Last Time:
- Strings
- Languages
  - union, intersection, complement
  - concatenation
    \[ L_1 L_2 = \{ xy \mid x \in L_1, y \in L_2 \} \]
  - Kleene star
    \[ L^* = \bigcup_{i=0}^{\infty} L^i \quad (\emptyset^* = \{\epsilon\}) \]

Regular Languages
- all langs obtainable from
  - union, concat, star
  - (starting from trivial base cases)

Inductive Def'n
(i) \( \emptyset, \{\epsilon\}, \{a\} \) are regular langs \( \forall a \in \Sigma \)
(ii) if \( L_1, L_2 \) are regular langs then so are \( L_1 \cup L_2, L_1 L_2, L_1^* \)

(Nothing else is regular unless it can be obtained by finite # of appl of (i),(ii))

Ex
\( \Sigma = \{0,1\} \)
- a) \( \{0110\} \) is regular
  \[ = \{0\} \{1\} \{1\} \{0\} \checkmark \]
- b) \( \{0110, 01, 0\} \) is regular
  \[ = \{0\} \{1\} \{1\} \{0\} \cup \]
c) more generally, all finite langs are regular
d) \( \{x \in \{0,1\}^* \mid |x| \text{ is odd}\} \)
is regular
\[= \{00, 01, 10, 11\}^* \cdot \{0,1\}^* \]
\[= (\{0\}\{0\} \cup \{0\}\{1\} \cup \{1\}\{0\} \cup \{1\}\{1\})^* \cdot (\{0\} \cup \{1\}) \]

**Notation** regular exprs

(i) \( \phi, \varepsilon, a \) are regular exprs for \( \phi, \{\varepsilon\}, \{a\} \) \( \forall a \in \Sigma \)

(ii) if \( r_1, r_2 \) are reg. exprs for \( L_1, L_2 \)

then \( (r_1 + r_2) \) are reg exprs for \( L_1 \cup L_2 \)

\( r_1 r_2 \) \( L_1 L_2 \)

\( r_1^* \) \( L_1^* \)

Let \( L(r) \) denotes lang. corresponding to expr \( r \).

**Ex** \( (00 + 01 + 10 + 11)^* (0+1) \)

**Rmk**
- omit unnecessary parentheses
  - prec order: \( *, +, \cdot \)
- shorthand: \( r^+ = r \cdot r^* \)
- a lang may have mult. reg exprs
  - e.g. \( (0+1)(0+1)^* (0+1) \leq (0+1) ((0+1)(0+1))^* \)
  - useful way to specify patterns

**Ex** \( \Sigma = \{0,1,3\} \)
a) all strings with 00 as a substring

\[(0+1)^* 00 (0+1)^*\]

\[(00 + 001)^*\]

b) all strings with 00 as a substring and having even length

2 cases:
- Part before 00: even
- Part after 00: even

or
- Odd
- Odd

\[(((0+1)(0+1))^* 00 ((0+1)(0+1))^* + ((0+1)(0+1))^* (0+1)00 ((0+1)(0+1))^* (0+1)\]

c) all strings with even # of 0's

\[((00)*\]

\[((00+1)*)^*\]

\[(1^* 01^* 01^*)^* \]

or
\[1^* 01^* 01^* 1^*\]

or
\[1^* (01^* 01^*)^*\]

d) all strings \textbf{not} beginning with 00
d) all strings not beginning with 00

\[\text{not } 00 (0+1)^*\]

Cases: begin with 1 √
or begin with 01 √

\[(1 + 01)(0+1)^* + 0 + 3\]

e) all strings not containing 00 as a substring

\[(1 + 01)^* (0+3)\]

f) all strings with even # of 0's and even # of 1's

idea - divide into blocks of two

\[00, 01, 10, 11\]

\[\((00 + 11)^* (01 + 10) (00 + 1)^* (01 + 10)(00 + 11)^*\]^*

+ \[(00 + 11)^*\]

f') odd # 0's & odd # 1's

8) all strings with even # 0's
& # 1's div by 3
& not containing 0106

hard but possible!

( in general, how to take intersection, take complement?)
(in general, how to take intersection; take complement?)

h) \{ 0^i 1^i \mid i \geq 0 \}

impossible! (but how to prove it?)

Some identities

a) \( L((r_1 + r_2)r_3) = L(r_1r_3 + r_2r_3) \)

b) \( L((r^*)^*) = L(r^*) \)

c) \( L((rs)^*r) = L(r(sr)^*) \)

d) \( L((r+s)^*) = L((r^*+s^*)^*) = L((r^*s^*)^*) = \ldots \)

(in general, how to tell if two reg exprs are equiv.? )