epsilon-NFA: Formulation (2)

An ϵ-NFA is a 5-tuple

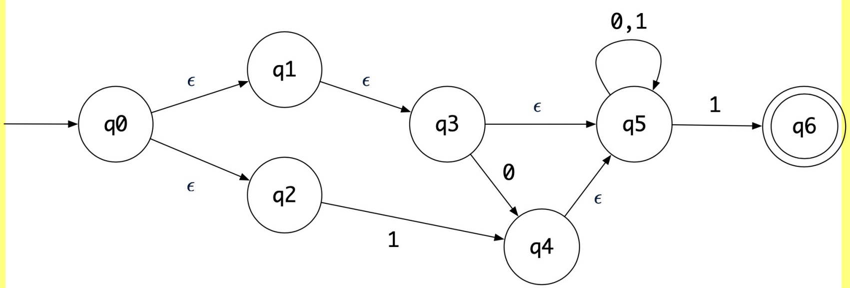
$$M = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$$

we define the *epsilon closure* (or *ϵ-closure*) as a function

$$\mathrm{ECLOSE} : Q \to \mathbb{P}(Q)$$

For any state $q \in Q$

$$\mathrm{ECLOSE}(q) = \{q\} \cup \bigcup_{p \in \delta(q,\epsilon)} \mathrm{ECLOSE}(p)$$



Derive ECLOSE(q0).

epsilon-NFA: Formulation (3)

An ϵ-*NFA* is a 5-tuple

$$M = (Q, \Sigma, \delta, q_0, F)$$

$$\hat{\delta} : (Q \times \Sigma^*) \rightarrow \mathbb{P}(Q)$$

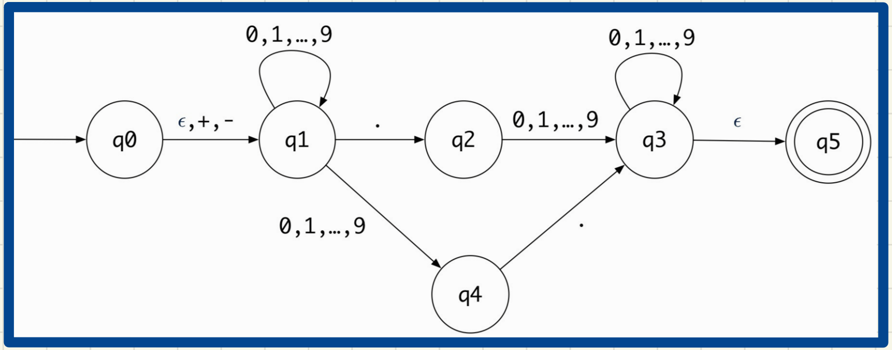
We may define $\hat{\delta}$ recursively, using δ!

$$\hat{\delta}(q, \epsilon) \quad =$$
$$\hat{\delta}(q, xa) \quad =$$

Language of a epsilon-NFA

$$L(M) = \{ \quad \}$$

epsilon-NFA: Processing Strings



How an epsilon-NFA determines if input 5.6 should be processed

$\hat{\delta}(q_0, \epsilon) =$

- **Read 5:**

$\hat{\delta}(q_0, 5) =$

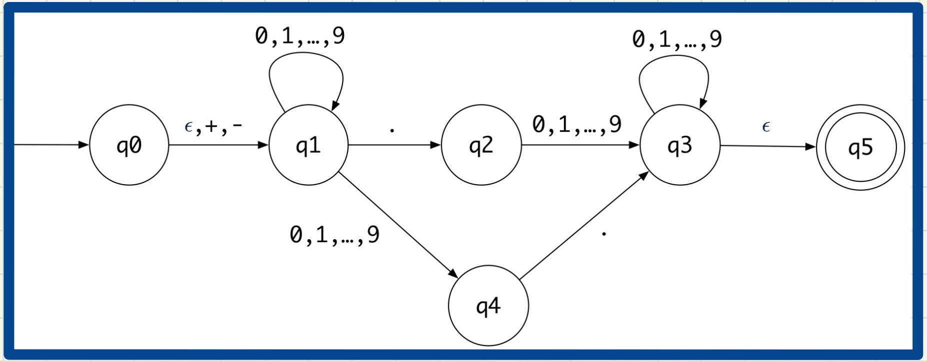
- **Read .:**

$\hat{\delta}(q_0, 5.) =$

- **Read 6:**

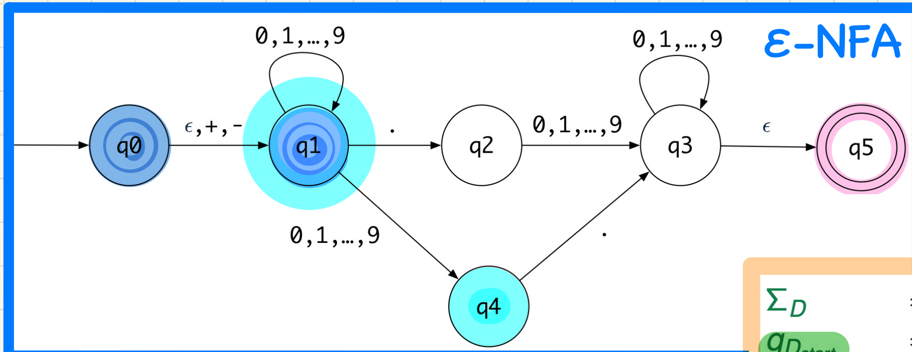
$\hat{\delta}(q_0, 5.6) =$

epsilon-NFA to DFA: Extended Subset Construction



	$d \in 0..9$	$s \in \{+,-\}$	.
		$\{q_1\}$	$\{q_2\}$
$\{q_1, q_4\}$	$\{q_1, q_4\}$	$\varnothing$	$\{q_2, q_3, q_5\}$
$\{q_1\}$	$\{q_1, q_4\}$	$\varnothing$	$\{q_2\}$
$\{q_2\}$	$\{q_3, q_5\}$	$\varnothing$	$\varnothing$
$\{q_2, q_3, q_5\}$	$\{q_3, q_5\}$	$\varnothing$	$\varnothing$
$\{q_3, q_5\}$	$\{q_3, q_5\}$	$\varnothing$	$\varnothing$

epsilon-NFA to DFA: Extended Subset Construction



$$
\begin{array}{lcl}
\Sigma_D & = & \Sigma_N \\
q_{D_{start}} & = & \mathrm{ECLOSE}(q_0) \\
F_D & = & \{\ S \mid S \subseteq Q_N \wedge S \cap F_N \neq \varnothing\ \} \\
Q_D & = & \{\ S \mid S \subseteq Q_N \wedge (\exists w \bullet w \in \Sigma^* \Rightarrow S = \hat{\delta}_N(q_0, w))\ \} \\
\delta_D(S, a) & = & \bigcup\{\ \mathrm{ECLOSE}(s') \mid s \in S \wedge s' \in \delta_N(s, a)\ \}
\end{array}
$$

	$d \in 0..9$	$s \in \{+, -\}$	.
$\{q_0, q_1\}$	$\{q_1, q_4\}$	$\{q_1\}$	$\{q_2\}$
$\{q_1, q_4\}$	$\{q_1, q_4\}$	$\varnothing$	$\{q_2, q_3, q_5\}$
$\{q_1\}$	$\{q_1, q_4\}$	$\varnothing$	$\{q_2\}$
$\{q_2\}$	$\{q_3, q_5\}$	$\varnothing$	$\varnothing$
$\{q_2, q_3, q_5\}$	$\{q_3, q_5\}$	$\varnothing$	$\varnothing$
$\{q_3, q_5\}$	$\{q_3, q_5\}$	$\varnothing$	$\varnothing$

DFA

Minimizing DFA: Algorithm

```
ALGORITHM: MinimizeDFAStates
  INPUT: DFA M = (Q, Σ, δ, q0, F)
  OUTPUT: M′ s.t. minimum |Q| and equivalent behaviour as M
PROCEDURE:
  P := ∅ /* refined partition so far */
  T := { F, Q − F } /* last refined partition */
  while (P ≠ T):
    P := T
    T := ∅
    for(p ∈ P):
      find the maximal S ⊂ p s.t. splittable(p, S)
      if S ≠ ∅ then
        T := T ∪ {S, p − S}
      else
        T := T ∪ {p}
      end
```

splittable(p, S) holds iff there is $c \in \Sigma$ s.t.

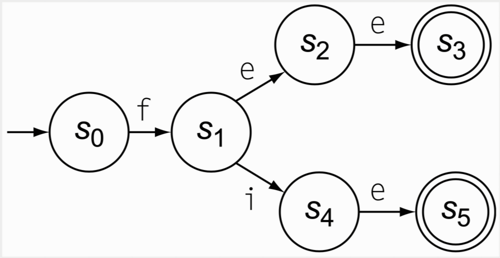
1. $S \subset p$ (or equivalently: $p - S \neq \varnothing$)
2. Transitions via c lead all $s \in S$ to states in ***same partition*** $p1$ $(p1 \neq p)$.

Partitions of States

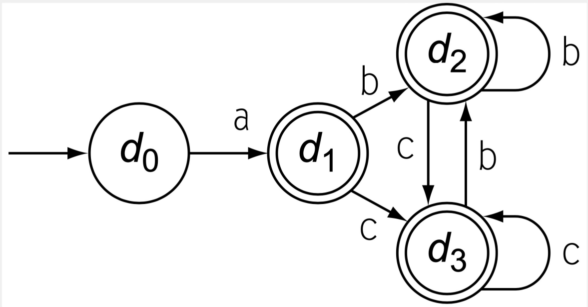
e.g., $Q = \{s0, s1, s2, s3\}$

- Smallest number of partitions
- Largest number of partitions
- Partitions somewhere in-between
- Analogy from Software Testing: Equivalent Classes

Minimizing DFA: Example (1)



Minimizing DFA: Example (2)



Minimizing DFA: Example (3)