بسم اللّه الرّحمن الرّحیم

دانشگاه صنعتی اصفهان ـ دانشکدهٔ مهندسی برق و کامپیوتر
(نیم‌سال تحصیلی ۴۰۲۱)

نظریهٔ زبان‌ها و ماشین‌ها

حسین فلسفین

دربارهٔ مفهوم بسته بودن تحت یک عملگر

Let $\mathbb{N} = \{1, 2, 3, \ldots\}$ be the set of natural numbers. When we say that $\mathbb{N}$ is closed under multiplication, we mean that for any x and y in $\mathbb{N}$, the product $x \cdot y$ also is in $\mathbb{N}$. In contrast, $\mathbb{N}$ is not closed under division, as 1 and 2 are in $\mathbb{N}$ but $\frac{1}{2}$ is not. Generally speaking, a collection of objects is closed under some operation if applying that operation to members of the collection returns an object still in the collection. We show that the collection of regular languages is closed under all three of the regular operations.

Theorem: The class of regular languages is closed under the union operation. In other words, if A_1 and A_2 are regular languages, so is $A_1 \cup A_2$.

PROOF IDEA We have regular languages A_1 and A_2 and want to show that $A_1 \cup A_2$ also is regular. Because A_1 and A_2 are regular, we know that some finite automaton M_1 recognizes A_1 and some finite automaton M_2 recognizes A_2. To prove that $A_1 \cup A_2$ is regular, we demonstrate a finite automaton, call it M, that recognizes $A_1 \cup A_2$. This is a proof by construction. We construct M from M_1 and M_2. Machine M must accept its input exactly when either M_1 or M_2 would accept it in order to recognize the union language.

☞ *It works by simulating both M_1 and M_2 and accepting if either of the simulations accept. How can we make machine M simulate M_1 and M_2? Perhaps it first simulates M_1 on the input and then simulates M_2 on the input. But we must be careful here! Once the symbols of the input have been read and used to simulate M_1, we can't "rewind the input tape" to try the simulation on M_2. We need another approach.*

☞ *Pretend that you are M. As the input symbols arrive one by one, you simulate both M_1 and M_2 simultaneously. That way, only one pass through the input is necessary. But can you keep track of both simulations with finite memory? All you need to remember is the state that each machine would be in if it had read up to this point in the input. Therefore, you need to remember a pair of states. How many possible pairs are there? If M_1 has k_1 states and M_2 has k_2 states, the number of pairs of states, one from M_1 and the other from M_2, is the product $k_1 \cdot k_2$.*

☞ *This product will be the number of states in M, one for each pair. The transitions of M go from pair to pair, updating the current state for both M_1 and M_2. The accept states of M are those pairs wherein either M_1 or M_2 is in an accept state.*

PROOF

Let M_1 recognize A_1, where $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$, and M_2 recognize A_2, where $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$. Construct M to recognize $A_1 \cup A_2$, where $M = (Q, \Sigma, \delta, q_0, F)$.

1. $Q = \{(r_1, r_2) | r_1 \in Q_1$ and $r_2 \in Q_2\}$. This set is the Cartesian product of sets Q_1 and Q_2 and is written $Q_1 \times Q_2$. It is the set of all pairs of states, the first from Q_1 and the second from Q_2.

2. Σ, the alphabet, is the same as in M_1 and M_2. In this theorem and in all subsequent similar theorems, we assume for simplicity that both M_1 and M_2 have the same input alphabet Σ. The theorem remains true if they have different alphabets, Σ_1 and Σ_2. We would then modify the proof to let $\Sigma = \Sigma_1 \cup \Sigma_2$.

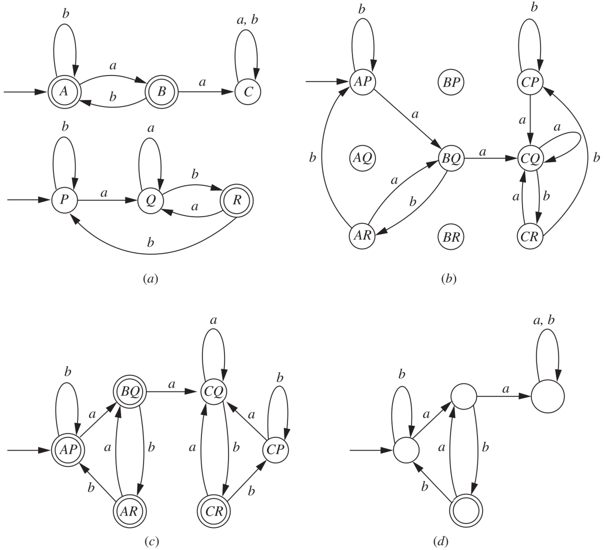
3. δ, the transition function, is defined as follows. For each $(r_1, r_2) \in Q$ and each $a \in \Sigma$, let $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$. Hence δ gets a state of M (which actually is a pair of states from M_1 and M_2), together with an input symbol, and returns M's next state.

4. q_0 is the pair (q_1, q_2).

5. F is the set of pairs in which either member is an accept state of M_1 or M_2. We can write it as $F = \{(r_1, r_2) \mid r_1 \in F_1 \text{ or } r_2 \in F_2\}$. This expression is the same as $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$. (Note that it is not the same as $F = F_1 \times F_2$. What would that give us instead?)

We have just shown that the union of two regular languages is regular, thereby proving that the class of regular languages is closed under the union operation.

Example: $L_1 = \{x \in \{a, b\}^* \mid aa$ ***is not a substring of*** $x\}$ **&** $L_2 = \{x \in \{a, b\}^* \mid x$ ***ends with*** $ab\}$



☜ چهار عملگر باینری:

$$L_1 \cup L_2 \quad L_1 \cap L_2 \quad L_1 \setminus L_2 \quad L_1 \circ L_2$$

☜ دو عملگر ***unary***:

$$\overline{L_1} \quad L_1^*$$

☜ قبلاً دربارهٔ بسته بودن تحت دو عملگر اجتماع و اشتراک حرف زدیم و بسته بودن را اثبات کردیم.

☜ اکنون به سراغ دو عملگر مکمل (***complement***) و تفاضل خواهیم رفت.

The operation of complement is with regard to Σ^* and hence the complement of $L \subseteq \Sigma^*$ denoted by $\overline{L}$, is such that $\overline{L} = \Sigma^* \setminus L$.

☞ *Let L be a regular language and let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA that accepts L. Then, we consider the DFA $M' = (Q, \Sigma, \delta, q_0, Q \setminus F)$ by exchanging the roles of the accept states and non-accept states of M. Clearly M' accepts the complement $\overline{L} = \Sigma^* \setminus L$ of L. Thus, we can conclude that the class of regular languages is closed under the operation of complement.*

☞ *The regular languages are closed under set difference (subtraction). Why? We note that:*

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

یک راه دیگر برای اثبات این مطلب که کلاس زبان‌های منظم تحت عملگر اشتراک بسته است:

Here is another way of proving that the class of regular languages is closed under the operation of intersection. First, let us note that for regular languages L_1 and L_2, their intersection is expressed by using De Morgan's law as

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}.$$

On the other hand, since the class of regular languages is closed under union and complement, as already shown, it is also closed under intersection, which is expressed as above in terms of union and complement.

دربارهٔ اثبات بسته بودن تحت دو عملگر $*$ و $\circ$ چه می‌توان گفت؟
اثبات بسته بودن تحت این دو عملگر، در مرحلهٔ کنونی سخت است! احتیاج به معرفی یک مفهوم مهم و اساسی داریم:

Nondeterminism!

در ادامهٔ بحث ذیل عنوان زبان‌های منظم، به این مفهوم مهم خواهیم پرداخت انشاءالله.