12 Apr 2024
Turing machine examples

Announcements:
1. No homework release today!
   Problem Set 8 to be released mid-week next week, we expect it will have 4 problems. Will be due Thurs, Apr 25.

2. Prelim 2 grading schedule, we're getting started tomorrow. We can't finish until after we grade the make-up prelim, which is administered Mon. evening.

Reminder: Turing machine has:
- finite state set $Q$
- 2 alphabets $\Sigma \subseteq \Gamma$
- infinite tape with 1 symbol on left end that cannot be overwritten
- transition rule $\delta(q,x) = (q', y, d)$
  
  "If reading $x$ in state $q$, write $y$, enter state $q'$, move in direction $d$,"

  $d$ is either LEFT or RIGHT, never stand still.

A standing still transition can be simulated using 2 steps + an extra "move left" state.
Ex. 1. Input alphabet \( \{0,1,3\} \).

TM should accept \( 0^n \), \( n \in \mathbb{N} \).

Ex. 2. Accept binary strings that start with 0 and alternate 0 and 1 thereafter.

Ex. 3. Accept binary strings that are palindromes (read the same backward as forward).

Tape alphabet will be \( \{\uparrow, \downarrow, 0, 1, \wedge, \vee\} \), symbols already checked.
1: \text{repeat}
2: \text{more right}
3: \text{if } x \text{ has } \\
4: \text{accept}
5: \text{write } X
6: \text{repeat}
7: \text{move right}
8: \text{until symbol on tape } = \text{ U or has } \\
9: \text{move left}
10: \text{if symbol on tape has } \\
11: \text{accept}
12: \text{if symbol on tape } \neq x
13: \text{reject}
14: \text{else}
15: \text{write } \\
16: \text{repeat}
17: \text{move left}
18: \text{until symbol on tape has } \\
19: \text{forever}

\text{Translation to TM:}

- \text{“marking with } \wedge \text{”} \\
  \rightarrow \text{enlarging tape alphabet} \\
  \{0, 1\} \rightarrow \{0, 1, \wedge, 7\}

- \text{State set} \quad Q = [18] \times \{0, 1\} \\
  \wedge \quad \wedge \\
  \text{program counter} \quad \text{value of } X
Can have finite # of variables each taking values in a finite set.

If the pseudocode uses function calls, the call stack depth should be bounded by a finite # that depends only on the program, not the input.