Backpropagation of Syntax Errors in Context-Sensitive Languages

Breandan Considine, Jin Guo, Xujie Si

Main Idea

- Matrices over \mathbb{Z}_2^n are useful structures for studying finite state machines
- The operators $\{XOR, \land, \top\}$ are functionally complete logical primitives
- We use them to implement probabilistic context-sensitive program repair

Algebraic Parsing

Given a CFG, $\mathcal{G}':\langle \Sigma, V, P, S \rangle$ in Chomsky Normal Form (CNF), we can define a *recognizer*, $R:\mathcal{G}'\to \Sigma^n\to \mathbb{B}$ for bounded strings $\sigma:\Sigma^n$ using the following construction. Let 2^V be our domain, 0 be \varnothing , \oplus be \cup , and \otimes :

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

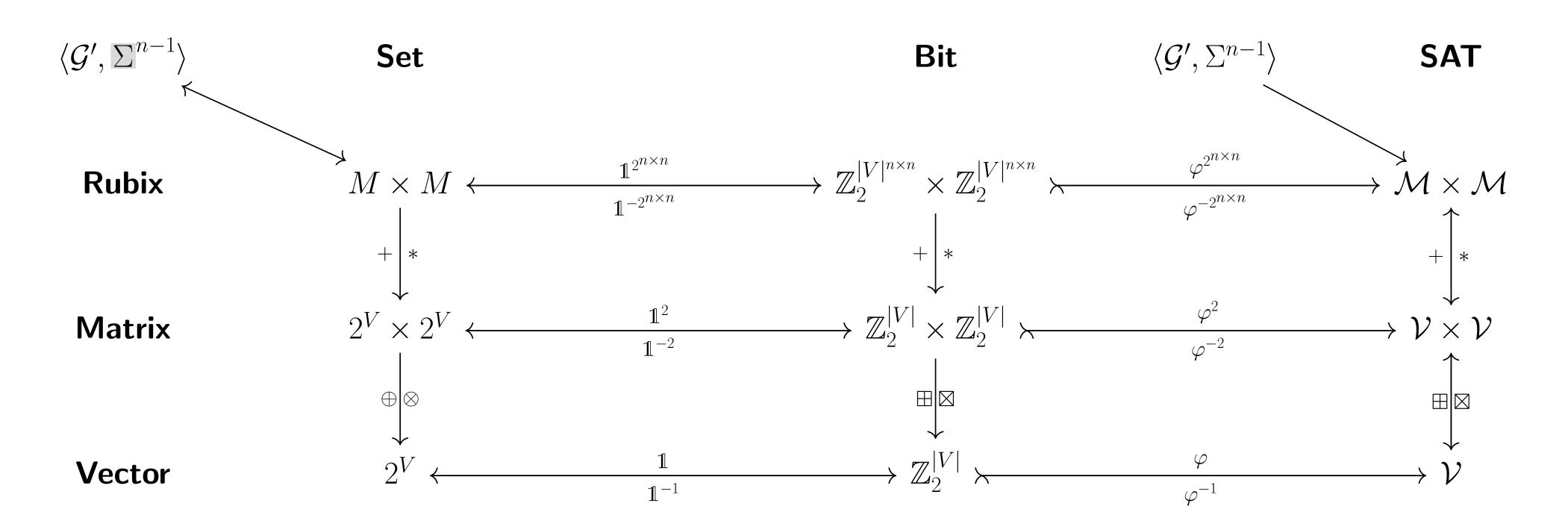
Valiant (1975) shows that if we let $\sigma_r^{\scriptscriptstyle \uparrow} \coloneqq \{V \mid (V \to \sigma_r^{\scriptscriptstyle \uparrow}) \in P\}$, initialize the matrix $M_{r+1=c}^0(\mathcal{G}',e) := \sigma_r^{\scriptscriptstyle \uparrow}$ and solve for its fixpoint $M^* = M + M^2$,

$$M^0 := egin{pmatrix} arphi & \sigma_1^{\scriptscriptstyle \uparrow} & arphi & arphi \ & arphi_n^{\scriptscriptstyle \uparrow} \ arphi & & arphi_n^{\scriptscriptstyle \uparrow} \ & & \sigma_n^{\scriptscriptstyle \uparrow} \ arphi & & arphi_n^{\scriptscriptstyle \uparrow} \ & & arphi_n^{\scriptscriptstyle \uparrow} \ arphi & & arphi_n^{\scriptscriptstyle \uparrow} \ \end{pmatrix} \Rightarrow M^* = egin{pmatrix} arphi & \sigma_n^{\scriptscriptstyle \uparrow} \ & & \sigma_n^{\scriptscriptstyle \uparrow} \ arphi & & arphi_n^{\scriptscriptstyle \uparrow} \ \end{matrix}$$

the recognizer is then defined as: $R(\mathcal{G}', \sigma) := S \in V_{\sigma}^*? \iff \sigma \in \mathcal{L}(\mathcal{G})?$

Galois Connection

- CYK parsers can be lowered onto $\mathbb{Z}_2^{|V| imes n imes n}$ or $\mathcal{M}: (\mathbb{Z}_2^{|V|} o \mathbb{Z}_2)^{|V| imes n imes n}$
- ullet \mathcal{M}^* can be solved for directly using Gaussian elimination or XOR-SAT
- Enables sketch-based synthesis in σ or \mathcal{G} : just use variables for holes!
- We can encode using the characteristic function, i.e., $\mathbb{1}_{\subseteq V}:V o\mathbb{Z}_2^{|V|}$
- \oplus , \otimes are defined as \boxplus , \boxtimes , so that the following diagram commutes:



Brozozowski's Derivative

Valiant's \otimes operator, which unifies known factors in a binary CFG, implies a left- and right-quotient, which yield the set of nonterminal forests that may appear to either side of a known factor and its corresponding root.

The left quotient coincides with Brzozowski's derivative (1964) over regular languages, here lifted into the context-sensitive setting (our work).

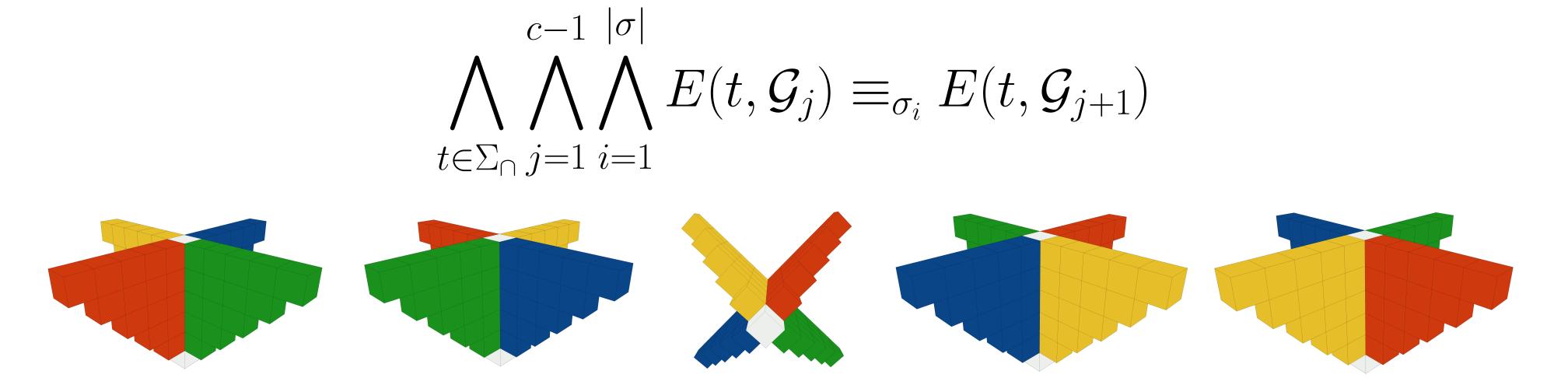
Context Sensitivity

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

$$P_1 := \left\{ S \to LR, L \to ab \mid aLb, R \to c \mid cR \right\}$$

$$P_2 := \left\{ S \to LR, R \to bc \mid bRc, L \to a \mid aL \right\}$$

 \mathcal{L}_{\cap} is equivalent to the language $\left\{ \begin{array}{l} a^db^dc^d \mid d>0 \end{array} \right\}$, which is not a CFL. 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. Specifically, we intersect all terminals $\Sigma_{\cap} := \bigcap_{i=1}^c \Sigma_i$, then for each $t \in \Sigma_{\cap}$, construct an equivalence class $E(t,\mathcal{G}_i) = \{w_i \mid (w_i \to t) \in P_i\}$ and glue them together at each σ_i :



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

As $c \to \infty$, this shape approximates a circular cone whose symmetric axis intersects orthonormal CNF unit productions $w_i \to t$, with $S_i \in V_\sigma^*$? encoded by bitvectors on the base perimeter. Equations of this form are equiexpressive with the family of CSLs realizable by finite CFL intersection.







