COMP2012/G52LAC Languages and Computation Lecture 4

Equivalence between NFA and DFA

Henrik Nilsson

University of Nottingham

Recap: Formal Definition of NFA (1)

Formally, a ***Nondeterministic Finite Automaton*** or ***NFA*** is defined by a 5-tuple

$$(Q, \Sigma, \delta, S, F)$$

where

Q	:	Finite set of States
Σ	:	Alphabet (finite set of symbols)
$\delta \in Q \times \Sigma \to \mathcal{P}(Q)$	:	Transition Function
$S \subseteq Q$	:	Initial States
$F \subseteq Q$	:	Accepting (or Final) States

Recap: Formal Definition of NFA (2)

Note:

Recap: Formal Definition of NFA (2)

Note:

- The transition function maps a state and an input symbol to ***zero or more*** successor states. Thus an NFA has "choice"; hence "nondeterministic".

Recap: Formal Definition of NFA (2)

Note:

- The transition function maps a state and an input symbol to ***zero or more*** successor states. Thus an NFA has “choice”; hence “nondeterministic”.
- However, nothing ambiguous about the ***language*** defined by an NFA! ***Not*** the case that some word $w \in L(A)$ sometimes, and $w \notin L(A)$ other times for some NFA A.

Recap: Formal Definition of NFA (2)

Note:

- The transition function maps a state and an input symbol to ***zero or more*** successor states. Thus an NFA has “choice”; hence “nondeterministic”.
- However, nothing ambiguous about the ***language*** defined by an NFA! ***Not*** the case that some word $w \in L(A)$ sometimes, and $w \notin L(A)$ other times for some NFA A.
- How? By considering ***all possible*** states simultaneously.

Recap: Extended Transition Function

For an NFA, The ***Extended Transition Function*** is defined on a ***set*** of states and a ***word*** (string of symbols).

For a NFA $A = (Q, \Sigma, \delta, S, F)$, the extended transition function is defined by:

$$
\begin{aligned}
\hat{\delta} &\in \mathcal{P}(Q) \times \Sigma^* \to \mathcal{P}(Q) \\
\hat{\delta}(P, \epsilon) &= P \\
\hat{\delta}(P, xw) &= \hat{\delta}(\bigcup\{\delta(q, x) \mid q \in P\}, w)
\end{aligned}
$$

where $P \in \mathcal{P}(Q)$ (or $P \subseteq Q$), $x \in \Sigma$, $w \in \Sigma^*$.

Recap: Language of an NFA

The ***language*** $L(A)$ defined by an NFA A is the set or words ***accepted*** by the NFA. For an NFA

$$A = (Q, \Sigma, \delta, S, F)$$

the language is defined by

$$L(A) = \{ \, w \in \Sigma^* \mid \hat{\delta}(S, w) \cap F \neq \emptyset \, \}$$

The Subset Construction (1)

Observations:

The Subset Construction (1)

Observations:

- An NFA can be in one of a ***set*** of states.

The Subset Construction (1)

Observations:

- An NFA can be in one of a ***set*** of states.
- When reading an input symbol, the machine enters one of a ***new set*** of states.

The Subset Construction (1)

Observations:

- An NFA can be in one of a ***set*** of states.
- When reading an input symbol, the machine enters one of a ***new set*** of states.
- Which are the ***sets*** of possible states?

The Subset Construction (1)

Observations:

- An NFA can be in one of a ***set*** of states.
- When reading an input symbol, the machine enters one of a ***new set*** of states.
- Which are the ***sets*** of possible states?
- Each set is a subset of Q, so the set of possible states is (at most) $\mathcal{P}(Q)$.

The Subset Construction (1)

Observations:

- An NFA can be in one of a ***set*** of states.
- When reading an input symbol, the machine enters one of a ***new set*** of states.
- Which are the ***sets*** of possible states?
- Each set is a subset of Q, so the set of possible states is (at most) $\mathcal{P}(Q)$.
- But Q is finite. Thus $\mathcal{P}(Q)$ is ***finite*** too!

The Subset Construction (1)

Observations:

- An NFA can be in one of a ***set*** of states.
- When reading an input symbol, the machine enters one of a ***new set*** of states.
- Which are the ***sets*** of possible states?
- Each set is a subset of Q, so the set of possible states is (at most) $\mathcal{P}(Q)$.
- But Q is finite. Thus $\mathcal{P}(Q)$ is ***finite*** too!
- There may be ***lots*** of states as $|\mathcal{P}(Q)| = 2^{|Q|}$. But the number of states is finite!

The Subset Construction (2)

- We can thus ***convert*** an NFA into a DFA by considering each possible set of NFA states as a single DFA state!

The Subset Construction (3)

Given an NFA A:

$$A = (Q, \Sigma, \delta, S, F)$$

we construct the ***equivalent*** DFA $D(A)$ as:

$$D(A) = (\mathcal{P}(Q), \Sigma, \delta_{D(A)}, S, F_{D(A)})$$

where

$$\begin{aligned} \delta_{D(A)}(P, x) &= \bigcup \{\delta(q, x) \mid q \in P\} \\ F_{D(A)} &= \{P \in \mathcal{P}(Q) \mid P \cap F \neq \emptyset\} \end{aligned}$$

The Subset Construction (3)

Given an NFA A:

$$A = (Q, \Sigma, \delta, S, F)$$

we construct the ***equivalent*** DFA $D(A)$ as:

$$D(A) = (\mathcal{P}(Q), \Sigma, \delta_{D(A)}, S, F_{D(A)})$$

where

$$\begin{aligned} \delta_{D(A)}(P, x) &= \bigcup \{\delta(q, x) \mid q \in P\} \\ F_{D(A)} &= \{P \in \mathcal{P}(Q) \mid P \cap F \neq \emptyset\} \end{aligned}$$

(Cf. def. $\hat{\delta}$ and language for NFA!)