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

- Consider the following regular languages (show regex and NFA/DFA, count # of states)
  - \{01^n \mid n \geq 0\}
  - \{0^n1^m \mid n \geq 0, m \geq 0\}
  - \{0^k1^k \mid 0 \leq k \leq n, \text{ for some fixed } n\}
  - \{w \mid w \text{ has equal # of 01 and 10 substrings}\} \text{ is regular (see Prob. 1.48)}
- If a language is finite, is it always regular? YES

- Now is \{0^n1^n \mid n \geq 0\} regular?

- Not all languages are regular (i.e. not all recognizable by some FA or expressible as a regex)
  - Need to “count & remember” some transition
    - But no way to count to an arbitrarily large number
  - \(C = \{w \mid w \text{ has equal # 0s and 1s}\}\) also not regular
    - Again have to “count”
• How to show some languages non-regular?

• Observation: If the set of strings $L$ is infinite & regular
  • 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, acbbbbca, 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
  • Proof by contradiction: 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 \( \epsilon \))
    • \( |xy| \leq p \), (note either \( x \) or \( y \) or both may be \( \epsilon \))
  • **For any \( i \geq 0 \), then \( xy^iz \) 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 \( z \) (i.e. at least one)
    • Such that the strings \( xz, xyz, xyyz, xyyyz, \ldots \ xy^iz \) 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 \( M = (Q, \Sigma, \delta, F) \) accepts \( A \)
  • Assume \( p = \# \) 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 \# 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 \textit{repeated} \( n+1-p \) states
  • Assume \( r_j \) is \textit{1st} state that is repeated
    • \( s_{j+1} \) is \textit{1st} character to cause leaving \( r_j \)
  • Assume \( s_l \) is \textit{1st} character that causes re-entry to state \( r_j \)
  • Since we are back at \( r_j \), we could repeat \( s_{j+1} \ldots s_l \) forever
    • i.e. \( s_{l+1} = s_{j+1}, s_{l+2} = s_{j+2} \ldots s_{l+l} = s_l \)
  • The substring \( s_{j+1} \ldots s_l \) (of length \( l-j \)) is \textit{thus} \( y \)
    • We could keep repeating \( s_{j+1} \ldots s_l \) arbitrarily often and still end up at \( r_j \) — i.e. \( (s_{j+1} \ldots s_l)^i \) for \( i \geq 0 \)
  • And \( x = s_1s_2 \ldots s_j, z = s_{j+i(l-j)} \ldots s_n, \)
  • Either/both \( x \) and \( z \) could be \( \varepsilon \)
• Use lemma to show B not regular – by contradiction
  • Assume B regular
  • Thus there is some p such that all strings of length ≥ p can be pumped
  • Find a string s in B that is ≥ p, but cannot be pumped
    • Look at all possible ways to divide string into xyz
    • For each way find an i such that xy^iz 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 # 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^p1^p0^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
Summary: Applying Pumping Lemma

- **The Lemma**: If L is regular and infinite, then
  - There is guaranteed to be some integer p such that
  - If you look at *ANY string s* where |s|≥p
  - You can *always find at least one partitioning* s=xyz where
    - |y|≥1 and |xy|≤p
    - AND xy^i z is also in L FOR ALL i≥0 (we are “pumping” the string)
- **How to apply**: To show that L IS NOT Regular
  - Assume L IS regular
  - You only need find *one string s in L, |s|≥p, that does not pump*
    - Choose a string where you can easily identify:
      - What are all the possible values of xy
      - And you can id what y is for any of these
      - Work thru all possible xy strings partitions
        - There are at most p*(p-1) of them:
          - Show that *each possibility* has at least one i where xy^i z does not belong to L
  - **Example**: \{0^n1^n | n ≥ 0\}
    - If we choose 0^p1^p then we know
      - xy must be all 0s
      - And thus y must be one or more 0s and no 1s
      - And z holds all p 1s (and perhaps some 0s from the end of the 1st half)
    - Now *regardless of what xy actually is* (need only find one value of i)
      - i=0 removes just 0s from the string and thus # of 0s is less than # of 1s, AND thus xy^0 z IS NOT IN L
      - Alternatively if we choose i=2, then we “add” more 0s to the string and thus more 0s than 1s, AND thus xy^2 z IS NOT IN L
    - Thus we have found a string that does not pump, and **THUS L CANNOT BE REGULAR**