Lecture 5:
Minimizing DFAs
Announcements:
- Pset 2 is up (as of last night)
  - Dylan says: “It’s fire.”
- How was Pset 1?
DFAs ↔ NFAs

DEFINITION

Regular Languages ↔ Regular Expressions
REPRESENTATION OF EVENTS IN NERVE NETS AND FINITE AUTOMATA

S. C. Kleene

RM-704

15 December 1951
Some Languages Are Not Regular:

Limitations on DFAs/NFAs

a.k.a.

“Lower Bounds” on DFAs/NFAs
Minimizing DFAs
Does this DFA have a minimal number of states?

NO
Is this minimal?

How can we tell in general?
DFA Minimization Theorem:

For every regular language $A$, there is a unique (up to re-labeling of the states) minimal-state DFA $M^*$ such that $A = L(M^*)$.

Furthermore, there is an efficient algorithm which, given any DFA $M$, will output this unique $M^*$.

If such algorithms existed for more general models of computation, that would be an engineering breakthrough!!
In general, there isn’t a uniquely minimal NFA
Distinguishing states with strings

For a DFA $M = (Q, \Sigma, \delta, q_0, F)$, and $q \in Q$, let $M_q$ be the DFA equal to $(Q, \Sigma, \delta, q, F)$

**Def.** $w \in \Sigma^*$ *distinguishes* states $p$ and $q$ if:

$M_p$ accepts $w \iff M_q$ rejects $w$
Distinguishing states with strings

For a DFA $M = (Q, \Sigma, \delta, q_0, F)$, and $q \in Q$, let $M_q$ be the DFA equal to $(Q, \Sigma, \delta, q, F)$

**Def.** $w \in \Sigma^*$ *distinguishes* states $p$ and $q$ if: $M_p$ and $M_q$ have *different outputs* on input $w$

---

**Diagram:**

- For $M_p$:
  - Input $w$: $p \xrightarrow{w} \text{accept}$
  - Input $w$: $q \xrightarrow{w} \text{reject}$

- For $M_q$:
  - Input $w$: $p \xrightarrow{w} \text{reject}$
  - Input $w$: $q \xrightarrow{w} \text{accept}$

**OR**

- $M_p$: $\text{accept}$
- $M_q$: $\text{reject}$
Distinguishing two states

Def. \( w \in \Sigma^* \) *distinguishes* states \( q_1 \) and \( q_2 \) iff \( M_p \) and \( M_q \) have *different outputs* on \( w \)

I’m in \( q_1 \) or \( q_2 \), but which? How can I tell?
Distinguishing two states

Def. \( w \in \Sigma^* \) *distinguishes* states \( p \) and \( q \) iff
\( M_p \) and \( M_q \) have *different outputs* on \( w \)

Here... read this

Ok, I’m *accepting*!
Must have been \( p \)

Ok, I’m *rejecting*!
Must have been \( q \)
Fix $M = (Q, \Sigma, \delta, q_0, F)$ and let $p, q \in Q$

Let $M_p = (Q, \Sigma, \delta, p, F)$ and $M_q = (Q, \Sigma, \delta, q, F)$

Definition(s):

State $p$ is *distinguishable* from state $q$

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

iff there is a $w \in \Sigma^*$ so that

$M_p$ accepts $w \iff M_q$ rejects $w$

State $p$ is *indistinguishable* from state $q$

iff $p$ is not distinguishable from $q$

iff for all $w \in \Sigma^*$, $M_p$ accepts $w \iff M_q$ accepts $w$

*Big Idea*: Pairs of indistinguishable states are redundant!
Which pairs of states are distinguishable?

Are \( q_0 \) and \( q_1 \) distinguishable?

The empty string \( \varepsilon \) distinguishes all final states from all non-final states.
The string 10 distinguishes $q_0$ and $q_3$

Are $q_0$ and $q_3$ distinguishable?
Are $q_1$ and $q_2$ distinguishable?

The string 0 distinguishes $q_1$ and $q_2$

In this DFA, all pairs of states are distinguishable!
Fix $M = (Q, \Sigma, \delta, q_0, F)$ and let $p, q, r \in Q$

Define a binary relation $\sim$ on the states of $M$: 

$p \sim q$ 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? Just look at the definition! $p \sim q$ means for all $w$, $M_p$ accepts $w \iff M_q$ accepts $w$
Fix $M = (Q, \Sigma, \delta, q_0, F)$ and let $p, q, r \in Q$.

Therefore, the relation $\sim$ partitions $Q$ into disjoint equivalence classes.

**Proposition:** $\sim$ is an equivalence relation.

$$[q] := \{ p \mid p \sim q \}$$
Algorithm: MINIMIZE-DFA

Input: DFA M

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

1. $L(M) = L(M_{\text{MIN}})$  
   \[\iff\]

2. $M_{\text{MIN}}$ has no inaccessible states

3. $M_{\text{MIN}}$ is irreducible

   \[\iff\]

   for all states $p \neq q$ of $M_{\text{MIN}}$, $p$ and $q$ are distinguishable

Theorem: Every $M_{\text{MIN}}$ satisfying 1,2,3 is the unique minimal DFA equivalent to $M$
Intuition:
States of $M_{\text{MIN}} = \text{Equivalence classes of states of } M$

We’ll uncover these equivalent states with a dynamic programming algorithm
The 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. $\text{EQUIV}_M = \{ [q] \mid q \in Q \}$

Idea:

- We know how to find those pairs of states that the string $\varepsilon$ distinguishes...
- Use this and \textit{iteration} to find those pairs distinguishable with longer strings
- The pairs of states left over will be indistinguishable
The 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. $\text{EQUIV}_M = \{[q] \mid q \in Q\}$

Diagram:

- $q_0$
- $q_1$
- $q_i$
- $q_n$

- $q_0$
- $q_1$
- $q_i$
- $q_n$
The Table-Filling Algorithm

Input: DFA $M = (Q, \Sigma, \delta, q_0, F)$

Output:  
1. $D_M = \{ (p, q) \mid p, q \in Q \text{ and } p \not\sim q \}$
2. $\text{EQUIV}_M = \{ [q] \mid q \in Q \}$

Base Case: For all $(p, q)$ such that $p$ accepts and $q$ rejects $\Rightarrow$ mark $p \not\sim q$
The 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) $\text{EQUIV}_M = \{ [q] \mid q \in Q \}$

Base Case: For all $(p, q)$ such that $p$ accepts and $q$ rejects $\Rightarrow$ mark $p \sim q$

Iterate: If there are states $p, q$ and symbol $\sigma \in \Sigma$ satisfying:

\[ \delta(p, \sigma) = p' \quad \text{mark} \]
\[ \sim \Rightarrow p \sim q \]
\[ \delta(q, \sigma) = q' \]

Repeat until no more D’s can be added
Are $q_1$ and $q_2$ distinguishable?

Are $q_0$ and $q_1$ distinguishable?

Are $q_0$ and $q_2$ distinguishable?
Claim: If \((p, q)\) is marked D by the Table-Filling algorithm, then \(p \sim q\)

Proof: Induction on the number of iterations \(n\) in the algorithm when \((p, q)\) is marked D

\(n = 0\): If \((p, q)\) is marked D in the base case, then exactly one of them is final, so \(\varepsilon\) distinguishes \(p\) and \(q\)

I.H. For all \((p', q')\) marked D in the first \(n\) iterations, \(p' \sim q'\)

Suppose \((p, q)\) is marked D in \((n + 1)\)th iteration.

To be marked, there must be states \(p', q'\) such that:

1. \(p' = \delta(p, \sigma)\) and \(q' = \delta(q, \sigma)\), for some \(\sigma \in \Sigma\)
2. \((p', q')\) is marked D \(\Rightarrow p' \sim q'\) (by induction)

So there’s a \(w\) s.t. \(M_{p'}\) and \(M_{q'}\) have different output on \(w\)

Then, the string \(\sigma w\) distinguishes \(p\) and \(q\)!
Claim: If \((p, q)\) is not marked D by the Table-Filling algorithm, then \(p \sim q\)

Proof (by contradiction):
Suppose the pair \((p, q)\) is not marked D by the algorithm, yet \(p \sim q\) (call this a “bad pair”)

Then there is a string \(w\) such that \(|w| > 0\) and:

\(M_p\) and \(M_q\) have different outputs on \(w\)  \(\text{(Why is } |w| > 0?)\)

Of all such bad pairs, let \((p, q)\) be a pair with a \textit{minimum-length} distinguishing string \(w\)
Claim: If \((p, q)\) is not marked D by the Table-Filling algorithm, then \(p \sim q\)

Proof (by contradiction):
Suppose the pair \((p, q)\) is not marked D by the algorithm, yet \(p \not\sim q\) (call this a “bad pair”)
Of all such bad pairs, let \((p, q)\) be a pair with a \textit{minimum-length} distinguishing string \(w\)
\(M_p\) and \(M_q\) have different outputs on \(w\)
We have \(w = \sigma w'\), for some string \(w'\) and some \(\sigma \in \Sigma\)
Let \(p' = \delta(p, \sigma)\) and \(q' = \delta(q, \sigma)\). \((p', q')\) distinguished by \(w'\)
Then \((p', q')\) is also a bad pair!
But then \((p', q')\) has a SHORTER distinguishing string, \(w'\)
Contradiction!
Algorithm MINIMIZE
Input: DFA M
Output: Equivalent minimal-state DFA $M_{\text{MIN}}$

1. Remove all inaccessible states from M

2. Run Table-Filling algorithm on M to get:
   $\text{EQUIV}_M := \{ [q] \mid q \text{ is an accessible state of } M \}$

3. Define: $M_{\text{MIN}} = (Q_{\text{MIN}}, \Sigma, \delta_{\text{MIN}}, q_{0\text{MIN}}, F_{\text{MIN}})$
   
   $Q_{\text{MIN}} = \text{EQUIV}_M, \quad q_{0\text{MIN}} = [q_0], \quad F_{\text{MIN}} = \{ [q] \mid q \in F \}$

   $\delta_{\text{MIN}}( [q], \sigma ) = [ \delta( q, \sigma ) ]$  \text{ (well-defined??)}

Claim: $L(M_{\text{MIN}}) = L(M)$
The MINIMIZE Algorithm in Pictures

1. Remove all inaccessible states
The MINIMIZE Algorithm in Pictures

2. Run Table-Filling to get equiv classes

\[ [q] := \{ p \mid p \sim q \} \]
The MINIMIZE Algorithm in Pictures

3. Define $M_{\text{MIN}}$ with states = equiv classes

States of $M_{\text{MIN}} = \text{Equivalence classes}$ of states of $M$
Suppose we have proved the Claim is true. Assuming the Claim we can prove the Thm:

Proof of Thm: Let M’ be any minimal DFA for M. Since M’ is minimal, M’ has no inaccessible states and is irreducible (otherwise, we could make M’ smaller!) By the Claim, there is an isomorphism between M’ and the DFA $M_{\text{MIN}}$ that is output by MINIMIZE(M). That is, $M_{\text{MIN}}$ is isomorphic to every minimal M’.
Thm: $M_{\text{MIN}}$ is the *unique* minimal DFA equivalent to $M$

Claim: Let $M'$ be any DFA where $L(M') = L(M_{\text{MIN}})$ and $M'$ has no inaccessible states and $M'$ is irreducible. Then there is an *isomorphism* between $M'$ and $M_{\text{MIN}}$

Proof: We recursively construct a map from the states of $M_{\text{MIN}}$ to the states of $M'$

- **Base Case:** $q_{0,\text{MIN}} \mapsto q_{0}'$

- **Recursive Step:** If $p \mapsto p'$

  $\xrightarrow{\sigma}$  $\xrightarrow{\sigma}$

  Then $q \mapsto q'$
Base Case: $q_{0 \text{MIN}} \rightarrow q'_0$

Recursive Step: If $p \rightarrow p'$

Then $q \rightarrow q'$
Base Case: $q_{0\ MIN} \mapsto q_0'$

Recursive Step: If $p \mapsto p'$

\[
\begin{array}{c}
\downarrow \sigma & \downarrow \sigma \\
q & q' \\
\end{array}
\]

Then $q \mapsto q'$

Claim: Map is an isomorphism. Need to prove:

- The map is defined everywhere
- The map is well defined
- The map is a bijection (one-to-one and onto)
- The map preserves all transitions:

  If $p \mapsto p'$ then $\delta_{MIN}(p, \sigma) \mapsto \delta'(p', \sigma)$

  \((this\ follows\ from\ the\ definition\ of\ the\ map!)\)
Base Case: $q_{0, \text{MIN}} \rightarrow q_0'$

Recursive Step: If $p \rightarrow p'$

Then $q \rightarrow q'$

The map is defined everywhere

That is, for all states $q$ of $M_{\text{MIN}}$
there is a state $q'$ of $M'$ such that $q \rightarrow q'$

If $q \in M_{\text{MIN}}$, there is a string $w$ such that
$M_{\text{MIN}}$ is in state $q$ after reading in $w$

Let $q'$ be the state of $M'$ after reading in $w$.Claim: $q \rightarrow q'$ (proof by induction on $|w|$)
Base Case: $q_{0 \text{MIN}} \mapsto q_0'$

Recursive Step: If $p \mapsto p'$

Then $q \mapsto q'$

The map is onto: $\forall q' \exists q$ such that $q \mapsto q'$

Want to show: For all states $q'$ of $M'$ there is a state $q$ of $M_{\text{MIN}}$ such that $q \mapsto q'$

For every $q'$ in $M'$ there is a string $w$ such that $M'$ reaches state $q'$ after reading in $w$

Let $q$ be the state of $M_{\text{MIN}}$ after reading in $w$.

Claim: $q \mapsto q'$ (proof by induction on $|w|$)
Base Case: $q_{0 \text{MIN}} \mapsto q_0'$

Recursive Step: If $p \mapsto p'$

$\Downarrow \sigma \quad \Downarrow \sigma$

Then $q \mapsto q'$

$q \quad q'$

The map is well defined: $\forall q \exists! q'$ such that $q \mapsto q'$

Suppose there are states $q'$ and $q''$ such that $q \mapsto q'$ and $q \mapsto q''$

We show that $q'$ and $q''$ are indistinguishable, so it must be that $q' = q''$ (why?)
Suppose there are states $q'$ and $q''$ such that $q \mapsto q'$ and $q \mapsto q''$

Assume for contradiction $q'$ and $q''$ are distinguishable.
Map is 1-to-1: \( \forall p \neq q, p \mapsto q' \) and \( q \mapsto q'' \) \( \Rightarrow q' \neq q'' \)

*Proof by contradiction.* Suppose there are states \( p \neq q \) such that \( p \mapsto q' \) and \( q \mapsto q' \)

If \( p \neq q \), then \( p \) and \( q \) are distinguishable
How can we prove that two regular expressions are equivalent?