COMP2012/G52LAC
Languages and Computation
Lecture 3

Non-deterministic Finite Automata (NFA)

Henrik Nilsson

University of Nottingham

Recap: Formal Definition of DFA

Formally, a ***Deterministic Finite Automaton*** or ***DFA*** is defined by a 5-tuple

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

where

Q	:	***Finite*** set of States
Σ	:	Alphabet (finite set of symbols)
$\delta \in Q \times \Sigma \to Q$	:	Transition Function
$q_0 \in Q$	:	Initial or Start State
$F \subseteq Q$	:	Accepting (or Final) States

Recap: Extended Transition Function

The ***Extended Transition Function*** is defined on a state and a ***word*** (string of symbols) instead of on a single symbol.

For a DFA $A = (Q, \Sigma, \delta, q_0, F)$, the extended transition function is defined by:

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

where $q \in Q$, $x \in \Sigma$, $w \in \Sigma^*$.

Recap: Language of a DFA

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

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

the language is defined by

$$L(A) = \{ \, w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F \, \}$$

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

Formal Definition of NFA (2)

Note:

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”.

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.

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.

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 $P \subseteq Q$), $x \in \Sigma$, $w \in \Sigma^*$.

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

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