Regular Expressions

Built up from characters and the empty string \( (\varepsilon \text{ or } \lambda) \)
via the operations \(+\) (also written \( \cup \) or \(|\)), \( \cdot \), and \(*\):
this \( + \) that: this or that
this \( \cdot \) that: this followed by that
\((\text{this})^*\): zero or more occurrences of this. **Examples:**
\[(a + b)(a + c) = aa + ac + ba + bc \quad (0 + 01)(10 + 0) = 010 + 00 + 0110\]
\[(a + bc)^* = \{\varepsilon, a, bc, aa, abc, bcbc, bca, aaa, \ldots\}\]
But not \(bac\) for instance.
\[(00)^* = \{\varepsilon, \varepsilon, \varepsilon\varepsilon, \ldots\} = \{0^n : n \text{ is even}\}\]
\[(11)^*1 = \{1, 111, 11111, \ldots\} = \{1^n : n \text{ is odd}\}\]
\(1(11)^*\) is equivalent.

Now how about strings over \(\{0,1\}\) containing an odd number of \(1s\)?
Try even \(1s\) first: \((0^*10^*10^*)^*\). This was not comprehensive, did not match \(0, 00, \ldots\)
Then add a \(1\) to make it odd: \((0^*10^*10^*)^*1\). Is that good? **Sound? Comprehensive?**
(needs to allow ending in \(0s\)) Note incidentally that \(0^*0^* = 0^*\).

Economical is: \((0^*10^*1)^*0^*10^*\). Fixing the even case, use \((0^*10^*1)^*0^*\).
How about \(\{x \in \{0,1\}^* : \text{every 5th char of } x \text{ is a } 1\}\)? We can try \((1(0 + 1)^4)^*\).
But this forces the string to have length a multiple of 5. To allow other lengths, try:
\[(1(0 + 1)^4)^*(\varepsilon + 1(\varepsilon + 0 + 1)(\varepsilon + 0 + 1)(\varepsilon + 0 + 1))\] [Will pause for why it works.]
Now how do we apply these ideas to make a regular expression for Wed.'s language $L(M_{5,2}) = \{ x : x \text{ has an odd } \# \text{ of 1s in positions } \equiv 2 \text{ mod } 5 \}$?

First, we need at least 3 chars, to get at least one 1 in such a position.

The first two such chars are arbitrary: $(0 + 1)^2$. Then we see the equation:

$$L(M_{5,2}) = (0 + 1)^2 \cdot L(M_{5,0})$$

Thus we can focus on "blocks" of the form $Z = 0(0 + 1)^4$ or $I = 1(0 + 1)^4$.

Take our previous "template" for an odd number of 1's and sub. 0 by $Z$, 1 by $I$:

$$L(M_{5,0}) = (Z^{*}IZ^{*}I)^{*}Z^{*}IZ^{*}$$

But this has another "overkill" problem: The last 1 in a multiple-of-5 position need not be followed by 4 chars. So instead define $Y = (0 + 1)^40$. Then:

$$L(M_{5,2}) = (0 + 1)^2 \cdot (Z^{*}IZ^{*}I)^{*}Z^{*}1Y^*(e + 0 + 1)^4.$$ 

$$= (0 + 1)^2 \cdot \left[ (0(0 + 1)^4)^{*}1(0 + 1)^4((0(0 + 1)^4)^{*}1(0 + 1)^4)^{*}1((0 + 1)^40)^{*}(e + 0 + 1)^4. \right]$$

Yuck---?---! But we got it by top-down reasoning.
New Lecture Idea: Talk in terms of "Trominoes" (like dominoes but with middle panel):

\[ [p, c, q] \quad \text{where } p \text{ and } q \text{ are numbers and } c \text{ is a char} \]

Or: \[ [p, \epsilon, q] \quad \text{using the empty string. (Not allowed to rotate them 180°)} \]

A sequence of trominoes is "legal" provided they "match like dominoes":

\[ [q_0, c_1, q_1][q_1, c_2, q_2][q_2, c_3, q_3] \ldots [q_{n-2}, c_{n-1}, q_{n-1}][q_{n-1}, c_n, q_n] \]

Its yield is the string \( x = c_1c_2c_3 \ldots c_n \). If some \( c_i \) are really \( \epsilon \) then \( |x| < n \).

**Definition:** A nondeterministic finite automaton (NFA) \( N \) is a set of trominoes, in which one number \( s = q_0 \) is "start" and certain numbers are "final". The language \( L(N) \) is the set of yields of legal sequences that begin with \( s \) and end with a final \( \# \).

\( N \) is deterministic (a DFA) if every \( p, c \) pair \((c \text{ a char, no } \epsilon)\) has one tromino \([p, c, q]\).