Generalised Nondeterministic Finite Automata - Lecture 6

James Marshall

Theorem (*from last lecture*)

language A is described by a regular expression ⇔ language A is regular

Definition - Generalised Nondeterministic Finite Automaton (GNFA)

A ***generalised nondeterministic finite automaton*** *is a 5-tuple* $(Q, \Sigma, \delta, q_{\text{start}}, q_{\text{start}})$ where

1. *Q is the finite set of states*

2. *Σ is the finite input alphabet*

3. δ: $(Q \setminus \{q_{\text{accept}}\}) \times (Q \setminus \{q_{\text{start}}\}) \rightarrow \mathcal{R}$ *defines the transition function, where $\mathcal{R}$ is the set of all regular expressions over the input alphabet*

4. *$q_{\text{start}} \in Q$ is the start state*

5. *$q_{\text{accept}} \in Q$ is the accept state*

Definition - Computation by an GNFA

A GNFA $N = (Q, \Sigma, \delta, q_{\text{start}}, q_{\text{accept}})$ accepts string $w \in \Sigma^*$ if and only if $w = w_1 w_2 ... w_k$ where $w_i \in \Sigma^*$, and a sequence of states $q_0, q_1, q_2 ... q_k$ exists in Q satisfying:

1. $q_0 = q_{\text{start}}$

2. $q_0 = q_{\text{accept}}$

3. $w_i \in L(\delta(q_{i-1}, q_i))$ for all $i = 0, 1, ..., k - 1$

Lemma

language A is regular $\Rightarrow$ language A is described by a regular expression

Proof Sketch (*by construction*)

Step 1: Convert a DFA for language A into an equivalent GNFA

Step 2: Reduce the GNFA until it has a single transition labelled with the regular expression describing language A