2 \; \ldots \; \sigma_n \quad \gamma_{1,2} = \varepsilon \right.$$

$$\text{Substitution} = \begin{cases} \sigma_1 \; \ldots \; \gamma_1 \, \gamma_2 \; \ldots \; \sigma_n & \gamma_1 \neq \varepsilon \land \gamma_2 = \varepsilon \\ \sigma_1 \; \ldots \; \gamma_1 \, \gamma_2 \; \ldots \; \sigma_n & \gamma_1 = \varepsilon \land \gamma_2 \neq \varepsilon \\ \sigma_1 \; \ldots \; \gamma_1 \, \gamma_2 \; \ldots \; \sigma_n & \{\gamma_1, \gamma_2\} \cap \{\varepsilon, \sigma_i\} = \varnothing \end{cases}$$

$$\text{Insertion} = \begin{cases} \sigma_1 \; \ldots \; \gamma_1 \, \gamma_2 \; \ldots \; \sigma_n & \gamma_1 = \sigma_i \land \gamma_2 \notin \{\varepsilon, \sigma_i\} \\ \sigma_1 \; \ldots \; \gamma_1 \, \gamma_2 \; \ldots \; \sigma_n & \gamma_1 \notin \{\varepsilon, \sigma_i\} \land \gamma_2 = \sigma_i \\ \sigma_1 \; \ldots \; \gamma_1 \, \gamma_2 \; \ldots \; \sigma_n & \gamma_{1,2} = \sigma_i \end{cases}$$

This approach is tractable for $n \lesssim 100, q \lesssim 3$, however more complex repairs require a more efficient gradient estimator.

2.2 Context-sensitive reachability

It is well-known that the family of CFLs is not closed under intersection. For example, consider $\mathcal{L}_\cap := \mathcal{L}(\mathcal{G}_1) \cap \mathcal{L}(\mathcal{G}_1)$:

$$P_1 := \{ \; S \to LR, \quad L \to ab \mid aLb, \quad R \to c \mid cR \; \}$$
$$P_2 := \{ \; S \to LR, \quad R \to bc \mid bRc, \quad L \to a \mid aL \; \}$$

Note that $\mathcal{L}_\cap$ generates the language $\{ \; a^d b^d c^d \mid d > 0 \; \}$, which according to the pumping lemma is not context-free. We can encode $\bigcap_{i=1}^c \mathcal{L}(\mathcal{G}_i)$ as a polygonal prism with upper-triangular matrices adjoined to each rectangular face. More precisely, we intersect all terminals $\Sigma_\cap := \bigcap_{i=1}^c \Sigma_i$, then for each $t_\cap \in \Sigma_\cap$ and CFG, construct an equivalence class $E(t_\cap, \mathcal{G}_i) = \{ w_i \mid (w_i \to t_\cap) \in P_i \}$ and bind them together:

$$\bigwedge_{t \in \Sigma_\cap} \bigwedge_{j=1}^{c-1} \bigwedge_{i=1}^{|\sigma|} E(t_\cap, \mathcal{G}_j) \equiv_{\sigma_i} E(t_\cap, \mathcal{G}_{j+1}) \tag{3}$$

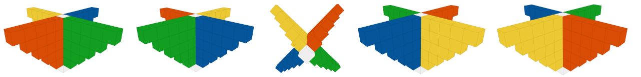


Figure 2. Orientations of a $\bigcap_{i=1}^4 \mathcal{L}(\mathcal{G}_i) \cap \Sigma^6$ configuration. As $c \to \infty$, this shape approximates a circular cone whose symmetric axis joins σ_i with orthonormal unit productions $w_i \to t_\cap$, and $S_i \in \Lambda^*_\sigma$? represented by the outermost bitvector inhabitants. Equations of this form are equiexpressive with the family of CSLs realizable by finite CFL intersection.

3 Conclusion

Not only is linear algebra over finite fields an expressive language for inference, but also an efficient framework for inference on languages themselves. We illustrate a few of its applications for parsing incomplete strings and repairing syntax errors in context- free and sensitive languages. In contrast with LL and LR-style parsers, our technique can recover partial forests from invalid strings by examining the structure of M^*. In future work, we hope to extend our method to more natural grammars like PCFG and LCFRS.

References

[1] Valentin Antimirov. 1996. Partial derivatives of regular expressions and finite automaton constructions. Theoretical Computer Science 155, 2 (1996), 291–319.

[2] Janusz A Brzozowski. 1964. Derivatives of regular expressions. Journal of the ACM (JACM) 11, 4 (1964), 481–494.

[3] Lillian Lee. 2002. Fast context-free grammar parsing requires fast boolean matrix multiplication. Journal of the ACM (JACM) 49, 1 (2002), 1–15. https://arxiv.org/pdf/cs/0112018.pdf

[4] Leslie G Valiant. 1975. General context-free recognition in less than cubic time. Journal of computer and system sciences 10, 2 (1975), 308–315. http://people.csail.mit.edu/virgi/6.s078/papers/valiant.pdf