Sec. 1.4 (pp. 77-81). **Nonregular Languages**

- Not all languages are regular (i.e. not all recognizable by some FA or expressible as a regex)
  - Consider $B = \{0^n1^n | n \geq 0\}$
    - Need to “remember n” when we see 01 transition
    - But no way to count to an arbitrarily large number
  - $C = \{w | w \text{ has equal } \# \text{ 0s and 1s}\}$ also not regular
    - Again have to “count”
  - However, $D = \{w | w \text{ has equal } \# \text{ of 01 and 10 substrings}\}$ is regular (see Prob. 1.48)
• How to show some languages non-regular?
• Observation: If the set of strings is infinite
  • Then matching regex must have at least one “*” or “+”
  • I.e. $R_xR_y^*R_z$ where $R_x$, $R_y$, $R_z$ all smaller regexs
  • E.g. $L = ac (bb \cup aa)^* ca$
  • acbbca is in $L$
  • but so is acca, acbbbca, acbbbbbbca, ......
  • i.e. there are an infinite number of strings of the form $ac(bb)^nca$ for all $n \geq 0$ also in $L$!
  • In general (with caveats) if $w$ is in $L$, there is some $w=xyz$ so that for all $n$, so is $xy^nz$
• So in general if we find one string we know is in $L$
• Then an infinite number of other strings also in $L$
• Why is this useful? Assume want to show $L$ is NOT regular
  • Assume $L$ IS regular
  • Find a string $w$ known to be in $L$
  • Look at all possible ways of dividing into $w=xyz$
    • $x$ from some $R_x$, $y$ from some $R_y$, $z$ from some $R_z$
  • In each case show for some $k$, $xy^kz$ is not in $L$
  • Contradiction! Assumption that $L$ is regular is FALSE
• Thus $L$ cannot be a regular language
• (p. 78) **PUMPING LEMMA.** If A is regular, then
  • There is some number \( p \) (called the **pumping length**)
  • Where if \( s \) is any string in A whose length \( \geq p \)
  • Then \( s \) can be divided somehow into **3 pieces** \( s = xyz \)
    • \( |y| > 0 \), (i.e. \( y \) cannot be \( \varepsilon \))
    • \( |xy| \leq p \), (note either \( x \) or \( y \) or both may be \( \varepsilon \))
    • **For any** \( i \geq 0 \), **then** \( xy^i z \) **is also in** A
  • What this means: If **L is regular** language of infinite size
    • \( L \) has associated with it some string length \( p \)
      • Such that if you take **any** string \( w \) from \( L \) where \( |w| \geq p \)
      • Then you can **always** write \( w \) as concatenation \( w = xyz \)
        for some strings \( x, y, \) and \( z \)
      • Such that the strings \( xz, xyz, xyyz, xyyyyz, \ldots xy^i z \) all in \( L \)
    • Note: finite languages cannot be pumped
  • Example: \{ade, abcde, abcabcde, \ldots\}
    • Regex = \( a(bc)^*de \)
    • GNFA equivalent has 3 states
    • \( p=4, x=a, y=bc, z=de \)
    • Easiest to see the \( y \) in a DFA loop, or “*” in the regex
• What this means: If **L is not regular**, then L does not obey the pumping lemma
  • Can use pumping lemma in a **proof by contradiction** to show language is not regular
    • Assume L is regular
    • Then there must exist some p (we don’t need to know exact value)
    • Show that there is **always** some string w in L, |w| ≥ p, that cannot be pumped, regardless of how we partition it into some xyz
      • Need find ONLY ONE SUCH STRING
    • Thus assumption is false and L not regular
• (p. 78) Proof in outline:
  • Assume $M = (Q, \Sigma, q_1, \delta, F)$ accepts $A$
  • Choose $p = \# \text{ of states in } M$
    • $Q = \{q_1, q_2, \ldots, q_p\}$
  • If no string in $A$ is $\geq p$, then theorem obviously true
  • Assume $s = s_1s_2 \ldots s_n$, $n \geq p$ ($n$ is $\# \text{ of characters in string}$)
  • Then state sequence must be $(r_0, r_1, \ldots, r_n)$ (see fig. 1.72)
    • where $r_0 = q_1$
    • and $\delta(r_{i-1}, s_i) = r_i$
  • But since $n \geq p$, then $n+1 > p$
    • But since only $p$ states, we must have $\text{repeated}$ $n+1-p$ states
    • Assume $r_j$ is $1^{\text{st}}$ state that is repeated
      • $s_{j+1}$ is $1^{\text{st}}$ character to cause leaving $r_j$
    • Also assume $s_k$ is $1^{\text{st}}$ character that causes re-entry to state $r_j$
    • Since we are back at $r_j$, we could repeat $s_{j+1} \ldots s_k$ forever
  • The substring $s_{j+1} \ldots s_k$ is thus $y$
    • We could keep repeating $s_{j+1} \ldots s_k$ arbitrarily often and still end up at $r_j$ – i.e. $(s_{ij+1} \ldots s_k)^i$ for $i \geq 0$
  • And $x = s_1s_2 \ldots s_j$, $z = s_{k+1} \ldots s_n$,
  • Either/both $x$ and $z$ could be $\epsilon$
Use lemma to show B not regular – by contradiction

- Assume B regular
- Thus there is some p such that all strings of length \( \geq p \) can be pumped
- Find a string s in B that is \( \geq p \), but cannot be pumped
  - Look at all possible ways to divide string into xyz
  - For each way find an i such that \( xyz^i \) not in B
- When found, we have a contradiction!
- Thus B is NOT regular

Examples

- P.80: \( B = \{0^n1^n \mid n \geq 0\} \)
  - Look at 3 cases of substrings: all 0s, ..01.., all 1s
- P.80: \( C = \{w \mid w \text{ has equal } \# \text{ of 0's and 1s}\} \)
  - Look at \( s = 0^p1^p \)
- P.81: \( F = \{ww \mid w \text{ in } \{0,1\}^*\} \)
  - Look at \( s = 0^p10^p1 \)
- P.82: \( D = \{1^{n^2} \mid n \geq 0\} \)
  - Look at \( s = 1^{p^2} \)
- P. 82: \( E = \{0^i1^j \mid i>j\} \)
  - Look at \( s = 0^{p+1}1^p \)
- Also look at problems 1.53-1.58