Linear Conjunctive Reachability as Tensor Completion

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, i.e., conjunctive languages, illustrating how to lower conjunctive language recognition onto a system of multilinear equations over finite fields. In addition to its theoretical value, this connection has yielded a number of useful applications in incremental parsing, code completion and program repair.

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). All CFGs are reducible to *Chomsky Normal Form*, $P'\colon V \rightarrow (V^2 \mid \Sigma)$, where every production has either the form $w \rightarrow xz$, or $w \rightarrow t$, where $w, x, z : V$ and $t : \Sigma$. 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:

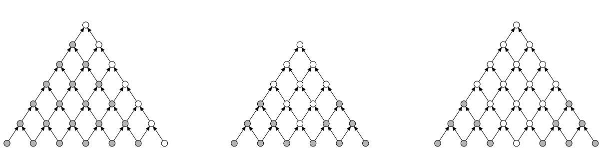
$$X \otimes Z := \left\{ \, w \mid \langle x, z \rangle \in X \times Z, (w \rightarrow xz) \in P \, \right\} \qquad (1)$$

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

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

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 [?] and Lee [?].

Incrementalizing the fixpoint solver allows us to handle Levenshtein edits with quadratic cost in $|\Sigma^*|$ assuming $\mathcal{O}(1)$ for each vector dot product. Visualized as a trellis automata:



Append: $\mathcal{O}(n+1)$ Delete: $\mathcal{O}\left(\frac{1}{4}(n-1)^2\right)$ Insert: $\mathcal{O}\left(\frac{1}{4}(n+1)^2\right)$

Incremental parsing is closely related to *dynamic matrix inverse* in the linear algebra setting, and *incremental transitive closure* with vertex updates in the graphical setting.

2 Galois Representation

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]

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

where $\mathcal{V}$ is a function $\mathbb{F}_2^{|V|} \rightarrow \mathbb{F}_2$. Note that while always possible to encode $\mathbb{F}_2^{|V|} \rightarrow \mathcal{V}$ using the identity function, φ^{-1} may not exist, as an arbitrary $\mathcal{V}$ might have zero, one, or in general, multiple solutions in $\mathbb{F}_2^{|V|}$. 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

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

x	w
	z

Right Quotient

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

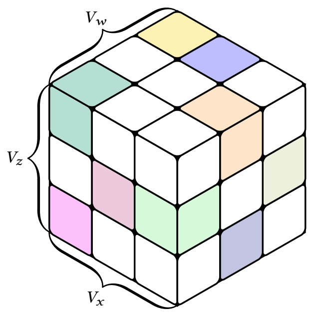
x	w
	z

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

$$o \to so \mid rs \mid rr \mid oo$$
$$r \to so \mid ss \mid rr \mid os$$
$$s \to so \mid rs \mid or \mid oo$$

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



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

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

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.

3 Context-sensitive reachability

It is well-known that the family of CFLs is not closed under intersection. For example, consider $\mathcal{L}_\cap := \mathcal{L}(\mathcal{G}_1) \cap \mathcal{L}(\mathcal{G}_2)$:

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

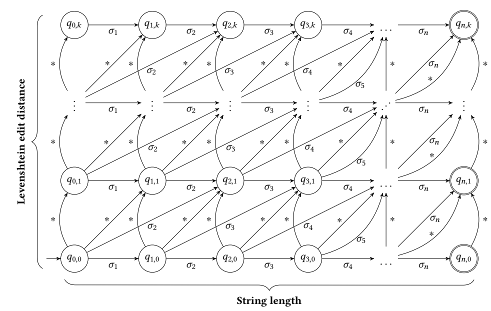
Note that $\mathcal{L}_\cap$ generates the language $\left\{\, a^d b^d c^d \mid d > 0 \,\right\}$, which according to the pumping lemma is not context-free. We can encode $\bigcap_{i=1}^{c} \mathcal{L}(\mathcal{G}_i)$ as a polygonal prism with upper-triangular matrices adjoined to each rectangular face. 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:

$$\bigwedge_{t \in \Sigma_\cap} \bigwedge_{j=1}^{c-1} \bigwedge_{i=1}^{|\sigma|} E(t_\cap, \mathcal{G}_j) \equiv_{\sigma_i} E(t_\cap, \mathcal{G}_{j+1}) \tag{2}$$

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.

4 Levenshtein Reachability

Levenshtein reachability is recognized by the nondeterministic infinite automaton (NIA) whose topology $\mathcal{L}$ = can be factored into a product of (a) the monotone Chebyshev topology, equipped with horizontal transitions accepting σ_i and vertical transitions accepting Kleene stars, and (b) the monotone knight's topology, equipped with transitions accepting σ_{i+2}. The structure of this space is approximated by an acyclic NFA [**?**], populated by accept states within radius k of $q_{n,0}$, or equivalently, a left-linear CFG whose productions finitely instantiate the transition dynamics:



Let $G(\underset{\sim}{\sigma} : \Sigma^*, d : \mathbb{N}^+) \mapsto \mathbb{G}$ be the construction described above accepting a string, $\underset{\sim}{\sigma}$, an edit distance, d, and returning a grammar that accepts the language of all strings within Levenshtein radius d of $\underset{\sim}{\sigma}$. To find the language edit distance and corresponding least-distance edit(s), we must find the least d such that $\mathcal{L}_d^\cap := \mathcal{L}(G(\underset{\sim}{\sigma}, d)) \cap \mathcal{L}(\mathcal{G}')$ is nonempty, i.e.: (1) $\tilde{\sigma} \in \mathcal{L}(\mathcal{G}')$, and (2) $\Delta(\underset{\sim}{\sigma}, \tilde{\sigma}) \leq d^* \iff \tilde{\sigma} \in \mathcal{L}(G(\underset{\sim}{\sigma}, d^*))$, and (3) $\nexists \sigma' \in \mathcal{L}(\mathcal{G}').[\Delta(\underset{\sim}{\sigma}, \sigma') < d^*]$. To satisfy these criteria, it suffices to check $d \in (1, d_{\max}]$ by encoding the Levenshtein automata and the original grammar as a single SAT formula, call it $\varphi_d(\cdot)$, and gradually admitting new acceptance states at increasing radii until either (1) a satisfying assignment is found or (2) $d_{\max}$ is attained. More precisely:

$$\varphi_{d+1} := \begin{cases} \varphi[\tilde{\sigma} \in \mathcal{L}(G(\underset{\sim}{\sigma}, d)) \land \tilde{\sigma} \in \mathcal{L}(\mathcal{G}')] & \text{if } d = 1 \text{ or SAT.} \\ \varphi_d \oplus \bigoplus_{\{q \in Q \mid \delta(q, q_{n,0}) = d+1\}} \varphi[S \to q] & \text{Otherwise.} \end{cases}$$

This procedure will terminate in either the number of steps required to overwrite every symbol in $\underset{\sim}{\sigma}$, or the length of the shortest string in $\mathcal{L}(\mathcal{G}')$, whichever is greater.

5 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 traditional 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.