A Trick to Prove Regularity

Arpon Basu

October 25, 2023

Let $\mathcal{L}$ be a regular language, and let $\mathcal{A} := (Q, \Sigma, \delta, q_0, \mathcal{F})$ be a DFA recognizing $\mathcal{L}$. Note that Q is the set of states of $\mathcal{A}$, $q_0 \in Q$ is the start state, $\mathcal{F} \subseteq Q$ is the set of final states, Σ is our alphabet, and $\delta : Q \times \Sigma \mapsto Q$ is the transition function.
Given $\mathcal{L}$, we often want to prove that some "transformed" version of $\mathcal{L}$ is also regular. For example, consider the following transformations:

1. $\text{Cycle}(\mathcal{L}) := \{vu : u, v \in \Sigma^*, uv \in \mathcal{L}\}$
2. $\text{CubeRoot}(\mathcal{L}) := \{w : w \in \Sigma^*, w^3 \in \mathcal{L}\}$

The typical method to show that these languages are regular is to transform the "wiring" of the DFA of $\mathcal{A}$ to get the desired language (see [1]). However, this is often clumsy, and it occasionally takes quite some effort to show that the transformed DFA recognizes the transformed language.
Our approach, on the other hand, reduces such problems (which are common exercises in automata theory courses) to the simple application of a "template".
Indeed, for any $\alpha, \beta \in Q$, define $\mathcal{L}(\alpha, \beta)$ to be the regular language recognized by the DFA $(Q, \Sigma, \delta, \alpha, \{\beta\})$, i.e. we take $\mathcal{A}$ and change the starting state to α and end state to β. Clearly, $\mathcal{L}(\alpha, \beta)$ is regular. Now, note that:

$$\text{Cycle}(\mathcal{L}) := \bigcup_{\substack{q \in Q \\ f \in \mathcal{F}}} \mathcal{L}(q, f) \cdot \mathcal{L}(q_0, q)$$

$$\text{CubeRoot}(\mathcal{L}) := \bigcup_{\substack{q_1, q_2 \in Q \\ f \in \mathcal{F}}} \mathcal{L}(q_0, q_1) \cap \mathcal{L}(q_1, q_2) \cap \mathcal{L}(q_2, f)$$

Since regular languages are closed under union, intersection, and concatenation, we're done. Very clearly, a long-winded argument involving automata was reduced to a one-line proof exploiting the properties of regular languages.

References

[1] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to automata theory, languages, and computation, 2nd edition*. Vol. 32. ACM Press, 2001.