Decision Properties of Regular Languages

Given a (representation, e.g., RE, FA, of a) regular language \( L \), what can we tell about \( L \)?
- Since there are algorithms to convert between any two representations, we can choose the rep that makes the test easiest.

Membership

Is string \( w \) in regular language \( L \)?
- Choose DFA representation for \( L \).
- Simulate the DFA on input \( w \).

Emptiness

Is \( L = \emptyset \)?
- Use DFA representation.
- Use a graph-reachability algorithm to test if at least one accepting state is reachable from the start state.

Finiteness

Is \( L \) a finite language?
- Note every finite language is regular (why?), but a regular language is not necessarily finite.

DFA method:
- Given a DFA for \( L \), eliminate all states that are not reachable from the start state and all states that do not reach an accepting state.
- Test if there are any cycles in the remaining DFA; if so, \( L \) is infinite, if not, then \( L \) is finite.

RE method: Almost, we can look for a * in the RE and say its language is infinite if there is one, finite if not. However, there are exceptions, e.g. \( 0*1 \) or \( 0^* \). Thus:

1. Find subexpressions equivalent to \( \emptyset \) by:
   - (Basis) \( \emptyset \) is; \( \epsilon \) and \( a \) are not.
   - (Induction) \( E + F \) is iff both \( E \) and \( F \) are;
     \( EF \) is if either \( E \) or \( F \) are; \( E* \) never is.

2. Eliminate subexpressions equivalent to \( \emptyset \) by:
   - Replace \( E + F \) or \( F + E \) by \( F \) whenever \( E \) is and \( F \) isn’t.
   - Replace \( E* \) by \( \epsilon \) whenever \( E \) is equivalent to \( \emptyset \).
3. Now, find subexpressions that are equivalent to $\epsilon$ by:

- **(Basis)** $\epsilon$ is; $a$ isn’t.
- **(Induction)** $E + F$ is iff both $E$ and $F$ are; ditto $EF$; $E^*$ is iff $E$ is.

4. Now, we can tell if $L(R)$ is infinite by looking for a subexpression $E^*$ such that $E$ is not equivalent to $\epsilon$.

**Example**

Consider $(0 + 1)^* + 1\emptyset^*$.

- **Step 1**: $\emptyset$ (twice) and $1\emptyset$ are subexpressions equivalent to $\emptyset$.
- **Step 2**: $0^* + 1\epsilon$ remains.
- **Step 3**: only subexpression $\epsilon$ is equivalent to $\epsilon$.
- **Since $0$ is starred, language is infinite.**

**Minimization of States**

- Real goal is testing equivalence of (reps of) two regular languages.
- Interesting fact: DFA’s have unique (up to state names) minimum-state equivalents.
  - But proof in course reader doesn’t quite get to that point.

**Distinguishable States**

Key idea: find states $p$ and $q$ that are distinguishable because there is some input $w$ that takes exactly one of $p$ and $q$ to an accepting state.

- **Basis**: any nonaccepting state is distinguishable from any accepting state ($w = \epsilon$).
- **Induction**: $p$ and $q$ are distinguishable if there is some input symbol $a$ such that $\delta(p, a)$ is distinguishable from $\delta(q, a)$.
  - All other pairs of states are indistinguishable, and can be merged into one state.

**Example (Very Simple)**

Consider:
• $p$ is distinguishable from $q$ and $r$ by basis.

Can we distinguish $q$ from $r$?

• No string beginning with 0 works, because both states go to $p$, and therefore any string of the form $0x$ takes $q$ and $r$ to the same state.

• No string beginning with 1 works.
  ♦ Technically, $\delta(q, 1) = r$ and $\delta(r, 1) = q$ are not distinguishable. Thus, induction does not tell us $q$ and $r$ are distinguishable.

  ♦ What happens is that, starting in either $q$ or $r$, as long as we have inputs 1, we are in one of the accepting states, and when a 0 is read, we go to the same state forever after.

**Constructing the Minimum-State DFA**

• For each group of indistinguishable states, pick a “representative.”
  ♦ Note a group can be large, e.g., $q_1, q_2, \ldots, q_k$, if all pairs are indistinguishable.

  ♦ Indistinguishability is transitive (why?) so indistinguishability partitions states.

• If $p$ is a representative, and $\delta(p, a) = q$, in minimum-state DFA the transition from $p$ on $a$ is to the representative of $q$’s group (to $q$ itself if $q$ is either alone in a group or a representative).

• State state is representative of the original start state.

• Accepting states are representatives of groups of accepting states.
  ♦ Notice we could not have a “mixed” (accepting + nonaccepting) group (why?).
- Delete any state that is not reachable from the start state.

**Example**

For the DFA above, $p$ is in a group by itself; $\{q, r\}$ is the other group.

![Diagram]

**Why Above Minimization Can't be Beaten**

Suppose we have a DFA $A$, and we minimize it to construct a DFA $M$. Yet there is another DFA $N$ that accepts the same language as $A$ and $M$, yet has fewer states than $M$. Proof contradiction that this can't happen:

- Run the state-distinguishability process on the states of $M$ and $N$ together.
- Start states of $M$ and $N$ are indistinguishable because $L(M) = L(N)$.
- If $\{p, q\}$ are indistinguishable, then their successors on any one input symbol are also indistinguishable.
- Thus, since neither $M$ nor $N$ could have an inaccessible state, every state of $M$ is indistinguishable from at least one state of $N$.
- Since $N$ has fewer states than $M$, there are two states of $M$ that are indistinguishable from the same state of $N$, and therefore indistinguishable from each other.
- But $M$ was designed so that all its states are distinguishable from each other.
- We have a contradiction, so the assumption that $N$ exists is wrong, and $M$ in fact has as few states as any equivalent DFA for $A$.
- In fact (stronger), there must be a 1-1 correspondence between the states of any other minimum-state $N$ and the DFA $M$, showing that the minimum-state DFA for $A$ is unique up to renaming of the states.