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 (1)

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.

Applications (2)

• Given two regular expressions \( E \) and \( F \), check if they denote the same language:

\[
\text{minimize}(D(N(E))) = \text{minimize}(D(N(F)))
\]

where \( = \) is a structural comparison of DFAs.

Not the only or necessarily the best way, but it is one way.
Testing Equivalence of States

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

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:

\[
\hat{\delta}(p, \epsilon) = p \in F \\
\hat{\delta}(q, \epsilon) = q \notin F
\]

The Table-filling Algorithm (1)

**Systematic discovery of distinguishable state pairs for DFA \((Q, \Sigma, \delta, q_0, F)\):**

**BASIS:**
For \(p, q \in Q\), if

\[(p \in F \land q \notin F) \lor (p \notin F \land q \in F)\]

then \((p, q)\) is a distinguishable state pair.

The Table-filling Algorithm (2)

**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**.