Measures on Languages

1 Introduction

1.1 Regular Languages

Definition 1 (Alphabet)**.** An alphabet Σ is a countable set of distinct *characters*. □

Example 1. The English alphabet of 26 (lower-case) characters forms an alphabet where we have $\Sigma = \{a, b, \ldots, y, z\}$. □

Definition 2 (Language)**.** Let Σ^* be the set of all (possibly countably infinite) strings of a language. A language L is a subset $L \subseteq \Sigma^*$. Denote the set of languages as $\mathcal{L}$. □

Definition 3 (Regular Language)**.** A regular language L_R is a language that can be *recognized* by a *deterministic finite automata* (DFA). In other words, there exists a DFA A_{L_R} such that:

$$\forall s \in L_R : A_{L_R}(s) = 1$$

Denote the set of regular languages as $\mathcal{L}_R$. □

First, all regular languages are languages: $\mathcal{L}_R \subseteq \mathcal{L}$.

It is well known that DFAs and regular languages form a bijection. Furthermore, all DFAs can be reduced to a canonical form. If a DFA A_{L_R} recognizes a regular language L_R, write $\eta(A_{L_R})$ to be a canonical DFA.

DFA's can also be described via *regular expressions*.

Definition 4 (Regular Expression)**.** A regular expression E is a finite string over an alphabet Σ defined inductively, such that:

- The empty string ε is a regular expression
- If E_1 and E_2 are regular expressions, then $E_1 + E_2$ is a regular expression
- If E_1 and E_2 are regular expressions, then $E_1 \cdot E_2$ is a regular expression

- If E is a regular expression, then E^* is a regular expression

If E is a regular expression, denote the regular language generated by it as $L_R(E)$. We implicitly assume a common alphabet, and denote the set of regular expression as $\mathcal{E}$. □

Regular expressions textually represent DFAs, and can be thought of as rules for how to generate a *regular language*. There should be a theorem somewhere about a bijection between regular expressions and DFAs. Here, the $+$ operator can be thought of as string choice, the $\cdot$ operator can be thought of as string concatenation, and the $*$ operator (Kleene star) denotes zero or more occurrences of a regular expression. We often elide $\cdot$ when the context is clear. The $\cdot$ syntactically binds tighter than $+$, meaning that "multiplication" distributes over "addition".

Example 2. Consider the alphabet $\Sigma = \{a, b, c, d\}$, with regular expression $E = ab + c + d^*$. Then the regular language $L_R(E)$ is the set:

$$L_R(E) = \{ab, c, \varepsilon, d, dd, ddd, \dots\}$$

□

Finite language union, intersection, and complement with respect to a common alphabet is also well-defined for regular languages.

We can likewise also define a canonical representation for a regular expression $\eta(E)$ with the following features:

$$\eta(a) = a \qquad \eta(\varepsilon) = \varepsilon \qquad \eta(E^*) = \eta(E)^*$$
$$\eta(E) = \eta(E_1) + \eta(E_2) \iff L_R(E_1) \cap L_R(E_2) = \emptyset$$
$$\eta(E) = \eta(E_1) \cdot \eta(E_2) \iff L_R(E_1) \cap L_R(E_2) = \emptyset$$

We take the existence of this magical η function for granted, because the properties that it gives for regular expression reduction are invaluable for later.

1.2 Measure Theory

Some talk about measure theory goes here

2 Pre-measures on Regular Languages

We make a few observations. First, languages are sets, and by extension so are regular languages. Second, regular languages are closed under finite union, finite intersection, and complement. This means that regular languages therefore forms an algebra closed under these operations.

2.1 Counting Language Size

The counting measure is well-known to be a measure, and therefore a pre-measure, on the algebra of sets with operations of union, intersection, and complement. Therefore, it is straightforward to use the counting measure to measure the size of a language L. Let us define the counting measure $\mu_c \colon \mathcal{L} \to [0, \infty]$ on (regular) languages as follows:

$$\mu_c(L) = |L|$$

2.2 Counting Regular Expressions

We have defined a convenient and natural way to count the size of a (regular) language, and it remains for us to apply a similar notion of (pre)-measure to the regular expressions that generate these languages. Recall that regular languages form an algebra, and therefore so do the canonicalized regular expressions that generate them. We wish to then define a counting pre-measure that counts the size of a language given the regular expression form. We attempt an inductive definition of $\mu_{0,c} \colon \mathcal{E} \to [0, \infty]$ on regular expressions as follows:

$$\begin{aligned}
\mu_{0,c}(\emptyset) = 0 \qquad & \forall a \in \Sigma : \mu_{0,c}(\{a\}) = 1 \\
\eta(E) = E_1 + E_2 \implies & \mu_{0,c}(L_R(E)) = \mu_{0,c}(L_R(E_1)) + \mu_{0,c}(L_R(E_2)) \\
\eta(E) = E_1 \cdot E_2 \implies & \mu_{0,c}(L_R(E)) = \mu_{0,c}(L_R(E_1)) \cdot \mu_{0,c}(L_R(E_2)) \\
\eta(E) = E_1^* \implies & \mu_{0,c}(L_R(E)) = \begin{cases} \infty & \mu_{0,c}(E_1) > 0 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

Observe that almost by definition, this counts precisely the size of the regular language generated by the canonicalized regular expression. In other words:

$$\mu_{0,c}(E) = \mu_c(L_R(E))$$

We assume a bijection exists between canonicalized regular expressions and regular languages (cite theorem from somewhere), which means that this mapping is unique for each language. Therefore, the counting pre-measure $\mu_{0,c}$ defined on regular expressions this way is indeed a pre-measure.

3 Measures on Languages

I saw someone [**aaa**] do it with σ-algebras, and I now feel brave.

3.1 The Language σ-algebra

Let $\mathcal{L}$ be a family of languages over the alphabet Σ. Take Σ^* to be the set of all finite strings. Then define $\sigma(\mathcal{L})$ the σ-algebra over $\mathcal{L}$ to be such that:

(1)

$$\emptyset, \Sigma^* \in \sigma(\mathcal{L})$$

(2)

$$L \in \sigma(\mathcal{L}) \implies L^c = \Sigma^* \setminus L \in \sigma(\mathcal{L})$$

(3)

$$L_1, L_2, \ldots \in \sigma(\mathcal{L}) \implies \bigcup_{n=1}^{\infty} L_i \in \sigma(\mathcal{L})$$

If $\mathcal{L}$ happens to be a family of regular languages, there is no guarantee that $\sigma(\mathcal{L})$ is also a regular family.

3.2 Weighting String Lenghts

We now attempt to define the measure on a language $L \in \mathcal{L}$. Consider a partition of L into strings of finite lenghts:

$$L = L_0 \cup L_1 \cup L_2 \cup \ldots$$

We may assign a discrete σ-finite measure λ over the natural numbers $\mathbb{N}$, and adapt it to $\mathcal{L}$, such that:

$$\forall i \in \mathbb{N} : \lambda(L_i) < \infty$$

If we take a finite alphabet $|\Sigma| < \infty$, then this yields that:

$$\forall s \in L_i : \lambda(\{s\}) = \begin{cases} \frac{\lambda(L_i)}{|L_i|} & |L_i| > 0 \\ 0 & \text{otherwise} \end{cases}$$

We claim that this is is a measure on $\sigma(\mathcal{L})$.