UMC205: Automata and Computability

Naman Mishra

January 2024

Contents

Lecture 02: Thu 04 Jan '24

1 Languages

Definition 1.1. An *alphabet* is a non-empty finite set of symbols or "letters".

A *string* or *word* over an alphabet A is a finite sequence of letters from A. Equivalently, a string is a map from a prefix (possibly empty) of $\mathbb{N}$ to A. The length of a string s, notated $\#s$, is the cardinality of its domain. The empty string is denoted ϵ.

The set of all strings over A is denoted A^*.

Example. $A = \{a, b, c\}$ and $\Sigma = \{0, 1\}$ are both alphabets. $aaba$ is a string over $A = \{a, b, c\}$.

Proposition 1.2. Let A be an alphabet. Then A^* is countably infinite.

Proof. Let $n = \#A$. Let $f : A \to \{1, \dots, n\}$ be a numbering of A. Replacing each letter in a string with its number gives a representation of the string as a natural number in base $n + 1$. This gives an injection $A^* \to \mathbb{N}$ and so A^* is countable. Infiniteness is obvious.

Alternatively, consider the strings in their Lexicographic order. □

Definition 1.3 (Language)**.** A *language* over an alphabet A is a subset of A^*.

Example. Let $A = \{a, b, c\}$. Then $\{abc, aaba\}$, $\{\epsilon, b, aa, bb, aab, aba, bbb, \ldots\}$, $\{\epsilon\}$, $\{\}$ are all languages over A.

Definition 1.4 (Concatenation)**.** Let u, v be strings over an alphabet A. Then $u \cdot v$ or simply uv is the string obtained by appending v to the end of u.

For two languages L_1, L_2 over A, define their concatenation

$$L_1 \cdot L_2 \coloneqq \{uv \mid u \in L_1, v \in L_2\}.$$

We will also write ua where u is a string and a is a letter to mean u concatenated with the string of length 1 consisting of the letter a.

Definition 1.5 (Lexicographic order)**.** Let $(A, <)$ be a totally ordered alphabet. We say $u < v$ for $u, v \in A^*$ if either $\#u < \#v$ or $\#u = \#v$ and $u = pxu'$, $v = pyv'$ for some $p, u', v' \in A^*$ and $x, y \in A$ with $x < y$.

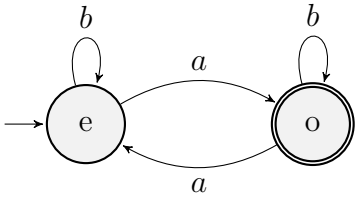
This is called the *lexicographic order* on A^*.

Proposition 1.6. Let A be an alphabet. Then the set of all languages over A is uncountable.

Proof. Diagonilization. Suppose there is an enumeration L of all languages over A. Let s be an enumeration of A^*. Then define $L' = \{s \in A^* \mid s \notin L_s\}$. Then L' is a language over A that is not in L. □

Definition 1.7 (Concatenations)**.** Let L_1, L_2 be languages over an alphabet A. Then $L_1 \cdot L_2$ is the language $\{uv \mid u \in L_1, v \in L_2\}$.

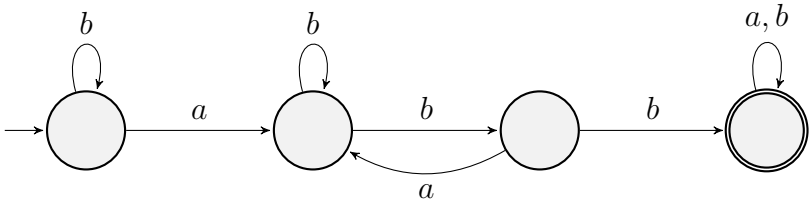
2 Deterministic Finite-State Automata



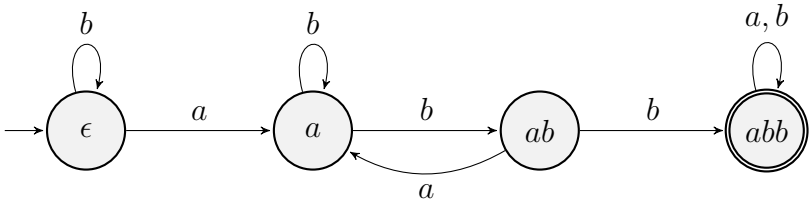
Each state represents a property of the input string read so far. State e is the start state, and state o is the only accepting state.

In this case, State e corresponds to an even number of a's read, and State o corresponds to an odd number of a's read. This can be proven by induction to conclude that the automaton accepts the language $\{w \in \{a, b\}^* \mid \#_a(w)\}$.

Example. Let $A = \{a, b\}$. Consider the DFA



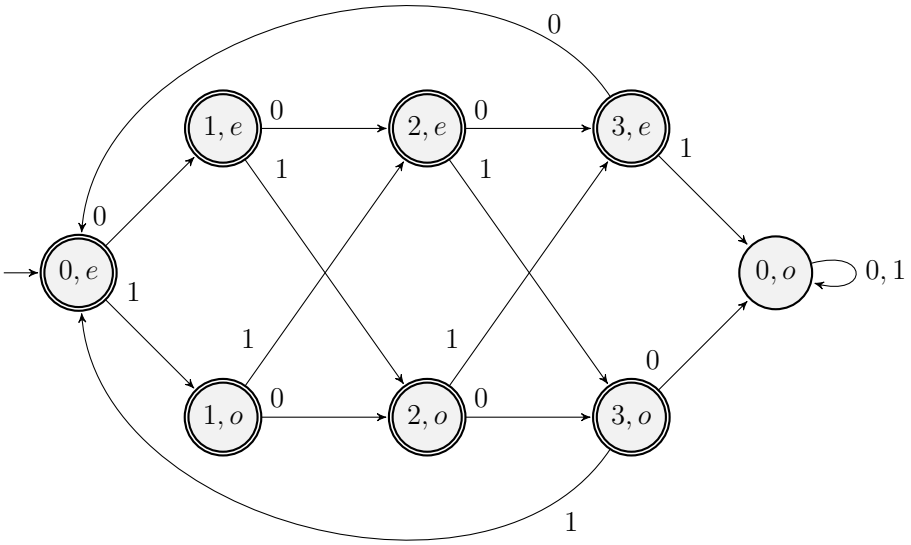
We label the nodes as ϵ, a, ab and abb



and consider the property corresponding to each state.

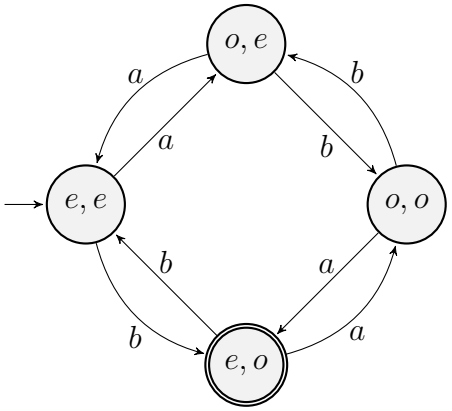
- State ϵ

Another example of a DFA which accepts strings over $\{0, 1\}$ which have even parity in each length 4 block.



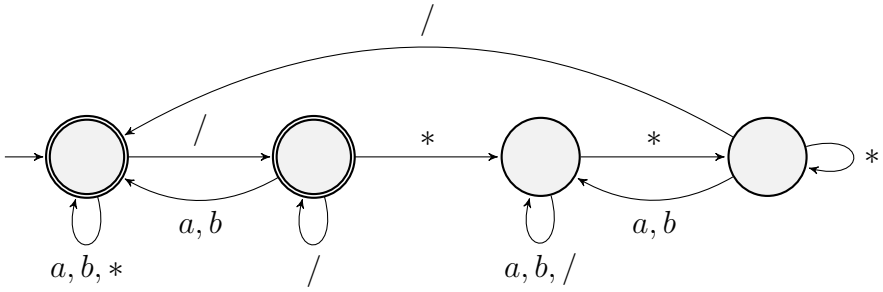
Problem 2.1. Give a DFA that accepts strings over the alphabet $\{a, b\}$ containing an even number of a's and an odd number of b's.

Solution.



Problem 2.2. Give a DFA that accepts strings over $\{a, b, /, *\}$ which don't end inside a C-style comment, *i.e.*, comments of the form `/* ... */`.

Solution.



Lecture 03: Tue 09 Jan '24

2.1 A good way to construct DFAs

Suppose we have to construct a DFA for a language L over an alphabet A.

- Think of a finite number of properties of strings that you might want to keep track of. For example, "number of a's seen so far is even".
- Identify an initial property that is true of the empty string, say p_0.
- Make sure there is a rule to update the properties which are being tracked for a string wa, based purely on the properties for w and the last input a.
- The properties should imply membership in L or non-membership in L.

2.2 DFAs Formally

Definition 2.1 (DFA)**.** A *deterministic finite-state automaton* $\mathcal{A}$ over an alphabet A is a tuple (Q, s, δ, F) where

- Q is a finite set of *states*,
- $s \in Q$ is the *start state*,
- $\delta : Q \times A \to Q$ is the *transition function*,
- $F \subseteq Q$ is the set of *final states*.

For example, the first example in section 2 can be written as

$$\begin{aligned} A &= \{a, b\} \\ Q &= \{e, o\} \\ s &= e \\ F &= \{o\} \\ \delta(e, a) &= o \\ \delta(e, b) &= e \\ \delta(o, a) &= e \\ \delta(o, b) &= o \end{aligned}$$

We further define $\hat{\delta} : Q \times A^* \to Q$ as the extension of δ to strings.

$$\begin{aligned} \hat{\delta}(q, \epsilon) &= q \\ \hat{\delta}(q, wa) &= \delta(\hat{\delta}(q, w), a) \end{aligned}$$

Definition 2.2 (Language of a DFA)**.** The *language of a DFA* $\mathcal{A}$ is

$$L(\mathcal{A}) = \left\{ w \in A^* \mid \hat{\delta}(s, w) \in F \right\}$$

2.3 Regular Languages

Definition 2.3 (Regular Language)**.** A language L is *regular* if there exists a DFA $\mathcal{A}$ over A such that $L(\mathcal{A}) = L$.

For example, the exercises we have done so far. Another example is *any* finite language.

Theorem 2.4. The class of regular languages over an alphabet is countable.

Proof. We partition the set of all DFAs over A by their number of states. For each $n \in \mathbb{N}$, there are finitely many DFAs with n states. A countable union of finite sets is countable. Thus the set of all DFAs over A is countable. Since each regular language corresponds to at least one DFA, the set of all regular languages over A is countable. □

However, we have seen that there are uncountably many languages over any alphabet. This immediately yields the following.

Corollary 2.5. There are uncountably many languages that are not regular.

Theorem 2.6 (Closure under set operations)**.** The class of regular languages is closed under union, intersection and complementation.

Proof. For complementation, simply invert the set of final states. That is, given $\mathcal{A} = (Q, s, \delta, F)$, let $\mathcal{A}' = (Q, s, \delta, Q \setminus F)$. Then $L(\mathcal{A}') = A^* \setminus L(\mathcal{A})$, since

$$\begin{aligned} w \in L(\mathcal{A}') &\iff \hat{\delta}(s, w) \in Q \setminus F \\ &\iff \hat{\delta}(s, w) \notin F \\ &\iff w \notin L(\mathcal{A}) \\ &\iff w \in A^* \setminus L(\mathcal{A}) \end{aligned}$$

For intersection and union, define the *product* of two DFAs.

Definition 2.7 (Product)**.** Given two DFAs $\mathcal{A} = (Q, s, \delta, F)$ and $\mathcal{B} = (Q', s', \delta', F')$ over the same alphabet A, the *product* of $\mathcal{A}$ and $\mathcal{B}$ is

$$\mathcal{A} \times \mathcal{B} = (Q \times Q', (s, s'), \Delta, F \times F')$$

where $\Delta((q, q'), a) = (\delta(q, a), \delta'(q', a))$.

Note that in the above definition, the extension of Δ to strings $\hat{\Delta}$ is given by

$$\hat{\Delta}((q, q'), w) = (\hat{\delta}(q, w), \hat{\delta}'(q', w))$$

This is easily proved by induction on the length of w (or structural induction on w).

$$\begin{aligned} \hat{\Delta}((q, q'), \epsilon) &= (q, q') \\ &= (\hat{\delta}(q, \epsilon), \hat{\delta}'(q', \epsilon)) \end{aligned}$$

and if

$$\hat{\Delta}((q, q'), w) = (\hat{\delta}(q, w), \hat{\delta}'(q', w))$$

then

$$\begin{aligned} \hat{\Delta}((q, q'), wa) &= \Delta(\hat{\Delta}((q, q'), w), a) \\ &= \Delta((\hat{\delta}(q, w), \hat{\delta}'(q', w)), a) \\ &= (\delta(\hat{\delta}(q, w), a), \delta'(\hat{\delta}'(q', w), a)) \\ &= (\hat{\delta}(q, wa), \hat{\delta}'(q', wa)). \end{aligned}$$

Now let $\mathcal{A}$, $\mathcal{B}$ be DFAs over A. Then $L(\mathcal{A} \times \mathcal{B}) = L(\mathcal{A}) \cap L(\mathcal{B})$, since

$$\begin{aligned} w \in L(\mathcal{A} \times \mathcal{B}) &\iff \hat{\Delta}((s, s'), w) \in F \times F' \\ &\iff (\hat{\delta}(s, w), \hat{\delta}'(s', w)) \in F \times F' \\ &\iff \hat{\delta}(s, w) \in F \wedge \hat{\delta}'(s', w) \in F' \\ &\iff w \in L(\mathcal{A}) \wedge w \in L(\mathcal{B}) \end{aligned}$$

Since $X \cup Y = \overline{\overline{X} \cap \overline{Y}}$, closure under union follows from closure under complementation and intersection.

More directly, the DFA $(Q \times Q', (s, s'), \Delta, F \times Q' \cup F' \times Q)$ accepts the language $L(\mathcal{A}) \cup L(\mathcal{B})$. □

Theorem 2.8 (Closure under concatenation)**.** The class of regular languages is closed under concatenation.

Proof. □

Lecture 04: Thu 11 Jan '24

2.3.1 Two Necessary Conditions for Regular Languages

In a given DFA $\mathcal{A}$ with n states, any path of length greater than n must have a loop. Let u be the string of symbols on the path from the start state to the beginning of the loop, let v be the (non-empty) string of symbols on the loop, and let w be the string of symbols on the path from the end of the loop to the final state.

Then if uvw is accepted by $\mathcal{A}$, then so is $uv^k w$ for any $k \geq 0$.

Theorem 2.9 (Pumping Lemma)**.** For any regular language L, there exists a constant k, such that for any word $t \in L$ of the form xyz with $|y| \geq k$, there exist strings u, v and w such that

(i) $y = uvw$, $v \neq \epsilon$, and

(ii) $xuv^i wz \in L$ for each $i \geq 0$.

Proposition 2.10. The language $\{a^n b^n \mid n \geq 0\}$ is not regular.

Proof. Let $k \in \mathbb{N}$. Choose $t = a^k b^k = xyz$ where $x = \epsilon$, $y = a^k$, and $z = b^k$. Let $y = uvw$ for some non-empty v. Then $v = a^j$ for some $j \geq 1$. Then $xuv^2wz = a^{k+j}b^k$, which is not in the language. Therefore, the language is not regular. □

Problem 2.3. Show that $\left\{a^{2^n} \mid n \geq 0\right\}$ is not a regular language.

Solution. Let $k \in \mathbb{N}$. Choose $t = a^{2^k} = xyz$ where $x = \epsilon$, $y = a^{2^k-1}$, and $z = a$. Let $y = uvw$ for some non-empty v. Then $v = a^j$ for some $1 \leq j < 2^k$. Then $xuv^2wz = a^{2^k+j}$, which is not in the language since $2^k < 2^k + j < 2^{k+1}$.

Problem 2.4. Is the language $\{w \cdot w \mid w \in \{0,1\}^*\}$ regular?

Proof. Let $k \in \mathbb{N}$. Choose $t = 0^k1^k0^k1^k = xyz$ where $x = 0^k$, $y = 1^k$, and $z = 0^k1^k$. Let $y = uvw$ for some non-empty v. Then $v = 1^j$ for some $1 \leq j \leq k$. If j is odd, we are done. Otherwise, $xuv^2wz = 0^k1^{k+m}1^m0^k1^k$, where $j = 2m$. This is not in the language since the second half starts with a 1. □