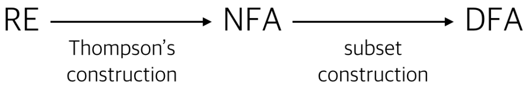
COSE312: Compilers

Lecture 5 — Lexical Analysis (4)

Hakjoo Oh
2017 Spring

Part 3: Automation

Transform the lexical specification into an executable string recognizers:



From NFA to DFA

Transform an NFA

$$(N, \Sigma, \delta_N, n_0, N_A)$$

into an equivalent DFA

$$(D, \Sigma, \delta_D, d_0, D_A).$$

From NFA to DFA

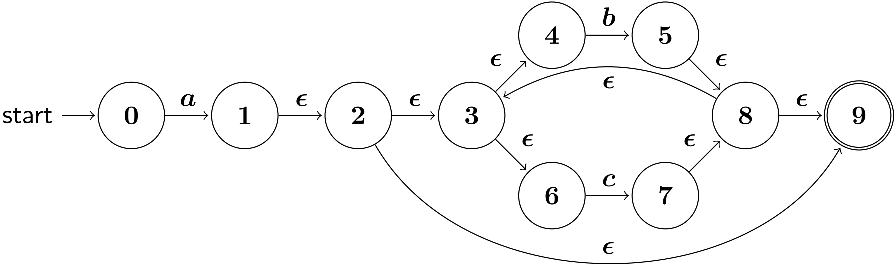
Transform an NFA

$$(N, \Sigma, \delta_N, n_0, N_A)$$

into an equivalent DFA

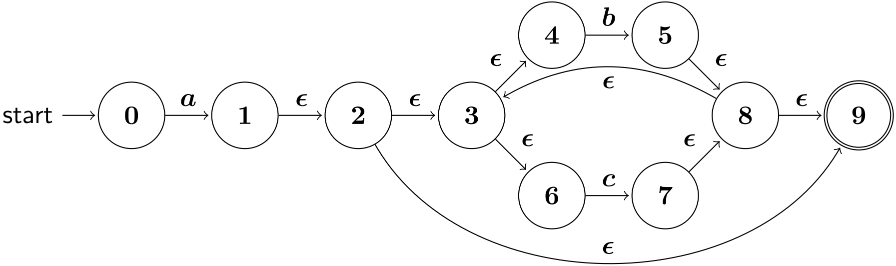
$$(D, \Sigma, \delta_D, d_0, D_A).$$

Running example:



ϵ-Closures

ϵ-**closure**(I): the set of states reachable from I without consuming any symbols.



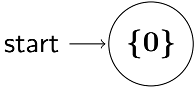
$$\epsilon\text{-closure}(\{1\}) = \{1, 2, 3, 4, 6, 9\}$$
$$\epsilon\text{-closure}(\{1, 5\}) = \{1, 2, 3, 4, 6, 9\} \cup \{3, 4, 5, 6, 8, 9\}$$

Subset Construction

- Input: an NFA $(N, \Sigma, \delta_N, n_0, N_A)$.
- Output: a DFA $(D, \Sigma, \delta_D, d_0, D_A)$.
- Key Idea: the DFA simulates the NFA by considering every possibility at once. A DFA state $d \in D$ is a set of NFA state, i.e., $d \subseteq N$.

Running Example (1/5)

The initial DFA state $d_0 = \epsilon$-**closure**$(\{0\}) = \{0\}$.



Running Example (2/5)

For the initial state S, consider every $x \in \Sigma$ and compute the corresponding next states:

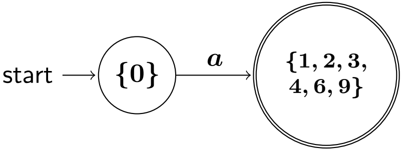
$$\epsilon\text{-}\mathbf{closure}(\bigcup_{s \in S} \delta(s, a)).$$

Running Example (2/5)

For the initial state S, consider every $x \in \Sigma$ and compute the corresponding next states:

$$\epsilon\text{-}\mathbf{closure}(\bigcup_{s \in S} \delta(s, a)).$$

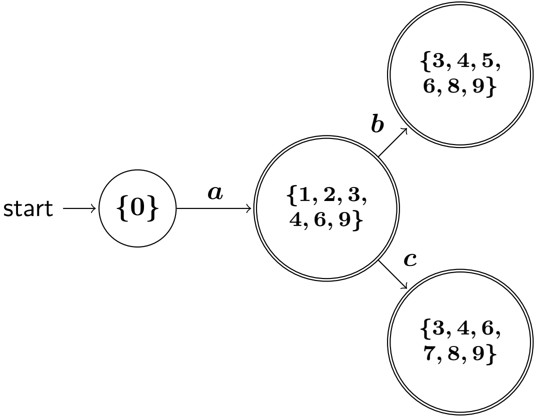
$$\begin{array}{rcl} \epsilon\text{-}\mathbf{closure}(\bigcup_{s \in \{0\}} \delta(s, a)) & = & \{1, 2, 3, 4, 6, 9\} \\ \epsilon\text{-}\mathbf{closure}(\bigcup_{s \in \{0\}} \delta(s, b)) & = & \emptyset \\ \epsilon\text{-}\mathbf{closure}(\bigcup_{s \in \{0\}} \delta(s, c)) & = & \emptyset \end{array}$$



Running Example (3/5)

For the state $\{1,2,3,4,6,9\}$, compute the next states:

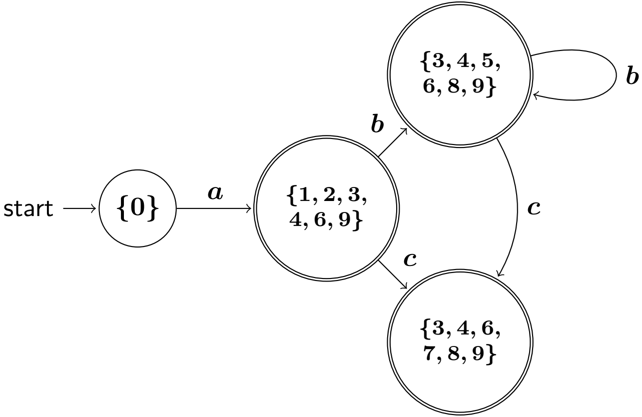
$$\epsilon\text{-closure}(\bigcup_{s\in\{1,2,3,4,6,9\}} \delta(s,a)) = \emptyset$$
$$\epsilon\text{-closure}(\bigcup_{s\in\{1,2,3,4,6,9\}} \delta(s,b)) = \{3,4,5,6,8,9\}$$
$$\epsilon\text{-closure}(\bigcup_{s\in\{1,2,3,4,6,9\}} \delta(s,c)) = \{3,4,6,7,8,9\}$$



Running Example (4/5)

Compute the next states of $\{3,4,5,6,8,9\}$:

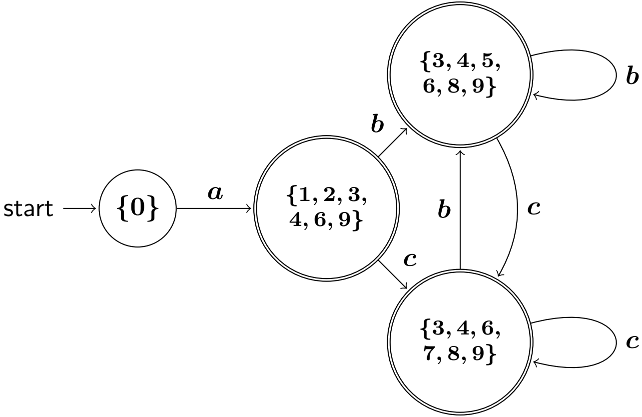
$$\epsilon\text{-closure}(\bigcup_{s\in\{3,4,5,6,8,9\}} \delta(s,a)) = \emptyset$$
$$\epsilon\text{-closure}(\bigcup_{s\in\{3,4,5,6,8,9\}} \delta(s,b)) = \{3,4,5,6,8,9\}$$
$$\epsilon\text{-closure}(\bigcup_{s\in\{3,4,5,6,8,9\}} \delta(s,c)) = \{3,4,6,7,8,9\}$$



Running Example (5/5)

Compute the next states of $\{3,4,6,7,8,9\}$:

$$\epsilon\text{-closure}(\bigcup_{s\in\{3,4,6,7,8,9\}} \delta(s,a)) = \emptyset$$
$$\epsilon\text{-closure}(\bigcup_{s\in\{3,4,6,7,8,9\}} \delta(s,b)) = \{3,4,5,6,8,9\}$$
$$\epsilon\text{-closure}(\bigcup_{s\in\{3,4,6,7,8,9\}} \delta(s,c)) = \{3,4,6,7,8,9\}$$



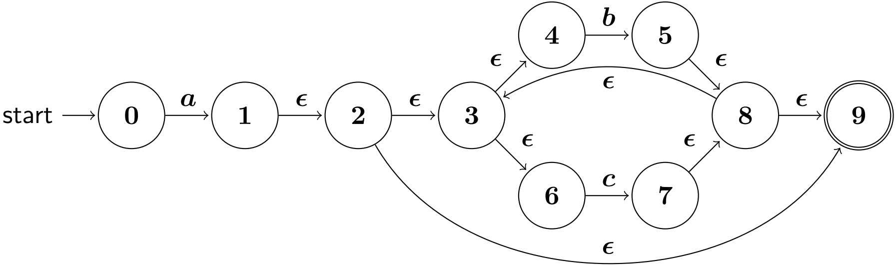
Subset Construction Algorithm

Algorithm 1 Subset construction

Input: An NFA $(N, \Sigma, \delta_N, n_0, N_A)$
Output: An equivalent DFA $(D, \Sigma, \delta_D, d_0, D_A)$

$d_0 = \epsilon\text{-closure}(\{n_0\})$
$D = \{d_0\}$
$W = \{d_0\}$
while $W \neq \emptyset$ **do**
  remove q from W
  for $c \in \Sigma$ **do**
    $t = \epsilon\text{-closure}(\bigcup_{s \in q} \delta(s, c))$
    $D = D \cup \{t\}$
    $\delta_D(q, c) = t$
    if t was newly added to D **then**
      $W = W \cup \{t\}$
    end if
  end for
end while
$D_A = \{q \in D \mid q \cap N_A \neq \emptyset\}$

Running Example (1/5)



The initial state $d_0 = \epsilon$-**closure**$(\{0\}) = \{0\}$. Initialize D and W:

$$D = \{\{0\}\}, \qquad W = \{\{0\}\}$$

Running Example (2/5)

Choose $q = \{0\}$ from W. For all $c \in \Sigma$, update δ_D:

	a	b	c
$\{0\}$	$\{1,2,3,4,6,9\}$	$\emptyset$	$\emptyset$

Update D and W:

$$D = \{\{0\}, \{1,2,3,4,6,9\}\}, \qquad W = \{\{1,2,3,4,6,9\}\}$$

Running Example (3/5)

Choose $q = \{1,2,3,4,6,9\}$ from W. For all $c \in \Sigma$, update δ_D:

	a	b	c
$\{0\}$	$\{1,2,3,4,6,9\}$	$\emptyset$	$\emptyset$
$\{1,2,3,4,6,9\}$	$\emptyset$	$\{3,4,5,6,8,9\}$	$\{3,4,6,7,8,9\}$

Update D and W:

$$\begin{aligned} D &= \{\{0\},\{1,2,3,4,6,9\},\{3,4,5,6,8,9\},\{3,4,6,7,8,9\}\} \\ W &= \{\{3,4,5,6,8,9\},\{3,4,6,7,8,9\}\} \end{aligned}$$

Running Example (4/5)

Choose $q = \{3, 4, 5, 6, 8, 9\}$ from W. For all $c \in \Sigma$, update δ_D:

	a	b	c
$\{0\}$	$\{1, 2, 3, 4, 6, 9\}$	$\emptyset$	$\emptyset$
$\{1, 2, 3, 4, 6, 9\}$	$\emptyset$	$\{3, 4, 5, 6, 8, 9\}$	$\{3, 4, 6, 7, 8, 9\}$
$\{3, 4, 5, 6, 8, 9\}$	$\emptyset$	$\{3, 4, 5, 6, 8, 9\}$	$\{3, 4, 6, 7, 8, 9\}$

D and W:

$$\begin{array}{rcl} D & = & \{\{0\}, \{1, 2, 3, 4, 6, 9\}, \{3, 4, 5, 6, 8, 9\}, \{3, 4, 6, 7, 8, 9\}\} \\ W & = & \{\{3, 4, 6, 7, 8, 9\}\} \end{array}$$

Running Example (5/5)

Choose $q = \{3, 4, 6, 7, 8, 9\}$ from W. For all $c \in \Sigma$, update δ_D:

	a	b	c
$\{0\}$	$\{1, 2, 3, 4, 6, 9\}$	$\emptyset$	$\emptyset$
$\{1, 2, 3, 4, 6, 9\}$	$\emptyset$	$\{3, 4, 5, 6, 8, 9\}$	$\{3, 4, 6, 7, 8, 9\}$
$\{3, 4, 5, 6, 8, 9\}$	$\emptyset$	$\{3, 4, 5, 6, 8, 9\}$	$\{3, 4, 6, 7, 8, 9\}$
$\{3, 4, 6, 7, 8, 9\}$	$\emptyset$	$\{3, 4, 5, 6, 8, 9\}$	$\{3, 4, 6, 7, 8, 9\}$

D and W:

$$\begin{aligned} D &= \{\{0\}, \{1, 2, 3, 4, 6, 9\}, \{3, 4, 5, 6, 8, 9\}, \{3, 4, 6, 7, 8, 9\}\} \\ W &= \emptyset \end{aligned}$$

The while loop terminates. The accepting states:

$$D_A = \{\{1, 2, 3, 4, 6, 9\}, \{3, 4, 5, 6, 8, 9\}, \{3, 4, 6, 7, 8, 9\}\}$$

Algorithm for computing ϵ-Closures

- The definition

  ϵ-**closure**(I) is the set of states reachable from I
  without consuming any symbols.

  is neither formal nor constructive.

Algorithm for computing ϵ-Closures

- The definition

  ϵ-**closure**(I) is the set of states reachable from I without consuming any symbols.

  is neither formal nor constructive.

- A formal definition:
  $T = \epsilon$-**closure**(I) is the smallest set such that

$$I \cup \bigcup_{s \in T} \delta(s, \epsilon) \subseteq T.$$

Algorithm for computing ϵ-Closures

- The definition

  ϵ-**closure**(I) is the set of states reachable from I without consuming any symbols.

  is neither formal nor constructive.
- A formal definition:
  $T = \epsilon$-**closure**(I) is the smallest set such that

$$I \cup \bigcup_{s \in T} \delta(s, \epsilon) \subseteq T.$$

- Alternatively, T is the smallest solution of the equation

$$F(X) \subseteq (X)$$

  where

$$F(X) = I \cup \bigcup_{s \in X} \delta(s, \epsilon).$$

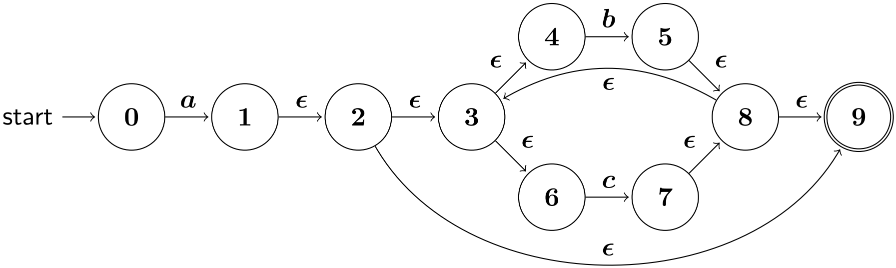
  Such a solution is called the least fixed point of F.

Fixed Point Iteration

The least fixed point of a function can be computed by the *fixed point iteration*:

$$
\begin{array}{l}
\boldsymbol{T = \emptyset} \\
\textbf{repeat} \\
\quad \boldsymbol{T' = T} \\
\quad \boldsymbol{T = T' \cup F(T')} \\
\textbf{until } \boldsymbol{T = T'}
\end{array}
$$

Example



ϵ-closure$(\{1\})$:

Iteration	T'	T
1	$\emptyset$	$\{1\}$
2	$\{1\}$	$\{1,2\}$
3	$\{1,2\}$	$\{1,2,3,9\}$
4	$\{1,2,3,9\}$	$\{1,2,3,4,6,9\}$
5	$\{1,2,3,4,6,9\}$	$\{1,2,3,4,6,9\}$

Summary

Key concepts in lexical analisis:

- Specification: Regular expressions
- Implementation: Deterministic Finite Automata
- Translation (homework 1)

Next class: OCaml programming tutorial by TAs.