Propagation of Syntax Errors in Context-Sensitive Languages using the Lifted Brzozowski Derivative

Breandan Mark Considine
McGill University
bre@ndan.co

Xujie Si
McGill University
xsi@cs.mcgill.ca

Abstract

Brzozowski defines the derivative of a regular language as the suffixes that complete a known prefix. In this work, we establish a Galois connection between Brzozowski's derivative and Valiant's 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 show how context-sensitive language recognition can be reduced into a tensor algebra over finite fields, drawing a loose analogy to partial differentiation in Euclidean spaces. In addition to its theoretical contributions, our method has yielded applications to incremental parsing, code completion and program repair. For example, we use it to repair syntax errors and perform sketch-based program synthesis, among other language decision problems.

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 $w \rightarrow xy$, or $w \rightarrow t$, where $w, x, y : V$ and $t : \Sigma$. For example, the CFG, $P := \{S \rightarrow SS \mid (S) \mid ()\}$, corresponds to the CNF:

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

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:

$$x \otimes y := \left\{ W \mid \langle X, Y \rangle \in x \times y, (W \rightarrow XY) \in P \right\} \qquad (1)$$

We initialize $\mathbf{M}^0_{r,c}(\mathcal{G}', e) := \{V \mid c = r + 1, (V \rightarrow \sigma_r) \in P\}$ and search for a matrix $\mathbf{M}^*$ via fixpoint iteration,

$$\mathbf{M}^* = \begin{pmatrix} \varnothing & \{V\}_{\sigma_1} & \cdots & \mathcal{T} \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \{V\}_{\sigma_n} \\ \varnothing & \cdots & \cdots & \varnothing \end{pmatrix} \qquad (2)$$

where $\mathbf{M}^*$ is the least solution to $\mathbf{M} = \mathbf{M} + \mathbf{M}^2$. We can then define the recognizer as: $S \in \mathcal{T}? \iff \sigma \in \mathcal{L}(\mathcal{G})?$

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, $\omega < 2.77$).

Note that $\bigoplus_{k=1}^{n} \mathbf{M}_{ik} \otimes \mathbf{M}_{kj}$ 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 defined as $\boxplus, \boxtimes$, so that the following diagram commutes:

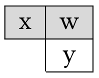
$$\begin{tikzcd} 2^V \times 2^V \arrow[r, "\oplus/\otimes"] \arrow[d, shift left, "\mathbb{1}^2"] & 2^V \arrow[d, shift left, "\mathbb{1}"] \\ \mathbb{Z}_2^{|V|} \times \mathbb{Z}_2^{|V|} \arrow[r, "\boxplus/\boxtimes"'] \arrow[u, shift left, "\mathbb{1}^{-2}"] & \mathbb{Z}_2^{|V|} \arrow[u, shift left, "\mathbb{1}^{-1}"] \end{tikzcd}$$

This construction can be lifted into the domain of bitvector variables, producing an algebraic expression for each scalar inhabitant of the northeasternmost bitvector $\mathcal{T}$, whose solutions correspond to valid parse forests for an incomplete string on the superdiagonal. Let us consider two cases, where $\mathbf{M}^*$ is either left- or right-constrained, i.e., $\alpha\gamma, \gamma\alpha$.[1]

Valiant's $\otimes$ operator, which solves for the set of productions unifying known factors in a binary CFG, implies the existence of a left- and right-quotient, which yield the set of nonterminals that may appear to the right- or left-side, respectively, of known factors in a ternary configuration.

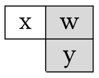
Left Quotient

$$\frac{\partial f}{\partial \overleftarrow{x}} = \left\{ y \mid (w \rightarrow xy) \in P \right\}$$



Right Quotient

$$\frac{\partial f}{\partial \vec{y}} = \left\{ x \mid (w \rightarrow xy) \in P \right\}$$



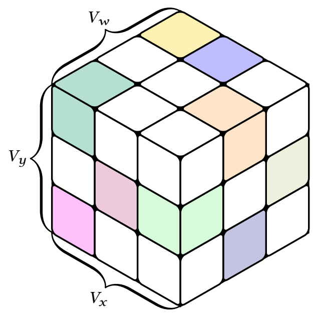
The left quotient coincides with the derivative operator first proposed by Brzozowski [2] and Antimirov [1] over regular languages, lifted into the context-free setting (our work).

Let $\mathcal{V}$ represent 2^V. Assuming the root is known, (e.g., S), the operator $\frac{\partial S}{\partial \overleftarrow{x}} : (\overleftarrow{V} \rightarrow S) \rightarrow \vec{\mathcal{V}}$ is a dependently-typed function which returns the unknown RHS factor. We may also define a gradient operator, $\overleftarrow{\nabla} S : (\overleftarrow{\mathcal{V}} \rightarrow S) \rightarrow \vec{\mathcal{V}}$, which tracks the partials with respect to multiple unknown factors.

If the root itself is unknown, we can define an operator, $\mathcal{H}_{\mathcal{W} \subseteq \mathcal{V}} : (\overleftarrow{\mathcal{V}} \times \vec{\mathcal{V}} \times \mathcal{W}) \rightarrow (\vec{\mathcal{V}} \times \vec{\mathcal{V}} \rightarrow \mathcal{W})$, which tracks second-order partial derivatives for all roots in $\mathcal{W}$. Unlike differential calculus on smooth manifolds, partials in this calculus do not necessarily commute depending on the CFG.

[1]Hereinafter, we shall use gray highlighting to distinguish between bound variables (i.e., constants) and free variables that are unhighlighted.

$$\begin{array}{l} o \rightarrow so \mid rs \mid rr \mid oo \\ r \rightarrow so \mid ss \mid rr \mid os \\ s \rightarrow so \mid rs \mid or \mid oo \end{array}$$



$$\mathcal{H}_{\{o\}} = \begin{pmatrix} \frac{\partial^2 o}{\partial\overleftarrow{o}\partial\overrightarrow{o}} & \frac{\partial^2 o}{\partial\overleftarrow{o}\partial\overrightarrow{r}} & \frac{\partial^2 o}{\partial\overleftarrow{o}\partial\overrightarrow{s}} \\ \frac{\partial^2 o}{\partial\overleftarrow{r}\partial\overrightarrow{o}} & \frac{\partial^2 o}{\partial\overleftarrow{r}\partial\overrightarrow{r}} & \frac{\partial^2 o}{\partial\overleftarrow{r}\partial\overrightarrow{s}} \\ \frac{\partial^2 o}{\partial\overleftarrow{s}\partial\overrightarrow{o}} & \frac{\partial^2 o}{\partial\overleftarrow{s}\partial\overrightarrow{r}} & \frac{\partial^2 o}{\partial\overleftarrow{s}\partial\overrightarrow{s}} \end{pmatrix}$$

$$\mathcal{H}_{\{r\}} = \begin{pmatrix} \frac{\partial^2 r}{\partial\overleftarrow{o}\partial\overrightarrow{o}} & \frac{\partial^2 r}{\partial\overleftarrow{o}\partial\overrightarrow{r}} & \frac{\partial^2 r}{\partial\overleftarrow{o}\partial\overrightarrow{s}} \\ \frac{\partial^2 r}{\partial\overleftarrow{r}\partial\overrightarrow{o}} & \frac{\partial^2 r}{\partial\overleftarrow{r}\partial\overrightarrow{r}} & \frac{\partial^2 r}{\partial\overleftarrow{r}\partial\overrightarrow{s}} \\ \frac{\partial^2 r}{\partial\overleftarrow{s}\partial\overrightarrow{o}} & \frac{\partial^2 r}{\partial\overleftarrow{s}\partial\overrightarrow{r}} & \frac{\partial^2 r}{\partial\overleftarrow{s}\partial\overrightarrow{s}} \end{pmatrix}$$

$$\mathcal{H}_{\{s\}} = \begin{pmatrix} \frac{\partial^2 s}{\partial\overleftarrow{o}\partial\overrightarrow{o}} & \frac{\partial^2 s}{\partial\overleftarrow{o}\partial\overrightarrow{r}} & \frac{\partial^2 s}{\partial\overleftarrow{o}\partial\overrightarrow{s}} \\ \frac{\partial^2 s}{\partial\overleftarrow{r}\partial\overrightarrow{o}} & \frac{\partial^2 s}{\partial\overleftarrow{r}\partial\overrightarrow{r}} & \frac{\partial^2 s}{\partial\overleftarrow{r}\partial\overrightarrow{s}} \\ \frac{\partial^2 s}{\partial\overleftarrow{s}\partial\overrightarrow{o}} & \frac{\partial^2 s}{\partial\overleftarrow{s}\partial\overrightarrow{r}} & \frac{\partial^2 s}{\partial\overleftarrow{s}\partial\overrightarrow{s}} \end{pmatrix}$$

Figure 1. CFGs are witnessed by a Rank-3 binary tensor, whose nonzero entries indicate CNF productions. Derivatives in this setting effectively condition the parse tensor. By backpropagating $\mathcal{H}$ across upper-triangular entries of $\mathbf{M}^*$, we constrain the superposition of admissible parse forests.

By allowing the matrix $\mathbf{M}^*$ to contain bitvector variables representing holes in σ, we obtain a set of multilinear equations whose solutions exactly correspond to the set of admissible repairs and their corresponding parse forests. Specifically, the repairs coincide with holes in the superdiagonal $\mathbf{M}^*_{r+1=c}$, and the parse forests occur along upper-triangular entries $\mathbf{M}^*_{r+1<c}$. In the case depicted below, $\mathbf{M}^*$ is left-constrained, although the holes may (in general) appear anywhere in σ:

$$\mathbf{M}^* = \begin{pmatrix} \varnothing & \{V\}_{\sigma_1} & \mathcal{L}_{1,3} & \mathcal{L}_{1,3} & \mathcal{V}_{1,4} & \cdots & \mathcal{V}_{1,n} \\ \vdots & \ddots & \{V\}_{\sigma_2} & \mathcal{L}_{2,3} & \vdots & & \vdots \\ & & \ddots & \{V\}_{\sigma_3} & \vdots & & \\ & & & \ddots & \mathcal{V}_{4,4} & \ddots & \\ \vdots & & & & & \ddots & \mathcal{V}_{n,n} \\ \varnothing & \cdots & & & & \cdots & \varnothing \end{pmatrix}$$

2 Context-Sensitive Reachability

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

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

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.

In our formalism, we encode finite intersections of CFLs $\bigcap_{i=0}^{c} \mathcal{L}(\mathcal{G}_i)$ as a prism with upper-triangular matrices joined to each rectangular face, like the fletches of an arrow. As $n \rightarrow \infty$, this shape approximates a right circular cone whose axis intersects equivalent nonterminals on the left hand side of unit productions, and base perimeter represents $S_i \in \mathcal{T}_i$. Equations of this form are equiexpressive with the family of CSLs realizable by intersecting a finite collection of CFLs.

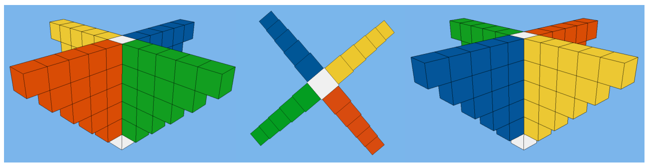


Figure 2. Orientations of a $\bigcap_{i=1}^{4} \mathcal{L}(\mathcal{G}_i) \cap \Sigma^6$ configuration.

2.1 Deletion, substitution, and insertion

Deletion, substitution and insertion can be simulated by adding a left- and right- ε production to each unit production:

$$\frac{\Gamma \vdash \varepsilon \in \Sigma}{\Gamma \vdash (\varepsilon^+ \rightarrow \varepsilon \mid \varepsilon^+\ \varepsilon^+) \in P}\ \varepsilon\text{-DUP}$$

$$\frac{\Gamma \vdash (A \rightarrow B) \in P}{\Gamma \vdash (A \rightarrow B\ \varepsilon^+ \mid \varepsilon^+\ B \mid B) \in P}\ \varepsilon^+\text{-INT}$$

To generate the sketch templates, we substitute two holes at an index-to-be-repaired, $H(\sigma, i) = \sigma_{1\ldots i-1}\ \_\ \_\ \sigma_{i+1\ldots n}$, then invoke the SAT solver. Five outcomes are then possible:

$$\sigma_1 \ldots \sigma_{i-1}\ \gamma_1\gamma_2\ \sigma_{i+1} \ldots \sigma_n, \gamma_{1,2} = \varepsilon \tag{3}$$
$$\sigma_1 \ldots \sigma_{i-1}\ \gamma_1\gamma_2\ \sigma_{i+1} \ldots \sigma_n, \gamma_1 \neq \sigma_i, \gamma_2 = \varepsilon \tag{4}$$
$$\sigma_1 \ldots \sigma_{i-1}\ \gamma_1\gamma_2\ \sigma_{i+1} \ldots \sigma_n, \gamma_1 = \varepsilon, \gamma_2 \neq \sigma_i \tag{5}$$
$$\sigma_1 \ldots \sigma_{i-1}\ \gamma_1\gamma_2\ \sigma_{i+1} \ldots \sigma_n, \gamma_1 = \sigma_i, \gamma_2 \neq \varepsilon \tag{6}$$
$$\sigma_1 \ldots \sigma_{i-1}\ \gamma_1\gamma_2\ \sigma_{i+1} \ldots \sigma_n, \gamma_1 \notin \{\varepsilon, \sigma_i\}, \gamma_2 = \sigma_i \tag{7}$$

Eq. (3) corresponds to deletion, Eqs. (4, 5) correspond to substitution, and Eqs. (6, 7) correspond to insertion. The solutions returned by the SAT solver will be strictly equivalent to handling each edit operation as separate cases.

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. http://maveric.uwaterloo.ca/reports/1964_JACM_Brzozowski.pdf

[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