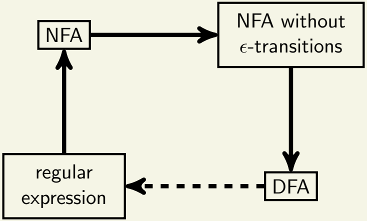
Computability and Complexity
COSC 4200

Regular Expressions II

Theorem

A language is regular if and only if it can be described by a regular expression.



Now: DFA $\rightarrow$ regular expression.

Converting a DFA to a Regular Expression

Let M be a DFA.

1. Creation of GNFA (Generalized NFA):
   - Add a new initial state s and a new final state f.
   - Put an ϵ-transition from s to the old initial state.
   - Put ϵ-transitions from all old final states to f.

Converting a DFA to a Regular Expression

Let M be a DFA.

1. Creation of GNFA (Generalized NFA):
    - Add a new initial state s and a new final state f.
    - Put an ϵ-transition from s to the old initial state.
    - Put ϵ-transitions from all old final states to f.

    - Replace transitions with multiple labels by $\cup$ regular expressions:

      $\bigcirc \xrightarrow{a,\, b} \bigcirc$ is replaced by $\bigcirc \xrightarrow{a \cup b} \bigcirc$

Converting a DFA to a Regular Expression

Let M be a DFA.

1. Creation of GNFA (Generalized NFA):
    - Add a new initial state s and a new final state f.
    - Put an ϵ-transition from s to the old initial state.
    - Put ϵ-transitions from all old final states to f.

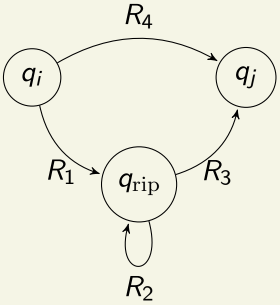
    - Replace transitions with multiple labels by $\cup$ regular expressions:

      $\bigcirc \xrightarrow{a, b} \bigcirc$ is replaced by $\bigcirc \xrightarrow{a \cup b} \bigcirc$

    - In a GNFA, all edges are labeled by regular expressions. Throughout the algorithm we will maintain a GNFA.

Converting a DFA to a Regular Expression

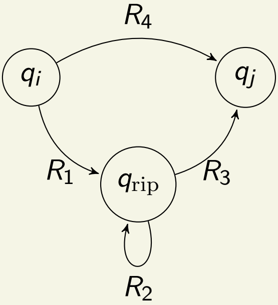
2. Select a state q_{rip} other than s or f to remove. For every pair of states q_i, q_j where there is a transition from q_i to q_{rip} and a transition from q_{rip} to q_j, do the following.



$\Longrightarrow$

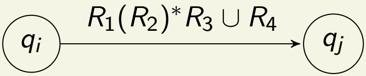
Converting a DFA to a Regular Expression

2. Select a state q_{rip} other than s or f to remove. For every pair of states q_i, q_j where there is a transition from q_i to q_{rip} and a transition from q_{rip} to q_j, do the following.



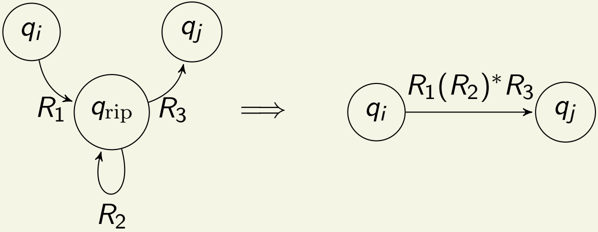
$\Longrightarrow$

New transition from q_i to q_j after q_{rip} is removed:



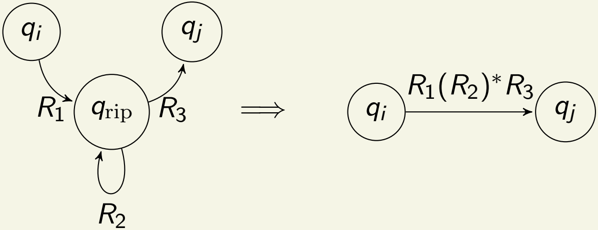
Special Cases of the Elimination Rule

No R_4:

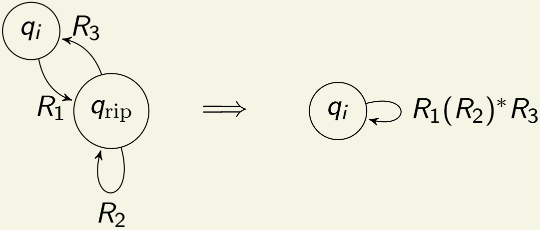


Special Cases of the Elimination Rule

No R_4:

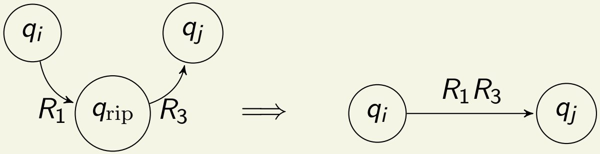


$q_i = q_j$, no R_4:



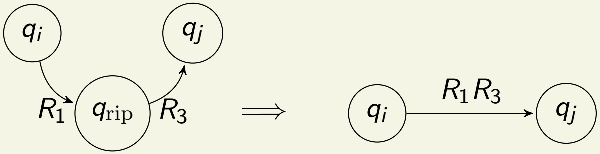
Special Cases of the Elimination Rule

No R_2 or R_4:

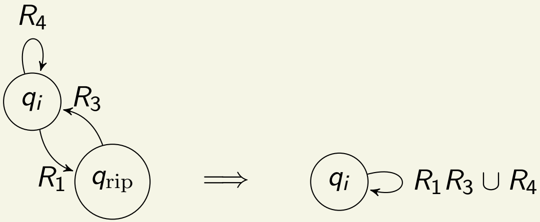


Special Cases of the Elimination Rule

No R_2 or R_4:



$q_i = q_j$, no R_2:



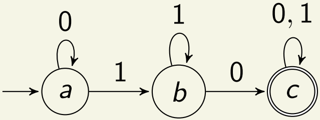
There are a few other special cases.

Converting a DFA to a Regular Expression

3. Repeat step 2 (removing states) until only s and f remain.
4. The regular expression on the transition from s to f is equivalent to the original DFA.

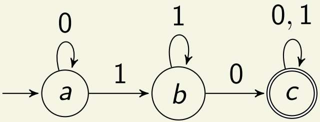
See book for proof of correctness.

Example

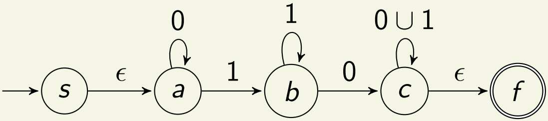
DFA: 

Example

DFA:

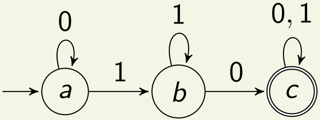


GNFA:

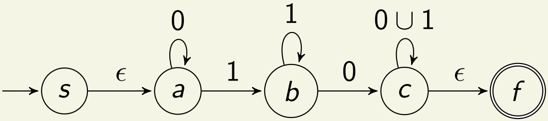


Example

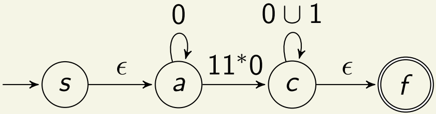
DFA:



GNFA:

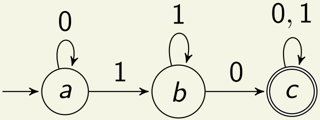


Remove b:

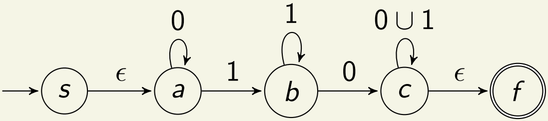


Example

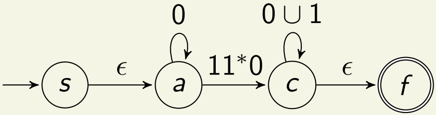
DFA:



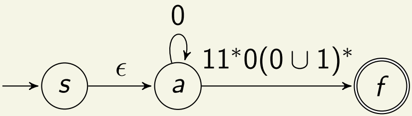
GNFA:



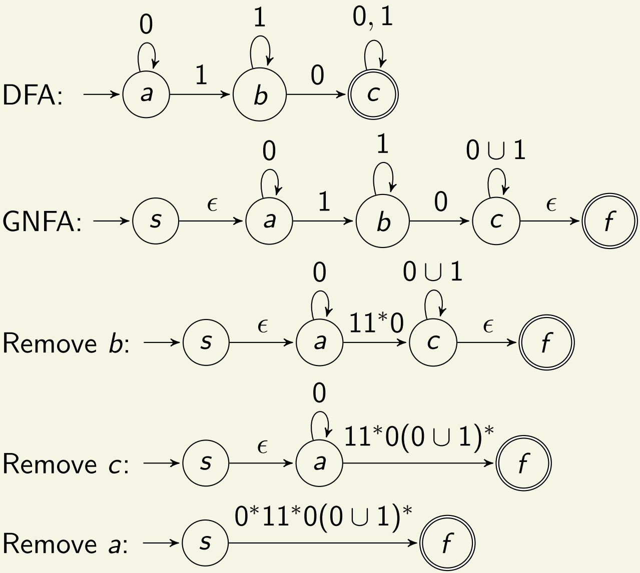
Remove b:



Remove c:

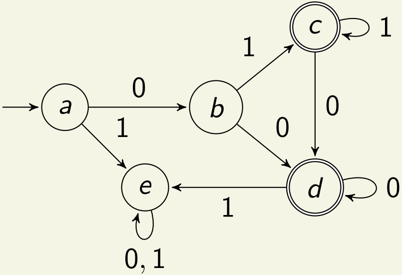


Example


DFA:
a
b
c
0
1
0
1
0, 1
GNFA:
s
a
b
c
f
ϵ
0
1
1
0
0 ∪ 1
ϵ
Remove b:
s
a
c
f
ϵ
0
11*0
0 ∪ 1
ϵ
Remove c:
s
a
f
ϵ
0
11*0(0 ∪ 1)*
Remove a:
s
f
0*11*0(0 ∪ 1)*

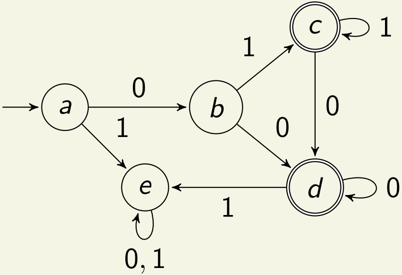
Example

DFA:

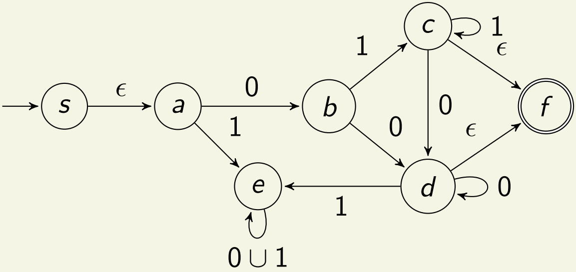


Example

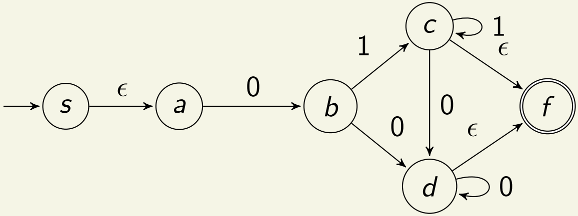
DFA:



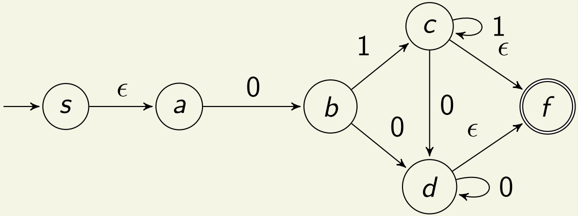
GNFA:



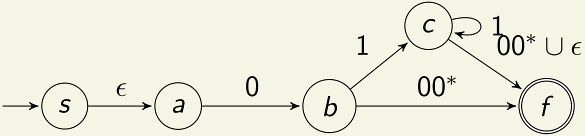
Remove e:



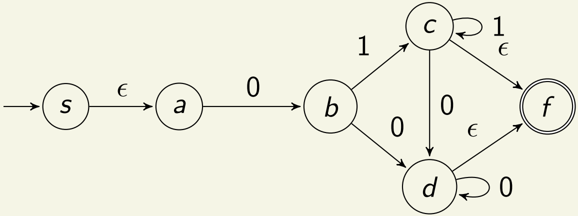
Remove e:



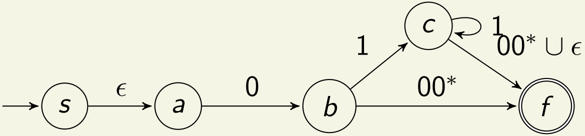
Remove d:



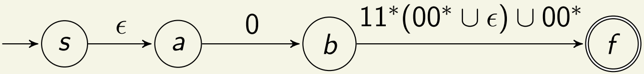
Remove e:



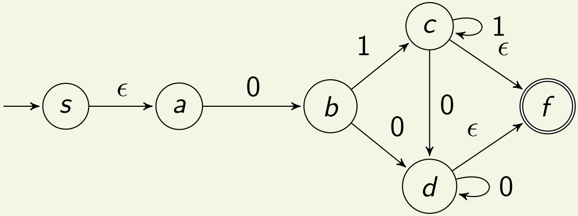
Remove d:



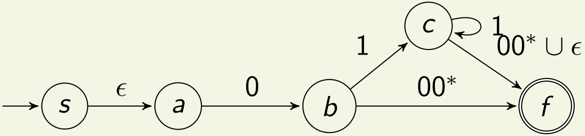
Remove c:



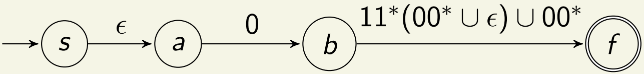
Remove e:



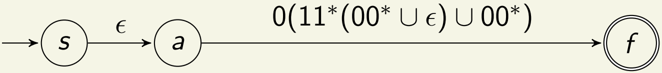
Remove d:



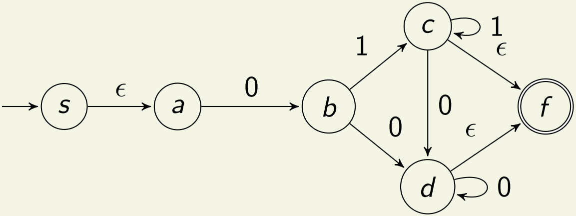
Remove c:



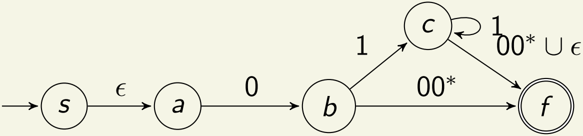
Remove b:



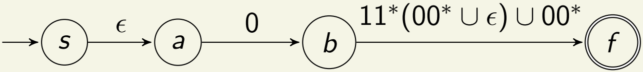
Remove e:



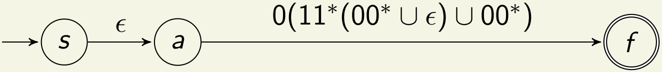
Remove d:



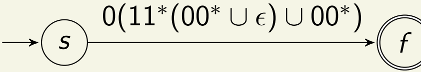
Remove c:



Remove b:



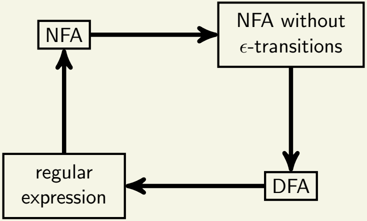
Remove a:



Theorem

A language is regular if and only if it can be described by a regular expression.

We have now completed the proof:



Equivalent Definitions of Regularity

Theorem

Let A be any language. The following are equivalent.

- *A is regular.*
- *$A = L(M)$ for some DFA M.*
- *$A = L(N)$ for some NFA N.*
- *$A = L(N')$ for some NFA N' without ϵ-transitions.*
- *$A = L(R)$ for some regular expression R.*