How can we prove that two regular expressions are equivalent?

How can we prove that two DFAs (or two NFAs) are equivalent?

How can we prove that two regular languages are equivalent?
(Does this question make sense?)

How can we prove that two DFAs (or two NFAs) are equivalent?

IS THIS MINIMAL?

MINIMIZING DFAs
THURSDAY Jan 23
IS THIS MINIMAL?

---

THEOREM

For every regular language $L$, there exists a UNIQUE (up to re-labeling of the states) minimal DFA $M$ such that $L = L(M)$.

Minimal means wrt number of states.

Given a specification for $L$, via DFA, NFA or regex, this theorem is constructive.

---

NOT TRUE FOR NFAs

---

EXTENDING $\delta$

Given DFA $M = (Q, \Sigma, \delta, q_0, F)$ extend $\delta$ to $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$ as follows:

- $\hat{\delta}(q, \varepsilon) = q$
- $\hat{\delta}(q, \sigma) = \delta(q, \sigma)$
- $\hat{\delta}(q, \sigma_1 \ldots \sigma_{k+1}) = \delta(\hat{\delta}(q, \sigma_1 \ldots \sigma_k), \sigma_{k+1})$

Note: $\hat{\delta}(q_0, w) \in F \iff M$ accepts $w$

String $w \in \Sigma^*$ distinguishes states $p$ and $q$ iff $\hat{\delta}(p, w) \in F \iff \hat{\delta}(q, w) \not\in F$

---

EXTENDING $\delta$

Given DFA $M = (Q, \Sigma, \delta, q_0, F)$ extend $\delta$ to $\hat{\delta} : Q \to Q$ as follows:

- $\hat{\delta}(q, \varepsilon) = q$
- $\hat{\delta}(q, \sigma) = \delta(q, \sigma)$
- $\hat{\delta}(q, \sigma_1 \ldots \sigma_{k+1}) = \delta(\hat{\delta}(q, \sigma_1 \ldots \sigma_k), \sigma_{k+1})$

Note: $\hat{\delta}(q_0, w) \in F \iff M$ accepts $w$

String $w \in \Sigma^*$ distinguishes states $p$ and $q$ iff exactly ONE of $\hat{\delta}(p, w), \hat{\delta}(q, w)$ is a final state.

---

Fix $M = (Q, \Sigma, \delta, q_0, F)$ and let $p, q \in Q$

DEFINITION:

$p$ is **distinguishable** from $q$ iff there is a $w \in \Sigma^*$ that distinguishes $p$ and $q$.

$p$ is **indistinguishable** from $q$ iff $p$ is not distinguishable from $q$.

For all $w \in \Sigma^*$, $\hat{\delta}(p, w) \in F \iff \hat{\delta}(q, w) \in F$.
Fix $M = (Q, \Sigma, \delta, q_0, F)$ and let $p, q, r \in Q$

Define relation $\sim$:

- $p \sim q$ iff $p$ is indistinguishable from $q$
- $p \not\sim q$ iff $p$ is distinguishable from $q$

Proposition: $\sim$ is an equivalence relation

- $p \sim p$ (reflexive)
- $p \sim q \Rightarrow q \sim p$ (symmetric)
- $p \sim q$ and $q \sim r \Rightarrow p \sim r$ (transitive)

Proof (of transitivity): for all $w$, we have:

$\delta(p, w) \in F \iff \delta(q, w) \in F \iff \delta(r, w) \in F$

Algorithm MINIMIZE

Input: DFA $M$

Output: DFA $M_{\text{MIN}}$ such that:

- $M = M_{\text{MIN}}$ (that is, $L(M) = L(M_{\text{MIN}})$)
- $M_{\text{MIN}}$ has no inaccessible states
- $M_{\text{MIN}}$ is irreducible
- all states of $M_{\text{MIN}}$ are pairwise distinguishable

Theorem: $M_{\text{MIN}}$ is the unique minimum DFA equivalent to $M$

Intuition: States of $M_{\text{MIN}}$ will be blocks of equivalent states of $M$

We'll find these equivalent states with a “Table-Filling” Algorithm
TABLE-FILLING ALGORITHM
Input: DFA \( M = (Q, \Sigma, \delta, q_0, F) \)
Output: (1) \( D_M = \{ (p,q) \mid p,q \in Q \text{ and } p \sim q \} \)
(2) \( E_M = \{ [q] \mid q \in Q \} \)

IDEA:
• We know how to find those pairs of states that \( \varepsilon \) distinguishes...
• Use this and recursion to find those pairs distinguishable with longer strings
• Pairs left over will be indistinguishable

Claim: If \( p, q \) are distinguished by Table-Filling algorithm (ie pair labelled by D), then \( p \neq q \)

Proof: By induction on the stage of the algorithm

If \( (p, q) \) is marked D at the start, then one’s in F and one isn’t, so \( \varepsilon \) distinguishes \( p \) and \( q \)

Suppose \( (p, q) \) is marked D at stage \( n+1 \)
Then there are states \( p', q' \), string \( w \in \Sigma^* \) and \( \sigma \in \Sigma \) such that:
1. \( (p', q') \) are marked D \( \Rightarrow p' \neq q' \) (by induction)
   \( \Rightarrow \delta(p', w) \in F \text{ and } \delta(q', w) \notin F \)

2. \( p' = \delta(p, \sigma) \) and \( q' = \delta(q, \sigma) \)
   The string \( \sigma w \) distinguishes \( p \) and \( q \)!
Claim: If \( p, q \) are not distinguished by Table-Filling algorithm, then \( p \sim q \)

Proof (by contradiction):
Suppose the pair \((p, q)\) is not marked D by the algorithm, yet \( p \neq q \) (a "bad pair")
Suppose \((p, q)\) is a bad pair with the shortest \( w \).
\[
\delta(p, w) \in F \text{ and } \delta(q, w) \notin F \quad \text{(Why is } |w| > 0?)
\]
So, \( w = \sigma w' \), where \( \sigma \in \Sigma \)
Let \( p' = \delta(p, \sigma) \) and \( q' = \delta(q, \sigma) \)
Then \((p', q')\) cannot be marked D (Why?)
But \((p', q')\) is distinguished by \( w' \)!
So \((p', q')\) is also a bad pair, but with a SHORTER \( w' \)!
Contradiction!

Algorithm MINIMIZE

Input: DFA \( M \)
Output: DFA \( M_{\text{MIN}} \)

(1) Remove all inaccessible states from \( M \)
(2) Apply Table-Filling algorithm to get:
\[
E_M = \{ [q] | q \text{ is an accessible state of } M \}
\]
Define: \( M_{\text{MIN}} = (Q_{\text{MIN}}, \Sigma, \delta_{\text{MIN}}, q_{0\text{MIN}}, F_{\text{MIN}}) \)
\[
Q_{\text{MIN}} = E_M, \quad q_{0\text{MIN}} = [q_0], \quad F_{\text{MIN}} = \{ [q] | q \in F \}
\]
\[
\delta_{\text{MIN}}([q], \sigma) = \{ \delta(q, \sigma) \}
\]
Claim: \( \delta_{\text{MIN}}([q], w) = \{ \delta(q, w) \}, w \in \Sigma^* \)

Follows: \( M_{\text{MIN}} \equiv M \)
**PROPOSITION.** Suppose \( M' \equiv M \) and \( M' \) has no inaccessible states and is irreducible

Then, there exists a 1-1 onto correspondence between \( M_{\text{MIN}} \) and \( M' \) (preserving transitions)

---

**COR:** \( M_{\text{MIN}} \) is unique minimal DFA \( \equiv M \)

---

**Proof of Prop:** We will construct a map recursively

**Base Case:** \( q_0 \rightarrow q_0' \)

**Recursive Step:** If \( p \rightarrow p' \) \( \sigma \)

Then \( q \rightarrow q' \)

---

**Base Case:** \( q_0 \rightarrow q_0' \)

**Recursive Step:** If \( p \rightarrow p' \)

Then \( q \rightarrow q' \)

The map is everywhere defined:

That is, for all \( q \in M_{\text{MIN}} \), there is a \( q' \in M' \) such that \( q \rightarrow q' \)

If \( q \in M_{\text{MIN}} \), there is a string \( w \) such that \( \delta_{\text{MIN}}(q_0, w) = q \) (WHY?)

Let \( q' = \delta'(q_0', w) \). \( q \) will map to \( q' \) (by induction)

---

**Base Case:** \( q_0 \rightarrow q_0' \)

**Recursive Step:** If \( p \rightarrow p' \)

Then \( q \rightarrow q' \)

The map is well defined

That is, for all \( q \in M_{\text{MIN}} \), there is at most one \( q' \in M' \) such that \( q \rightarrow q' \)

Suppose there exist \( q' \) and \( q'' \) such that \( q \rightarrow q' \) and \( q \rightarrow q'' \)

We show that \( q' \) and \( q'' \) are indistinguishable, so it must be that \( q' = q'' \) (Why?)
Suppose there exist $q'$ and $q''$ such that $q \rightarrow q'$ and $q \rightarrow q''$

Suppose $q'$ and $q''$ are distinguishable

Base Case: $q_{0 \text{MIN}} \rightarrow q_0'$
Recursive Step: If $p \rightarrow p'$
\[ \sigma \quad \sigma \]
Then $q \rightarrow q'$

The map is onto
That is, for all $q' \in M'$ there is a $q \in M_{\text{MIN}}$
such that $q \rightarrow q'$

If $q' \in M'$, there is $w$ such that
\[ \delta'(q_0', w) = q' \]

Let $q = \hat{\delta}_{\text{MIN}}(q_{0 \text{MIN}}, w)$. $q$ will map to $q'$ (why?)

How can we prove that two regular expressions are equivalent?

The map is 1-1
Suppose there are distinct $p$ and $q$ such that $p \rightarrow q'$ and $q \rightarrow q'$
$p$ and $q$ are distinguishable (why?)

Base Case: $q_{0 \text{MIN}} \rightarrow q_0'$
Recursive Step: If $p \rightarrow p'$
\[ \sigma \quad \sigma \]
Then $q \rightarrow q'$

The map preserves transitions
That is, if $\delta(p, \sigma) = q$ and $p \rightarrow p'$ and $q \rightarrow q'$
then, $\delta'(p', \sigma) = q'$
(Why?)

How can we prove that two regular expressions are equivalent?

www.flac.ws
Read Chapters 2.1 & 2.2 for next time