

**DFA Minimization**

Atri Rudra

April 28

---

**Announcements**

- Turn in H/W #4
- Handouts
  - Midterm exam topics list
  - Sample Midterm 1
  - Sample Midterm 2
  - Feedback Form
  - Solutions to H/W #3
    - If you did not pick up one last time
- No homework this week

---

**Puzzle**

- Use Myhill-Nerode theorem to prove that the following language is not regular:

  \[ L = \{ 0^n | n \text{ is prime} \} \]

---

**The equivalence relation \( \equiv_A \)**

- Let \( A \) be a language
- Given \( \equiv_A \), we know how to build minimized DFA for \( A \)
- Recall
  - For strings \( x \) and \( y \), \( x \equiv_A y \) iff
  - For all strings \( z \), either both \( xz \) and \( yz \) are in \( A \) or both are not in \( A \)

---

**The equivalence relation \( \equiv_M \)**

- Given DFA \( M \)
- Recall
  - String \( x \) and \( y \), \( x \equiv_M y \) iff
  - \( x \) and \( y \) end up in the same state in \( M \)
- Each equivalence class corresponds to a state
- Assume all states in \( M \) are reachable from the start state

---

**Relationship between \( \equiv_A \) and \( \equiv_M \)**

- Let \( M \) be such that \( A = L(M) \)
- If \( x \equiv_M y \) then \( x \equiv_A y \)
- An eqv class in \( \equiv_A \) is the union of eqv classes in \( \equiv_M \)
Basic idea

- In the minimized DFA, every state corresponds to an eqv. class in \( \equiv_A \)
- But we only know \( \equiv_M \)
- Group states in M to get states corresponding to \( \equiv_A \)

When should we group states?

- Given two states \( p \) and \( q \), when should we group them?
- When \( x \equiv_A y \)
- If for all strings \( z \), either
  - Both \( p \) and \( q \) go to a final state, or
  - Both go to a non-final state

In other words…

- For all strings \( z \), of length 0
  - Both \( p \) and \( q \) are final states or both are not
- For all strings \( z \), of length 1
  - Both \( p \) and \( q \) are final states or both are not
- For all strings \( z \), of length 2 …

Putting it together…

- Group \( p \) and \( q \) together, if
  - For all \( i \geq 0 \)
    - For all strings \( z \) of length \( i \), both \( p \) and \( q \) are in either final states or not
- Do not group \( p \) and \( q \), if
  - There exist an \( i \geq 0 \)
    - Exists a string \( z \) of length \( i \) that takes \( p \) to a final state but not \( q \) (or vice versa)

Stating it more formally

- \( p \equiv q \) if
  - For all strings \( z \) of length at most \( i \), either both \( p \) and \( q \) reach final state or neither does
- Thus, group \( p \) and \( q \) if
  - \( p \equiv q \) for all \( i \geq 0 \)
- Do not group \( p \) and \( q \) if
  - \( p \equiv q \) for some \( i \geq 0 \)

Questions?
The procedure

- We will decide in a top down manner
- First group all states together

- Separate the states that are not equivalent under $\equiv_0$
- Separate the states that are not equivalent under $\equiv_1$
- And so on…

And so on till when?

What we need…

- The process needs to terminate
- At termination, all grouped states belong to the same eqv class in $\equiv_A$
- Any two state in different groups must be in different eqv classes in $\equiv_A$