Introduction to Finite Automata

Fabrizio D'Angelo, Michal Hecko

RedHat

February 25, 2022

Example problem

Task: write a C function `is_str_rh(char* word)` that returns 1 iff the given word is *RedHat*.

Example problem

Task: write a C function `is_str_rh(char* word)` that returns 1 iff the given word is *RedHat*.

```
int is_str_rh(char* word) {
    const char* rh = "RedHat";
    int i;
    for (i = 0; i < 6; i++) {
        if (word[i] == '\0')  break;
        if (rh[i] != word[i]) break;
    }
    return i == 6;
}
```

Example problem

Task: write a C function `is_str_rh(char* word)` that returns 1 iff the given word is *RedHat*.

```
int is_str_rh(char* word) {
    const char* rh = "RedHat";
    int i;
    for (i = 0; i < 6; i++) {
        if (word[i] == '\0')  break;
        if (rh[i] != word[i]) break;
    }
    return i == 6;
}
```

What is the function state?

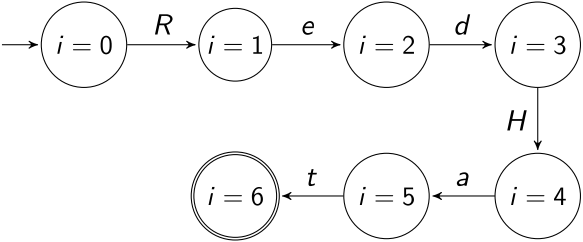


Figure: State space of `is_str_rh(char* word)`

Formal definition of the model

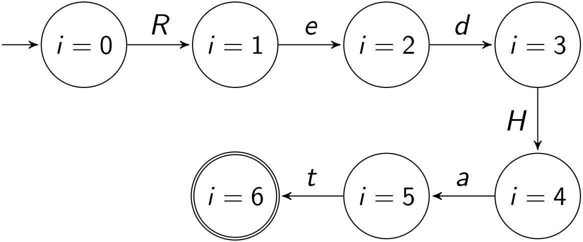
Why?

1. study the entire class of similar problems mathematically
2. decouple the problem structure from the implementation

Formal definition of the model

A *finite automaton* $\mathcal{A}$ is the 5-tuple $(Q, \Sigma, \delta, Q_0, F)$, where:

1. Q is a finite non-empty set of states,
2. Σ is a finite non-empty set of symbols called an alphabet,
3. $\delta \subseteq Q \times \Sigma \times Q$ is the set of transitions,
4. $Q_0 \subseteq Q$ is the set of initial states, and
5. $F \subseteq Q$ is the set of final states.



In our example:

1. $Q = \{i = 0, i = 1, i = 2, i = 3, i = 4, i = 4, i = 5\}$,
2. $\Sigma = \{R, e, d, H, a, t\}$,
3. $\delta = \{(i = 0) \xrightarrow{R} (i = 1), (i = 1) \xrightarrow{e} (i = 2), (i = 2) \xrightarrow{d} (i = 3), (i = 3) \xrightarrow{H} (i = 4), (i = 4) \xrightarrow{a} (i = 5), (i = 5) \xrightarrow{t} (i = 6)\}$
4. $Q_0 = \{(i = 0)\}$
5. $F = \{(i = 6)\}$

Richer automaton

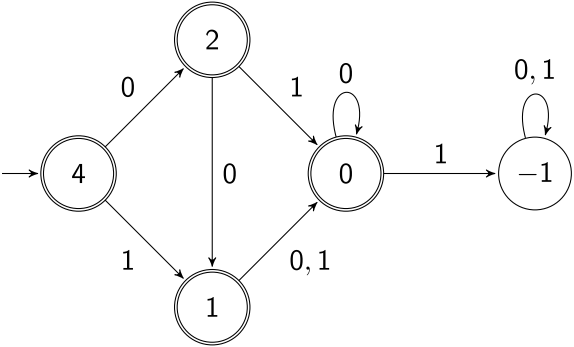
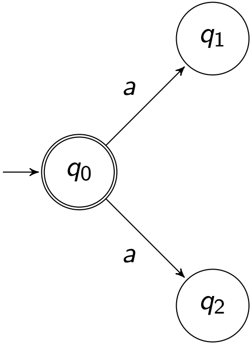


Figure: Automaton A_φ for the inequality φ: $x \leq 4$ over $\mathbb{N}$

Determinism

Notice our definition ($\delta \subseteq Q \times \Sigma \times Q$) allows nondeterminism:



We say that automata that satisfy our original definition are nondeterministic finite automata (NFAs).

However, allowing deterministic automata have some interesting properties (e.g. easy interpretation). Therefore, we define a deterministic finite automaton (DFA) to be a FA $\mathcal{A}$ that further satisfies:

- δ is a function,
- $|Q_0| = 1$ (there is only one initial state).

.

Important definitions

- A word w is a sequence of alphabet letters ($w \in \Sigma^*$).

Important definitions

- A word w is a sequence of alphabet letters ($w \in \Sigma^*$).
- A **run** r of a automaton $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$ over a word w of length n is a sequence of states $r = q_0, q_1, \ldots q_n$ ($q_i \in Q$ for $0 \leq i < n$) such that $q_0 \in Q_0$, and for every $1 \leq i \leq n$ there is a transition $q_{i-1} \xrightarrow{w_i} q_i$.
- A run is **accepting** iff $q_n \in F$.

Important definitions

- A word w is a sequence of alphabet letters ($w \in \Sigma^*$).
- A **run** r of a automaton $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$ over a word w of length n is a sequence of states $r = q_0, q_1, \ldots q_n$ ($q_i \in Q$ for $0 \leq i < n$) such that $q_0 \in Q_0$, and for every $1 \leq i \leq n$ there is a transition $q_{i-1} \xrightarrow{w_i} q_i$.
- A run is **accepting** iff $q_n \in F$.
- A language of an automaton is a set of all words for which an accepting run of $\mathcal{A}$ exists.