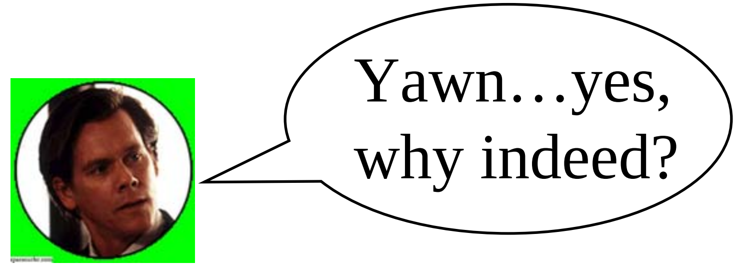
Pumping Lemma for CFLs

- Intuition: If L is CF, then some CFG G produces strings in L
  - If some string in L is very long, it will have a very tall parse tree
  - If a parse tree is taller than the number of distinct variables in G, then *some variable A repeats* $\Rightarrow$ A will have at least two sub-trees
  - We can pump up the original string by replacing A's smaller sub-tree with larger, and pump down by replacing larger with smaller
- Pumping Lemma for CFLs in all its glory:
  - If L is a CFL, then there is a number p (the "pumping length") such that for all strings *s* in L such that $|s| \geq$ p, there exist *u*, *v*, *x*, *y*, and *z* such that $s = uvxyz$ and:

1. $uv^ixy^iz \in$ L for all $i \geq 0$, and
2. $|vy| \geq 1$, and
3. $|vxy| \leq$ p.

Why is the PL useful?



- Can use the pumping lemma to show a language L is *not context-free*
  - 5 steps for a proof by contradiction:

1. Assume L is a CFL.
2. Let p be the pumping length for L given by the pumping lemma for CFLs.
3. Choose cleverly an *s* in L of length at least p, such that
4. For *all possible ways* of decomposing *s* into *uvxyz*, where $|vy| \geq 1$ and $|vxy| \leq p$,
5. Choose an $i \geq 0$ such that uv^ixy^iz is not in L.

Example 1

Oh boy…
Jolly good

- Show that $L = \{0^n 1^n 0^n \mid n \geq 0\}$ is not a CFL

1. Assume L is a CFL.
2. Let p be the pumping length for L given by the pumping lemma for CFLs.
3. Let $s = 0^p 1^p 0^p$ (note that $|s| > p$)
4. For *all possible ways* of decomposing $s = 0^p 1^p 0^p$ into $uvxyz$, where $|vy| \geq 1$ and $|vxy| \leq p$,
5. We need $i \geq 0$ such that $uv^i xy^i z$ is not in L:

   Case 1: Both v and y contain only 0s or only 1s

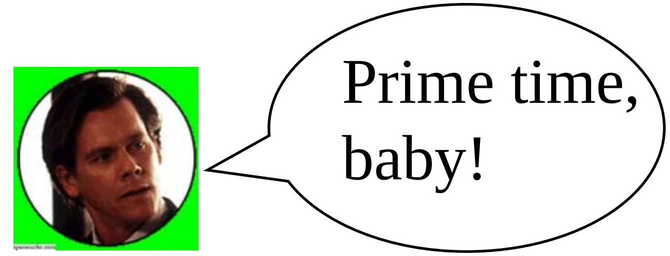
   - Then uv^2xy^2z contains unequal no. of 0s, 1s, and 0s.

   Case 2: v or y contain both 0 and 1

   - Then uv^2xy^2z is not of the form 0*1*0*.

In both cases, uv^2xy^2z is not in L, contradicting pumping lemma. Therefore L cannot be a CFL.

Example 2



- Show $L = \{0^n \mid n$ is a prime number$\}$ is not a CFL
  1. Assume L is a CFL.
  2. Let p be the pumping length for L given by the pumping lemma for CFLs.
  3. Let $s = 0^n$ where n is a prime $\geq p$
  4. Consider *all possible ways* of decomposing *s* into *uvxyz*, where $|vy| \geq 1$ and $|vxy| \leq p$.
     Then, $vy = 0^r$ and $uxz = 0^q$ where $r + q = n$ and $r \geq 1$
  5. We need an $i \geq 0$ such that $uv^ixy^iz = 0^{ir+q}$ is not in L.
     ($i = 0$ won't work because q could be prime: e.g. $2 + 17 = 19$)
     Choose $i = (q + 2 + 2r)$. Then, $ir + q = qr + 2r + 2r^2 + q =$
     $q(r+1) + 2r(r+1) = (q+2r)(r+1) =$ not prime (since $r \geq 1$).

So, 0^{ir+q} is not in L $\Rightarrow$ contradicts pumping lemma. L is not a CFL.

Closure properties of CFLs

- You showed in homework that CFLs are closed under union, concatenation and star.

- How about intersection?
- How about complement?

Two surprising results about CFLs

- CFLs are <u>not closed</u> under intersection
  - **Proof**: $L_1 = \{0^n1^n0^m \mid n, m \geq 0\}$ and $L_2 = \{0^m1^n0^n \mid n, m \geq 0\}$ are both CFLs but $L_1 \cap L_2 = \{0^n1^n0^n \mid n \geq 0\}$ is not a CFL.
- CFLs are <u>not closed</u> under complement
  - **Proof by contradiction**:

    Suppose CFLs are closed under complement.

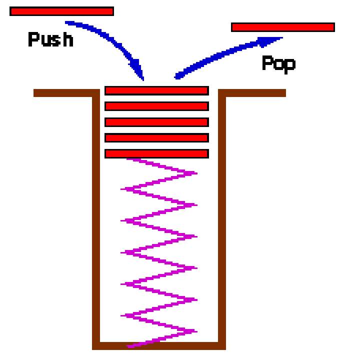
    Then, for L_1, L_2 above, $\overline{\overline{L_1} \cup \overline{L_2}}$ must be a CFL (since CFLs are closed under $\cup$ - see this week's homework).

    But, $\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$ (by de Morgan's law).

    $L_1 \cap L_2 = \{0^n1^n0^n \mid n \geq 0\}$ is not a CFL $\Rightarrow$ contradiction.

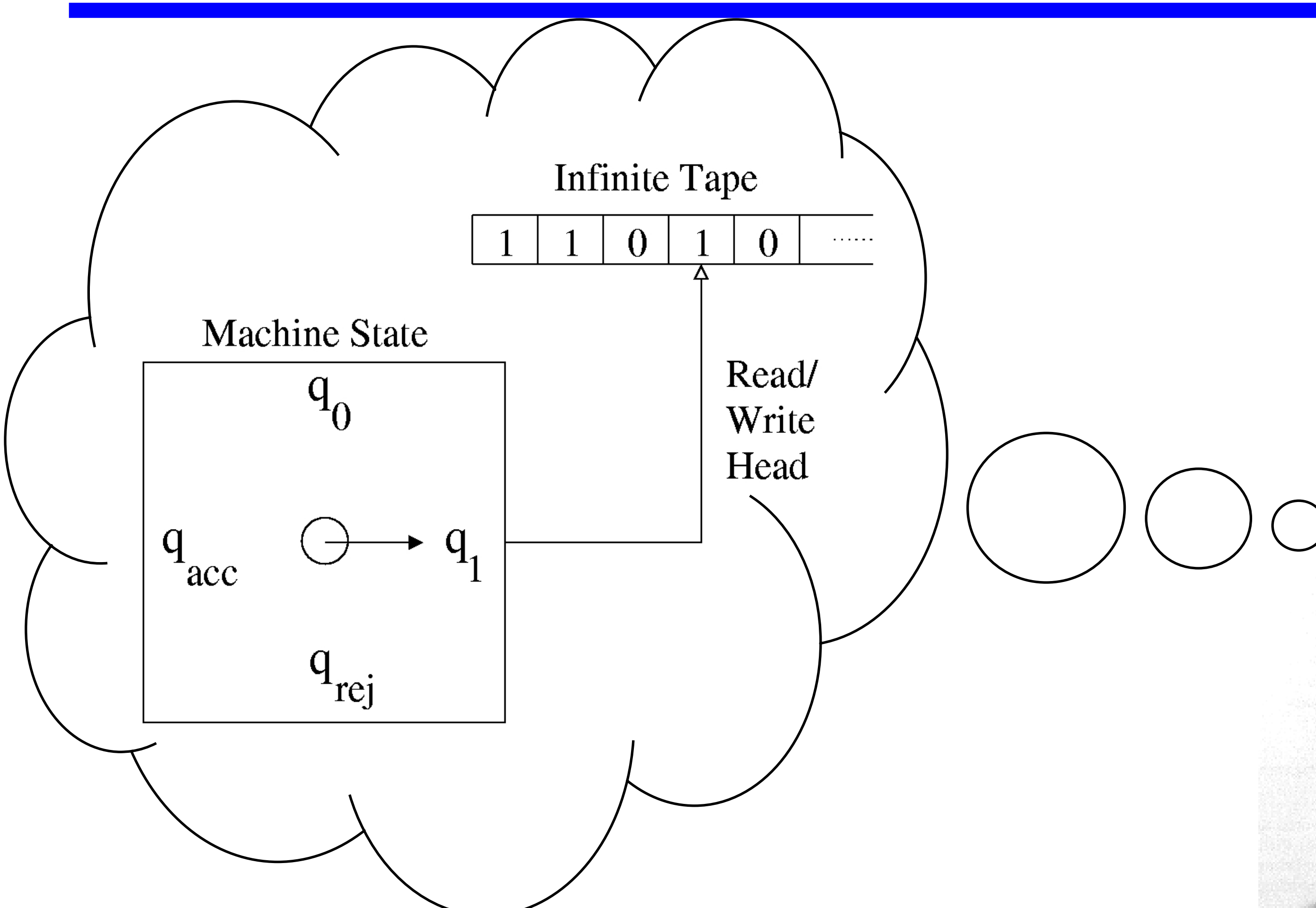
    Therefore CFLs are not closed under complement.

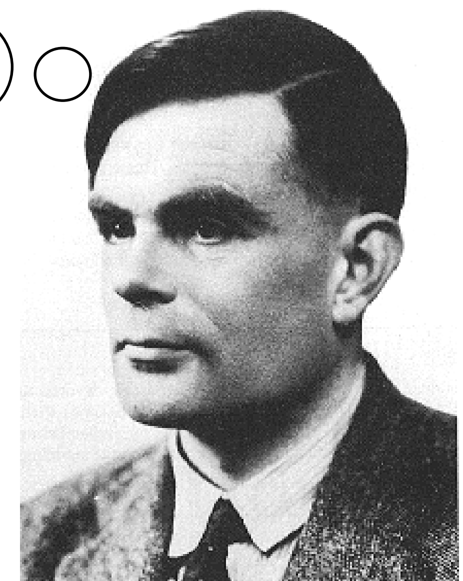
Can we make PDAs more powerful?

✦ PDA = NFA + 

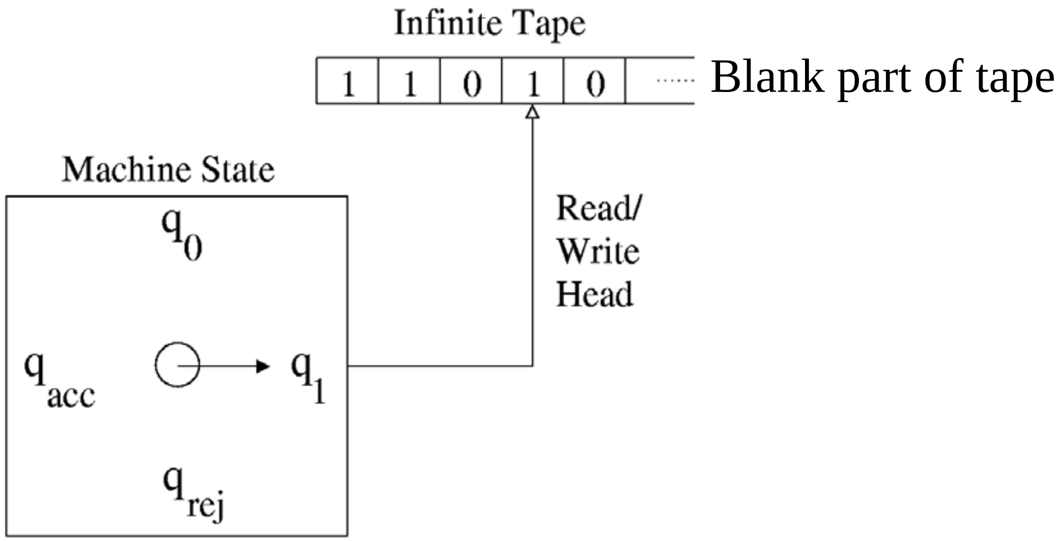
What if we allow arbitrary reads/writes to the stack instead of only push and pop?

Enter…the Turing Machine





Turing Machines



Just like a DFA except:

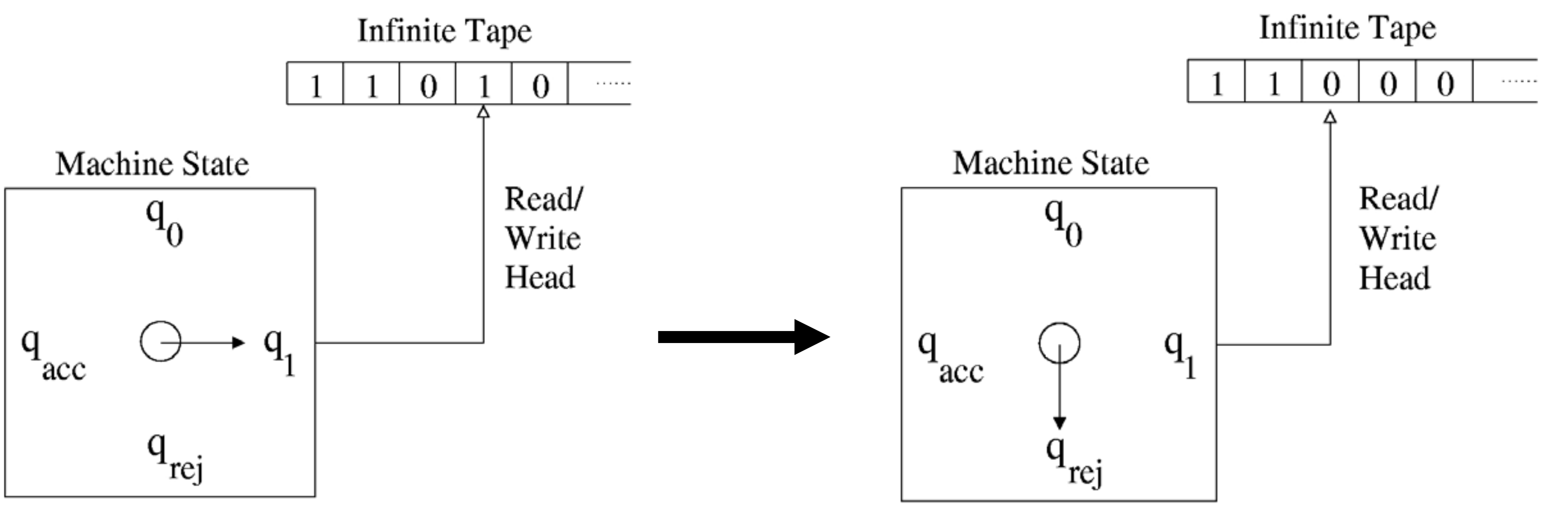
- You have an infinite "tape" memory (or scratchpad) on which you receive your input and on which you can do your calculations
- You can <u>read</u> one symbol at a time from a cell on the tape, <u>write</u> one symbol, then <u>move</u> the read/write pointer (head) left (L) or right (R)

Who was Turing?

- Alan Turing (1912-1954): one of the most brilliant mathematicians of the 20^{th} century (one of the "founding fathers" of computing)
- Click on "Theory Hall of Fame" link on class web under "Lectures"
- Introduced the Turing machine as a formal model of what it means to compute and solve a problem (i.e. an "algorithm")
  - Paper: `On computable numbers, with an application to the Entscheidungsproblem, Proc. London Math. Soc. 42 (1936).`

How do Turing Machines compute?

- δ(current state, symbol under the head) = (next state, symbol to write over current symbol, direction of head movement)



- Diagram shows: **$\delta(q_1,1) = (q_{rej}, 0, L)$** (R = right, L = left)
- In terms of "Configurations": $110q_1\underline{1}0 \Rightarrow 11q_{rej}\underline{0}00$

Next Time: Turing-Recognizable versus Decidable Languages

How does a TM accept a string?

How can a TM reject a string?

What is a decider TM?