Deriving (finite) intersection non-emptiness, courtesy of Brzozowski

Breandan Considine

March 11, 2025

1 Syntax and semantics

Definition 1 (Generalized Regex)**.** Let E be an expression defined by the grammar:

$$E ::= \varnothing \mid \varepsilon \mid \Sigma \mid E \cdot E \mid E \vee E \mid E \wedge E$$

Semantically, we interpret these expressions as denoting regular languages:

$$\begin{aligned} \mathcal{L}(\quad \varnothing \quad) &= \varnothing & \qquad \mathcal{L}(\quad R \cdot S \quad) &= \mathcal{L}(R) \times \mathcal{L}(S) \\ \mathcal{L}(\quad \varepsilon \quad) &= \{\varepsilon\} & \qquad \mathcal{L}(\quad R \vee S \quad) &= \mathcal{L}(R) \cup \mathcal{L}(S) \\ \mathcal{L}(\quad a \quad) &= \{a\} & \qquad \mathcal{L}(\quad R \wedge S \quad) &= \mathcal{L}(R) \cap \mathcal{L}(S) \end{aligned}$$

Definition 2 (Brzozowski, 1964)**.** To compute the quotient $\partial_a(L) = \{b \mid ab \in L\}$, we:

$$\begin{aligned} \partial_a(\quad \varnothing \quad) &= \varnothing & \qquad \delta(\quad \varnothing \quad) &= \varnothing \\ \partial_a(\quad \varepsilon \quad) &= \varnothing & \qquad \delta(\quad \varepsilon \quad) &= \varepsilon \\ \partial_a(\quad b \quad) &= \begin{cases} \varepsilon & \text{if } a = b \\ \varnothing & \text{if } a \neq b \end{cases} & \qquad \delta(\quad a \quad) &= \varnothing \\ \partial_a(\quad R \cdot S \quad) &= (\partial_a R) \cdot S \vee \delta(R) \cdot \partial_a S & \qquad \delta(\quad R \cdot S \quad) &= \delta(R) \wedge \delta(S) \\ \partial_a(\quad R \vee S \quad) &= \partial_a R \vee \partial_a S & \qquad \delta(\quad R \vee S \quad) &= \delta(R) \vee \delta(S) \\ \partial_a(\quad R \wedge S \quad) &= \partial_a R \wedge \partial_a S & \qquad \delta(\quad R \wedge S \quad) &= \delta(R) \wedge \delta(S) \end{aligned}$$

Theorem 1 (Recognition)**.** *For any regex E and $\sigma : \Sigma^*$, $\sigma \in \mathcal{L}(E) \iff \varepsilon \in \mathcal{L}(\partial_\sigma E)$, where:*

$$\partial_\sigma(E) : RE \to RE = \begin{cases} E & \text{if } \sigma = \varepsilon \\ \partial_b(\partial_a E) & \text{if } E = a \cdot b, a \in \Sigma \end{cases}$$

Theorem 2 (Generation)**.** *For any $(\varepsilon, \wedge)$-free regex, R, to generate a witness $\sigma \sim \mathcal{L}(R)$:*

$$\texttt{follow}(R) : RE \to 2^\Sigma = \begin{cases} \{R\} & \text{if } R \in \Sigma \\ \texttt{follow}(S) & \text{if } R = S \cdot T \\ \texttt{follow}(S) \cup \texttt{follow}(T) & \text{if } R = S \vee T \end{cases}$$

$$\texttt{choose}(R) : RE \to \Sigma^+ = \begin{cases} R & \text{if } R \in \Sigma \\ \big(s \sim \texttt{follow}(R)\big) \cdot \texttt{choose}(\partial_s R) & \text{if } R = S \cdot T \\ \texttt{choose}(R' \sim \{S, T\}) & \text{if } R = S \vee T \end{cases}$$

2 Language intersection

Theorem 3 (Bar-Hillel, 1961)**.** *For any context-free grammar (CFG), $G = \langle V, \Sigma, P, S\rangle$, and nondeterministic finite automata, $A = \langle Q, \Sigma, \delta, I, F\rangle$, there exists a CFG $G_\cap = \langle V_\cap, \Sigma_\cap, P_\cap, S_\cap\rangle$ such that $\mathcal{L}(G_\cap) = \mathcal{L}(G) \cap \mathcal{L}(A)$.*

Definition 3 (Salomaa, 1973)**.** One could construct $G_\cap$ like so,

$$\frac{q \in I \quad r \in F}{\big(S \to qSr\big) \in P_\cap}\surd \qquad \frac{(A \to a) \in P \qquad (q \xrightarrow{a} r) \in \delta}{\big(qAr \to a\big) \in P_\cap}\uparrow \qquad \frac{(w \to xz) \in P \qquad p, q, r \in Q}{\big(pwr \to (pxq)(qzr)\big) \in P_\cap}\bowtie$$

however most synthetic productions in $P_\cap$ will be non-generating or unreachable.

Theorem 4 (Considine, 2025)**.** *For every CFG, G, and every acyclic NFA (ANFA), A, there exists a decision procedure φ : CFG $\to$ ANFA $\to \mathbb{B}$ such that $\varphi(G, A) \models [\mathcal{L}(G) \cap \mathcal{L}(A) \neq \varnothing]$ which requires $\mathcal{O}\big((\log|Q|)^c\big)$ time using $\mathcal{O}\big((|V||Q|)^k\big)$ parallel processors for some $c, k < \infty$.*

Proof sketch. WTS there exists a path $p \rightsquigarrow r$ in A such that $p \in I, r \in F$ where $S \vdash p \rightsquigarrow r$.

There are two cases, at least one of which must hold for $w \in V$ to parse a given $p \rightsquigarrow r$ pair:

1. p steps directly to r in which case it suffices to check $\exists s.\big((p \xrightarrow{s} r) \in \delta \wedge (w \to s) \in P\big)$, or,
2. there is some midpoint $q \in Q$, $p \rightsquigarrow q \rightsquigarrow r$ such that $\exists x, z.\big((w \to xz) \in P \wedge \overbrace{\underbrace{p \rightsquigarrow q}_{x}, \underbrace{q \rightsquigarrow r}_{z}}^{w}\big)$.

This decomposition suggests a dynamic programming solution. Let M be a matrix of type $RE^{|Q|\times|Q|\times|V|}$ indexed by Q. Since we assumed δ is acyclic, there exists a topological sort of δ imposing a total order on Q such that M is strictly upper triangular (SUT). Initiate it thusly:

$$M_0[r, c, v] = \bigvee_{a \,\in\, \Sigma} \{a \mid (v \to a) \in P \wedge (q_r \xrightarrow{a} q_c) \in \delta\} \tag{1}$$

The algebraic operations $\oplus, \otimes : RE^{2|V|} \to RE^{|V|}$ will be defined elementwise:

$$[\ell \oplus r]_w = [\ell_w \vee r_w] \tag{2}$$

$$[\ell \otimes r]_w = \bigvee_{x,z \,\in\, V} \{\ell_x \cdot r_z \mid (w \to xz) \in P\} \tag{3}$$

By slight abuse of notation, we will redefine the matrix exponential over this domain as:

$$\exp(M) = \sum_{i=0}^{\infty} M_0^i = \sum_{i=0}^{|Q|} M_0^i \text{ (since } M \text{ is SUT.)} \tag{4}$$

To solve for the fixpoint, we can instead use exponentiation by squaring:

$$S(2n) \;=\; \begin{cases} M_0, & \text{if } n = 1, \\ S(n) + S(n)^2 & \text{otherwise.} \end{cases} \tag{5}$$

Therefor, we only need a maximum of $\lceil \log_2 |Q| \rceil$ sequential steps to reach the fixpoint. Finally,

$$S_\cap = \bigvee_{q \in I,\, q' \in F} \exp(M)[q, q', S] \text{ and } \varphi = [S_\cap \neq \varnothing] \tag{6}$$

To decode a witness in case of non-emptiness, we simply `choose` (φ). □

3 Combinatorics

To enumerate, we first need $|\mathcal{L}(R)|$, which is denoted $|R|$ for brevity.

Definition 4 (Cardinality)**.** $|R| : RE \to \mathbb{N} = \begin{cases} 1 & \text{if } R \in \Sigma \\ S \times T & \text{if } R = S \cdot T \\ S + T & \text{if } R = S \vee T \end{cases}$

Theorem 5 (Enumeration)**.** *To enumerate, invoke* $\bigcup_{i=0}^{|R|} \textit{enum}(R, i)$*:*

$$\textit{enum}\,(R, n) : RE \times \mathbb{N} \to \Sigma^* = \begin{cases} R & \textit{if } R \in \Sigma \\ \textit{enum}\big(S, \lfloor \frac{n}{|T|} \rfloor\big) \cdot \textit{enum}\big(T,\, n \bmod |T|\big) & \textit{if } R = S \cdot T \\ \big(n < |S|\big)\ ?\ \textit{enum}(S, n)\ :\ \textit{enum}(T, n - |S|) & \textit{if } R = S \vee T \end{cases}$$

4 Fermi estimation

Suppose a PRAM-based model with 1 TFLOP throughput $\approx 2^{40}$ FLOPS and assume a 2^8 FLOP overhead per entry, n^3 entries per matrix multiplication, and approximately $\log_2(n)$ matrix multiplications. Thus, we would expect somewhere on the order of $2^{10} n^3 \log_2(n)$ total FLOPS per instance with a ballpark latency:

$$T \approx \frac{2^8\, n^3\, \log_2(n)}{2^{40}} = \frac{n^3\, \log_2(n)}{2^{32}}\ \text{s}$$

For example assuming, $n^3 = \left(2^9\right)^3 = 2^{27}, \log_2(n) = 9$, hence we can estimate:

$$T \approx \frac{2^{27} \cdot 9}{2^{32}} = \frac{9}{2^5} = \frac{9}{64} \approx 280\ \text{ms}$$

Further optimizations are likely realizable via bitpacking and vectorization.

5 Future work

Broadly interested in questions related to formal languages and finite model theory, following the Carnapian program of logical syntax. Encoding semantics into syntax would allow us to do type checking and static analysis in the parser. A few lines of attack here:

Result 1 (Büchi–Elgot–Trakhtenbrot)**.** A language is regular iff it is MSO-definable. For every MSO formula, there is a corresponding FSA. Complexity may be nonelementary.

Result 2 (Pentus, 1993)**.** Lambek categorial grammars, a weak kind of substructural logic, recognize exactly the context-free languages.

Result 3 (DeYoung & Pfenning, 2016)**.** Describes a certain equivalence between subsingleton logic, a weak kind of linear logic, and automata.

Result 4 (Knuth & Wegner, 1968)**.** Attribute grammars permit some notation of semanticity.

6 Acknowledgements

The author wishes to thank Fr. Paul Pomkowski for his phonetic advice.