Minimization? What and Why?

Q: Is there a unique smallest DFA for recognizing a particular regular language?
A: Yes! (Up to renaming of states.)
Moreover, this minimal DFA can be found mechanically.

Why useful?
- Small improves efficiency if we want to implement a DFA.
- Unique means it is easy to check if two automata really are the same.

Applications

Applying all we know after this lecture, we can for example:
- Given a regular expression $E$, construct the smallest possible DFA for recognizing the language:
  \[
  \text{minimize}(D(N(E)))
  \]
  This is in essence what tools like Lex and Flex do generate efficient scanners from declarative specifications stated in terms of regular expressions.

Testing Equivalence of States

For DFA $(Q, \Sigma, \delta, q_0, F)$, states $p, q \in Q$ are equivalent if for all $w \in \Sigma^*$, $\hat{\delta}(p, w) \in F \iff \hat{\delta}(q, w) \in F$.

If two states are not equivalent, then they are distinguishable on at least one word $w$.

Note that an accepting state is always distinguishable from a non-accepting state on the empty word $\epsilon$. To see this, assume $p \in F, q \notin F$. Then:
\[
\begin{align*}
\hat{\delta}(p, \epsilon) &= p \in F \\
\hat{\delta}(q, \epsilon) &= q \notin F
\end{align*}
\]

The Table-filling Algorithm

INDUCTION:
For $p, q, r, s \in Q, a \in \Sigma$, if
\[
(r, s) = (\hat{\delta}(p, a), \hat{\delta}(q, a))
\]
a distinguishable state pair, then $(p, q)$ is also a distinguishable state pair.

Theorem: If two states are not distinguishable by the table-filling algorithm, then they are equivalent.