

Symbolic and Automatic Differentiation of Languages

Conal Elliott

ICFP, August 2021

Languages via type theory

A language is a predicate on “strings” yielding proofs/explanations/parsings.

$$\text{Lang} = A^* \rightarrow \text{Set } \ell$$

$$\emptyset w = \perp$$

$$(P \cup Q) w = P w \uplus Q w$$

$$\mathcal{U} w = \top$$

$$(P \cap Q) w = P w \times Q w$$

$$\mathbf{1} w = w \equiv []$$

$$(P * Q) w = \exists \lambda (u, v) \rightarrow (w \equiv u ++ v) \times P u \times Q v$$

$$' c w = w \equiv [c]$$

$$(P \star) w = \exists \lambda ws \rightarrow (w \equiv \text{concat } ws) \times \text{All } P ws$$

$$(s \cdot P) w = s \times P w$$

Puzzle: how to compute parsings?

- Normal form for arbitrary (type-level) languages
- Normal form lemmas for standard language building blocks
- Decidability
- As corollaries, dual verified parsing implementations

Decomposing languages

Consider each list constructor:

$\nu : \text{Lang} \rightarrow \text{Set } \ell$ -- “nullable”

$\nu P = P []$

$\delta : \text{Lang} \rightarrow A \rightarrow \text{Lang}$ -- “derivative”

$\delta P a w = P (a :: w)$

ν and repeated δ capture languages fully:

$\nu \circ \text{foldl} \delta : \nu \circ \text{foldl} \delta P \cong P$

$\nu \circ \text{foldl} \delta [] = \text{refl}$

$\nu \circ \text{foldl} \delta (a :: as) = \nu \circ \text{foldl} \delta as$

$\text{foldl} : (X \rightarrow A \rightarrow X) \rightarrow X \rightarrow A^* \rightarrow X$

$\text{foldl } h x [] = x$

$\text{foldl } h x (a :: as) = \text{foldl } h (h x a) as$

Language calculus lemmas

$$\nu \emptyset : \nu \emptyset \equiv \perp$$

$$\nu \mathcal{U} : \nu \mathcal{U} \equiv \top$$

$$\nu \cup : \nu (P \cup Q) \equiv (\nu P \uplus \nu Q)$$

$$\nu \cap : \nu (P \cap Q) \equiv (\nu P \times \nu Q)$$

$$\nu \cdot : \nu (s \cdot P) \equiv (s \times \nu P)$$

$$\nu \mathbf{1} : \nu \mathbf{1} \leftrightarrow \top$$

$$\nu * : \nu (P * Q) \leftrightarrow (\nu P \times \nu Q)$$

$$\nu \star : \nu (P \star) \leftrightarrow (\nu P) \star$$

$$\nu ' : \nu (' c) \leftrightarrow \perp$$

$$\delta \emptyset : \delta \emptyset a \equiv \emptyset$$

$$\delta \mathcal{U} : \delta \mathcal{U} a \equiv \mathcal{U}$$

$$\delta \cup : \delta (P \cup Q) a \equiv \delta P a \cup \delta Q a$$

$$\delta \cap : \delta (P \cap Q) a \equiv \delta P a \cap \delta Q a$$

$$\delta \cdot : \delta (s \cdot P) a \equiv s \cdot \delta P a$$

$$\delta \mathbf{1} : \delta \mathbf{1} a \leftrightarrow \emptyset$$

$$\delta * : \delta (P * Q) a \leftrightarrow \nu P \cdot \delta Q a \cup \delta P a * Q$$

$$\delta \star : \delta (P \star) a \leftrightarrow (\nu P) \star \cdot (\delta P a * P \star)$$

$$\delta ' : \delta (' c) a \leftrightarrow (a \equiv c) \cdot \mathbf{1}$$

Decidable types

```
data Dec (X : Set ℓ) : Set ℓ where
  yes : X → Dec X
  no  : ¬ X → Dec X      (¬ X = X → ⊥)
```

For predicates (languages),

Decidable $P = \forall x \rightarrow \text{Dec } (P\ x)$

Isomorphisms:

$_ \triangleleft _ : B \leftrightarrow A \rightarrow \text{Dec } A \rightarrow \text{Dec } B$

$_ \blacktriangleleft _ : Q \leftrightarrow P \rightarrow \text{Decidable } P \rightarrow \text{Decidable } Q$

Compositionally decidable types

$\perp^? : \text{Dec } \perp$

$\perp^? = \text{no } (\lambda ())$

$\top^? : \text{Dec } \top$

$\top^? = \text{yes tt}$

$_ \uplus^? _ : \text{Dec } A \rightarrow \text{Dec } B \rightarrow \text{Dec } (A \uplus B)$

$\text{no } \neg a \uplus^? \text{no } \neg b = \text{no } [\neg a, \neg b]$

$\text{yes } a \uplus^? \text{no } \neg b = \text{yes } (\text{inj}_1 a)$

$_ \uplus^? \text{yes } b = \text{yes } (\text{inj}_2 b)$

$_ \times^? _ : \text{Dec } A \rightarrow \text{Dec } B \rightarrow \text{Dec } (A \times B)$

$\text{yes } a \times^? \text{yes } b = \text{yes } (a, b)$

$\text{no } \neg a \times^? \text{yes } b = \text{no } (\neg a \circ \text{proj}_1)$

$_ \times^? \text{no } \neg b = \text{no } (\neg b \circ \text{proj}_2)$

$_ \star^? : \text{Dec } A \rightarrow \text{Dec } (A \star)$

$_ \star^? = \text{yes } []$

Reflections

$$\nu : \text{Lang} \rightarrow \text{Set } \ell$$

$$\delta : \text{Lang} \rightarrow A \rightarrow \text{Lang}$$

$$\nu \circ \text{foldl} \delta : \nu \circ \text{foldl} \delta P \stackrel{\circ}{=} P$$

$$\nu^* : \nu (P * Q) \leftrightarrow (\nu P \times \nu Q)$$

$$\delta^* : \delta (P * Q) a \longleftrightarrow \nu P \cdot \delta Q a \cup \delta P a * Q$$

-- etc

ν and δ are *higher-order functions*. How to apply these rules?

Same challenge in differential calculus.

Solve via *symbolic* or *automatic* differentiation.

Regular expressions (inductive)

data Lang : $\diamond\text{Lang} \rightarrow \text{Set} (\text{succ } \ell)$ where

\emptyset : Lang $\diamond\emptyset$

\mathcal{U} : Lang $\diamond\mathcal{U}$

$_ \cup _$: Lang $P \rightarrow \text{Lang } Q \rightarrow \text{Lang } (P \diamond\cup Q)$

$_ \cap _$: Lang $P \rightarrow \text{Lang } Q \rightarrow \text{Lang } (P \diamond\cap Q)$

$_ \cdot _$: Dec $s \rightarrow \text{Lang } P \rightarrow \text{Lang } (s \diamond\cdot P)$

$\mathbf{1}$: Lang $\diamond\mathbf{1}$

$_ * _$: Lang $P \rightarrow \text{Lang } Q \rightarrow \text{Lang } (P \diamond* Q)$

$_ \star$: Lang $P \rightarrow \text{Lang } (P \diamond\star)$

$_ \langle$: $(a : A) \rightarrow \text{Lang } (\diamond\langle a)$

$_ \blacktriangleleft$: $(Q \longleftrightarrow P) \rightarrow \text{Lang } P \rightarrow \text{Lang } Q$

ν : Lang $P \rightarrow \text{Dec } (\diamond\nu P)$

δ : Lang $P \rightarrow (a : A) \rightarrow \text{Lang } (\diamond\delta P a)$

$\llbracket _ \rrbracket^?$: Lang $P \rightarrow \text{Decidable } P$

$\llbracket p \rrbracket^? \quad [] = \nu p$

$\llbracket p \rrbracket^? (a :: w) = \llbracket \delta p a \rrbracket^? w$

Symbolic differentiation (column-major / patterns)

$$\nu \emptyset = \perp?$$

$$\delta \emptyset a = \emptyset$$

$$\nu \mathcal{U} = \top?$$

$$\delta \mathcal{U} a = \mathcal{U}$$

$$\nu (p \cup q) = \nu p \uplus \nu q$$

$$\delta (p \cup q) a = \delta p a \cup \delta q a$$

$$\nu (p \cap q) = \nu p \times \nu q$$

$$\delta (p \cap q) a = \delta p a \cap \delta q a$$

$$\nu (s \cdot p) = s \times \nu p$$

$$\delta (s \cdot p) a = s \cdot \delta p a$$

$$\nu \mathbf{1} = \nu \mathbf{1} \triangleleft \top?$$

$$\delta \mathbf{1} a = \delta \mathbf{1} \triangleleft \emptyset$$

$$\nu (p * q) = \nu * \triangleleft (\nu p \times \nu q)$$

$$\delta (p * q) a = \delta * \triangleleft (\nu p \cdot \delta q a \cup \delta p a * q)$$

$$\nu (p \star) = \nu \star \triangleleft (\nu p \star?)$$

$$\delta (p \star) a = \delta \star \triangleleft (\nu p \star? \cdot (\delta p a * p \star))$$

$$\nu (c a) = \nu c \triangleleft \perp?$$

$$\delta (c a) = \delta c \triangleleft ((a \stackrel{?}{=} c) \cdot \mathbf{1})$$

$$\nu (f \triangleleft p) = f \triangleleft \nu p$$

$$\delta (f \triangleleft p) a = f \triangleleft \delta p a$$

Tries (coinductive)

record Lang ($P : \diamond \text{Lang}$) : Set (suc ℓ) where
coinductive
field

$v : \text{Dec } (\diamond v P)$

$\delta : (a : A) \rightarrow \text{Lang } (\diamond \delta P a)$

$\llbracket _ \rrbracket^? : \text{Lang } P \rightarrow \text{Decidable } P$

$\llbracket p \rrbracket^? \quad \square = v p$

$\llbracket p \rrbracket^? (a :: w) = \llbracket \delta p a \rrbracket^? w$

$\emptyset : \text{Lang } \diamond \emptyset$

$\mathcal{U} : \text{Lang } \diamond \mathcal{U}$

$_ \cup _ : \text{Lang } P \rightarrow \text{Lang } Q \rightarrow \text{Lang } (P \diamond \cup Q)$

$_ \cap _ : \text{Lang } P \rightarrow \text{Lang } Q \rightarrow \text{Lang } (P \diamond \cap Q)$

$_ \cdot _ : \text{Dec } s \rightarrow \text{Lang } P \rightarrow \text{Lang } (s \diamond \cdot P)$

$\mathbf{1} : \text{Lang } \diamond \mathbf{1}$

$_ * _ : \text{Lang } P \rightarrow \text{Lang } Q \rightarrow \text{Lang } (P \diamond * Q)$

$_ \star : \text{Lang } P \rightarrow \text{Lang } (P \diamond \star)$

$_ \text{' } : (a : A) \rightarrow \text{Lang } (\diamond \text{' } a)$

$_ \blacktriangleleft _ : (Q \longleftrightarrow P) \rightarrow \text{Lang } P \rightarrow \text{Lang } Q$

Automatic differentiation (row-major / copatterns)

$$\mathbf{v} \emptyset = \perp?$$

$$\delta \emptyset a = \emptyset$$

$$\mathbf{v} \mathcal{U} = \top?$$

$$\delta \mathcal{U} a = \mathcal{U}$$

$$\mathbf{v} (p \cup q) = \mathbf{v} p \uplus \mathbf{v} q$$

$$\delta (p \cup q) a = \delta p a \cup \delta q a$$

$$\mathbf{v} (p \cap q) = \mathbf{v} p \times \mathbf{v} q$$

$$\delta (p \cap q) a = \delta p a \cap \delta q a$$

$$\mathbf{v} (s \cdot p) = s \times \mathbf{v} p$$

$$\delta (s \cdot p) a = s \cdot \delta p a$$

$$\mathbf{v} \mathbf{1} = \mathbf{v} \mathbf{1} \triangleleft \top?$$

$$\delta \mathbf{1} a = \delta \mathbf{1} \triangleleft \emptyset$$

$$\mathbf{v} (p * q) = \mathbf{v} * \triangleleft (\mathbf{v} p \times \mathbf{v} q)$$

$$\delta (p * q) a = \delta * \triangleleft (\mathbf{v} p \cdot \delta q a \cup \delta p a * q)$$

$$\mathbf{v} (p \star) = \mathbf{v} \star \triangleleft (\mathbf{v} p \star?)$$

$$\delta (p \star) a = \delta \star \triangleleft (\mathbf{v} p \star? \cdot (\delta p a * p \star))$$

$$\mathbf{v} (‘ a) = \mathbf{v} ‘ \triangleleft \perp?$$

$$\delta (‘ c) a = \delta ‘ \triangleleft ((a \stackrel{?}{=} c) \cdot \mathbf{1})$$

$$\mathbf{v} (f \triangleleft p) = f \triangleleft \mathbf{v} p$$

$$\delta (f \triangleleft p) a = f \triangleleft \delta p a$$

- Simple, non-computational formal specification.
- Reasoning from propositionally defined languages to decidable parsing via decision types.
- Duality of regular expressions and tries (symbolic and automatic differentiation).

Same code with dual interpretations.

Also in the paper

- Termination checking
- Generalizing languages
- Algebraic properties of languages
- Transporting algebraic properties