Week 9: Equivalence of Regular Expressions, DFA, and NFA

CSC 236:Introduction to the Theory of Computation

Summer 2024

Instructor: Lily

Announcement

- Got suggestion for review topics? Discuss on Piazza!
  - Proof of correctness
  - Structural induction
- Final exam schedule is coming out soon:
  - Number of questions: 8 + bonus
  - Q1 + Q2: T/F, multiple choice, short answer
  - Q3, 4, 5, 6, 7, 8: 20% “I don’t know” policy
  - Aid: two-sided A4 sheet
  - Duration: 3 hours
- Tutorials this week: regular expressions practice

Review

- A **language** L over an alphabet Σ is a subset of Σ^*.
- **Regular expressions** describes languages. **Regular languages** are sets of strings which can be *represented* by a regular expression.
- **Finite automatons** are collections of states and transition rules upon those states base on input characters (and ϵ in the case of NFA) in Σ**.**

DFA	NFA
• Each state ρ has exactly one transition for each symbol $a \in \Sigma$ • ϵ-transitions *not* allowed	• ρ has any number of transitions on a symbol $a \in \Sigma$ • ϵ-transitions *are* allowed

Question: DFA vs NFA, which is more powerful?

Finite Automaton Correctness

Consider

$$L = \{w \in \{0,1\}^*: w \text{ has odd number } 0s \text{ and even number } 1s\}$$

Construct DFA M such that $\mathcal{L}(M) = L$.

Proving Correctness of DFA

For DFA $M = (Q, \Sigma, \delta, s, F)$, construct *disjoint* and *exhaustive* **state invariant** for each state $q \in Q$.

Proving Correctness of DFA

For DFA $M = (Q, \Sigma, \delta, s, F)$, construct *disjoint* and *exhaustive* **state invariant** for each state $q \in Q$.

Equivalence

Want to show: Regular expression, DFA, and NFAs have the same expressive power.

Application

Suppose L is a regular language, is the *complement* of L, denoted $\bar{L}$, regular? $\bar{L} = \{w \in \Sigma^*: w \notin L\}$.

Application

Suppose L is a regular language, is the *reversal* of L, denoted $\mathrm{Rev}(L)$, regular? $\mathrm{Rev}(L) = \{(w)^R \in \Sigma^* : w \in L\}$.

Now you try!

1. Let L be a language on the alphabet $\Sigma = \{0, \dots, n-1\}$ of size n. Let $s_1, \dots., s_{n-1}$ be string. Let $L(s_0, \dots, s_{n-1})$ is the *replacement language*, where every instance of i is replaced with s_i. Is $L(s_0, \dots, s_{n-1})$ regular?
2. Consider the language
$$L_1 = \{w \in \{0,1\}^* \colon \text{second last letter of } w \text{ is } 0\}.$$

Construct a finite automaton for L_1 and prove it is correct.

3. Consider the language we saw in the previous lecture
$$L_2 = \{0^n 1^m \colon m, n \geq 0, m + n \text{ is odd}\}.$$

Show that the finite automaton for L_2 is correct.

1. Let L be a language on the alphabet $\Sigma = \{0, \ldots, n-1\}$ of size n. Let $s_1, \ldots., s_{n-1}$ be string. Let $L(s_0, \ldots, s_{n-1})$ is the *replacement language*, where every instance of i is replaced with s_i. Is $L(s_0, \ldots, s_{n-1})$ regular?

2. Consider the language

$$L_1 = \{w \in \{0,1\}^*: \text{second last letter of } w \text{ is } 0\}.$$

Construct a finite automaton for L_1 and prove it is correct.

2. Consider the language

$$L_1 = \{w \in \{0,1\}^* : \text{second last letter of } w \text{ is } 0\}.$$

Construct a finite automaton for L_1 and prove it is correct.

3. Consider the language we saw in the previous lecture

$$L_2 = \{0^n 1^m : m, n \geq 0, m + n \text{ is odd}\}.$$

Show that the finite automaton for L_2 is correct.

3. Consider the language we saw in the previous lecture

$$L_2 = \{0^n 1^m : m, n \geq 0, m + n \text{ is odd}\}.$$

Show that the finite automaton for L_2 is correct.

NFA to DFA

Given a NFA $M_N = (Q_N, \Sigma, \delta_N, s_N, F_N)$, construct a DFA $M_D = (Q_D, \Sigma, \delta_D, s_D, F_D)$ such that $\boldsymbol{\mathcal{L}}(M_N) = \boldsymbol{\mathcal{L}}(M_D)$.

NFA to DFA Example

Consider $L = \mathcal{L}\left(\left(0(0+10+1)\right)^*\right)$ and its corresponding NFA.

NFA to DFA

Given a NFA $M_N = (Q_N, \Sigma, \delta_N, s_N, F_N)$, construct a DFA $M_D = (Q_D, \Sigma, \delta_D, s_D, F_D)$ such that $\boldsymbol{\mathcal{L}}(M_N) = \boldsymbol{\mathcal{L}}(M_D)$.

NFA to DFA

General problem format

Given a language L, you will be asked: is L regular?

1. Make a decision: regular or not regular.
2. If regular: proof of regularity.
   - Produce a finite automaton which accepts a language
   - Prove correctness of finite automaton
3. If not regular: proof of non-regularity.

???

Recap

- Proved the correctness of finite automaton using *state invariants* for each state (remember these must be *disjoint* and *exhaustive*).
- Showed that DFA, NFA, and regular expression have the same expressive power (some work left to be done --- left to homework)
  - Reduce NFA to DFA by adding more states; each state represents a subset of states in the NFA (remember: If there are finitely many states then it's okay)
  - **Regular languages are those which are accepted by finite automatons**

Next time… limitation of finite automatons