Regular expressions

$$\Longleftrightarrow$$

Regular languages

Sipser 1.3 (pages 63-76)

Last time…

Regular expressions

- Definition 1.52:

  Say that R is a *regular expression* if R is

1. a for some a in the alphabet Σ
2. ε
3. ∅
4. $(R_1 \cup R_2)$, where R_1 and R_2 are regular expressions
5. $(R_1 \circ R_2)$, where R_1 and R_2 are regular expressions
6. $(R_1{}^*)$, where R_1 is a regular expression

Regular expressions and NFAs

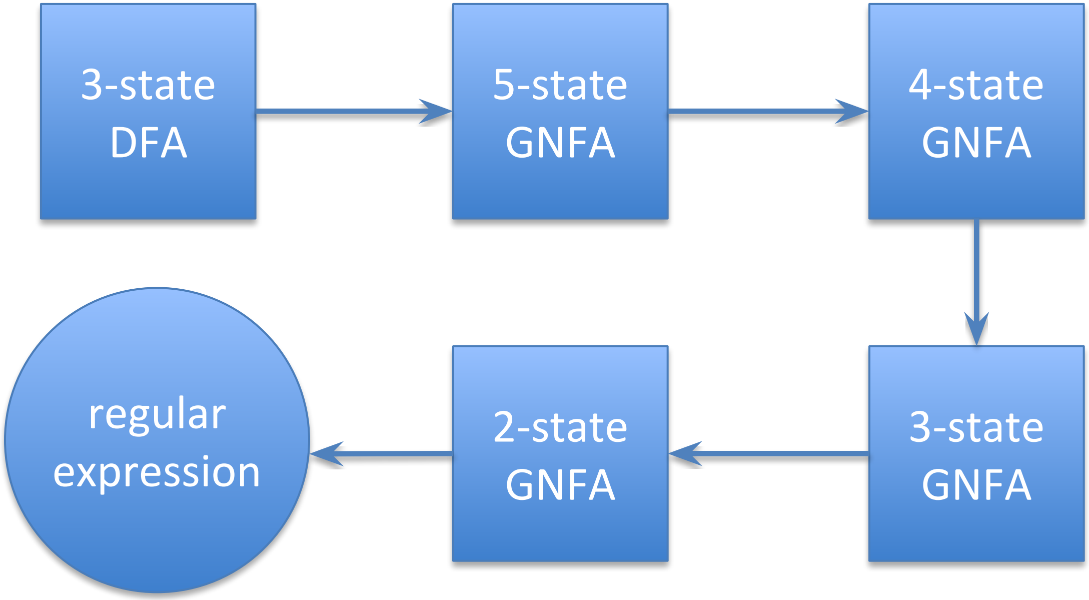
- Theorem 1.54: A language is regular if and only if some regular expression describes it.
- Proof (⇐)

1. ***If a ∈ Σ, then a is regular.***
2. ***ε is regular.***
3. ***$\emptyset$ is regular.***
4. ***If R_1 and R_2 are regular, then $(R_1 \cup R_2)$ is regular.***
5. ***If R_1 and R_2 are regular, then $(R_1 \circ R_2)$ is regular.***
6. ***If R_1 is a regular, then $(R_1{}^*)$ is regular***

Today…

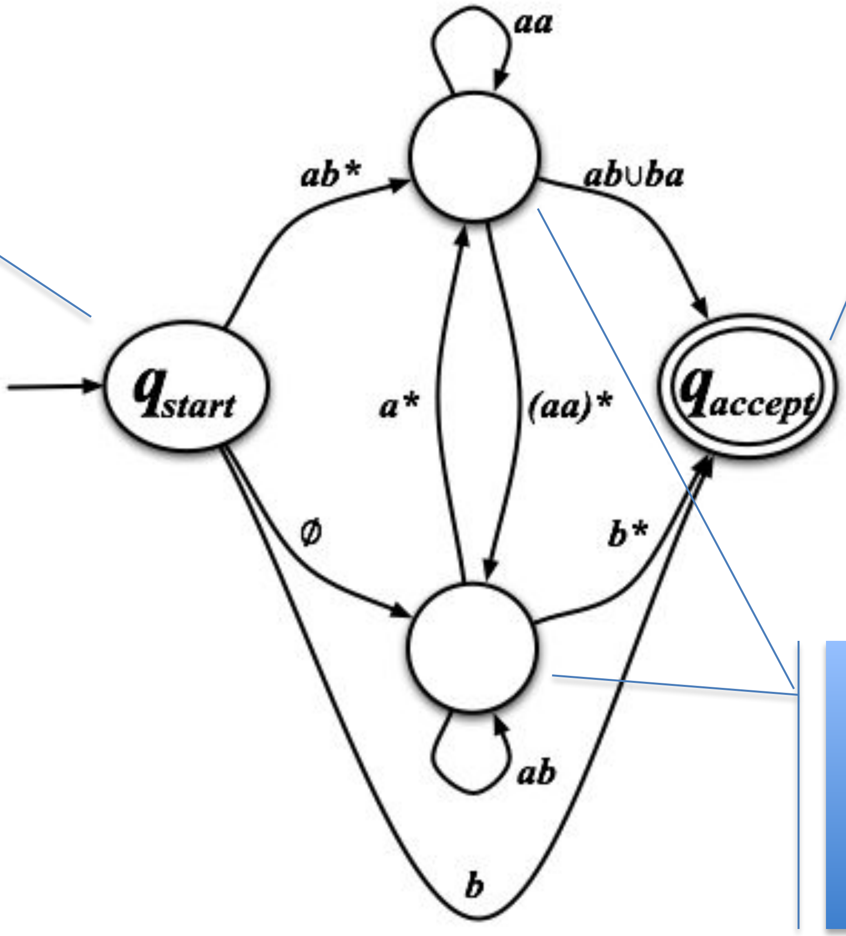
Going forward

- Theorem 1.54: A language is regular if and only if some regular expression describes it.
- (⇒)



Generalized NFA

Transition to every other state, but no transitions in



Transitions in from every other state, but no transitions out

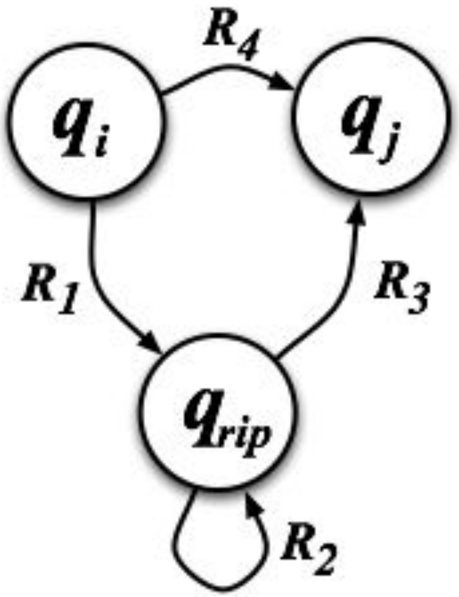
Exactly one arrow to and from every state (including itself) other than start and accept

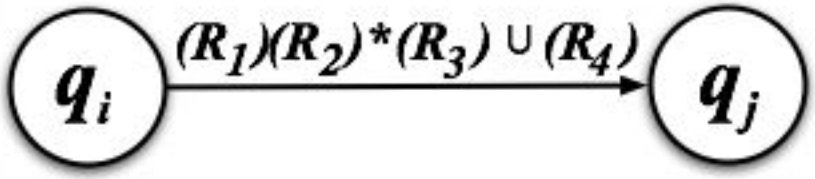
Proof

DFA to GNFA…

- Step 1: Add a unique start state with an ***ε*** jump to the original one
- Step 2: Add a unique accept state with ***ε*** jumps from the previous accept states
- Step 3: Convert multiple labels to ∪
- Step 4: Add Ø jumps for any transition that's missing

Induction step: rip a state





A simple example