CS 3313 Foundations of Computing:

Turing Machine Examples

http://gw-cs3313-2021.github.io

Turing Machine

- Takes two arguments:
  1. A state, in Q.
  2. A tape symbol in Γ.
- $\delta(q, Z)$ is either undefined or a triple of the form (p, Y, D).
  - p is a state.
  - Y is the new tape symbol.
  - D is a *direction*, L or R – move the tape head to the Left or Right
- Convention: If undefined then TM halts
  - If it halts in a final state then it accepts
  - If it halts in a non-final state then it rejects

Example 1: $L = \{0^{2^n} \mid n \geq 0\}$

- Check whether the amount of zeros being an exponent of 2.
  - Given input 0^m, where it should be $m = 2^n, n \geq 0$.

➢ **Approach**: try to reduce the length by half each iteration.

$$|w|: 2^n \to 2^{n-1} \to \cdots \to 8 \to 4 \to 2 \to 1$$

➢ Lastly, accept if there's a single 0 left.

We observe:

- Reverse of the function computation of $f(n) = 2^n, n \geq 0$.
  - Given input 0^n, output should be 0^m where $m = 2^n$.
  - **Approach**: double the output length each iteration. [Example 9.10, Linz]

Example 1: $L = \{0^{2^n} \mid n \geq 0\}$

- Base case: ...B B 0 B B...
  - ➢ Consider $\delta(q, 0) = (q', B, R)$ and $\delta(q', B) = (q_f, B, R)$, i.e., see if we reached end of input after eliminating that 0.

    …BBq0BB… → …BBBq'BB… → …BBBBq$_f$B…

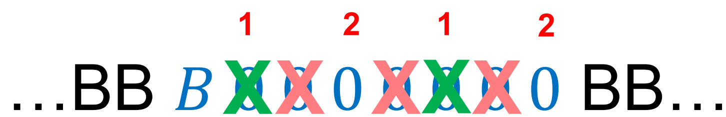
- Example: ...BB q00000000 BB...
  - After checking the first step: ...BB Bq'0000000 BB...; not getting to q_f.
  - ➢ Now, we mark off <u>every other</u> 0 from this point: <u>halves</u> the length.

    …BB B 0 ~~0~~ 0 ~~0~~ 0 ~~0~~ 0 BB…

  - ➢ After we hit a B on the rightmost, sweep left and start next iteration.

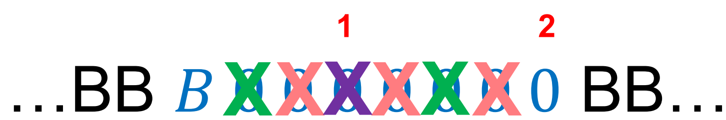
Example 1: $L = \{0^{2^n} \mid n \geq 0\}$

- Next Iteration



- Think about how can we do this type of "primitive" counting.

- Next Iteration



✓ Now (we know) we have the base case.

- For the TM, one more iteration before getting into q_f.

❖ If it was not an exponent of 2, then at some point, there will be a "first count" without its matching "second count".

Example 1: $L = \{0^{2^n} \mid n \geq 0\}$

- "Algorithm"
  1. Sweep left to right across the tape, crossing off every other 0.
  2. If in stage 1 the tape contained a single 0, *accept*.
  3. If in stage 1 the tape contained more than a single 0 and the number of 0s was odd, *reject*.
  4. Return the tape head to the left-hand end of the tape.
  5. Go to stage 1.

- Then figure out the more detailed descriptions, including how to design states and handle transitions from the algorithm.

Example 1: $L = \{0^{2^n} \mid n \geq 0\}$

Sketch (needs bit more details for HW):

- Need a transition to deal with the base case. If that's the end of input, then accept.
  - Use q_0 as initial and $\delta(q_0, 0) = (q_1, B, R)$ [1st count];
  - Use an intermediate q_6 to move to the left and check if everything is already marked off. Then go to q_f when read the leftmost B.

- Need a small cycle to deal with "mark-off" of the first count then try to reach the second count.
  - Use q_1 and q_2 as the two ends of the cycle with $\delta(q_1, 0) = (q_2, 0, R)$ [2nd count] and $\delta(q_2, 0) = (q_1, X, R)$ [1st count].

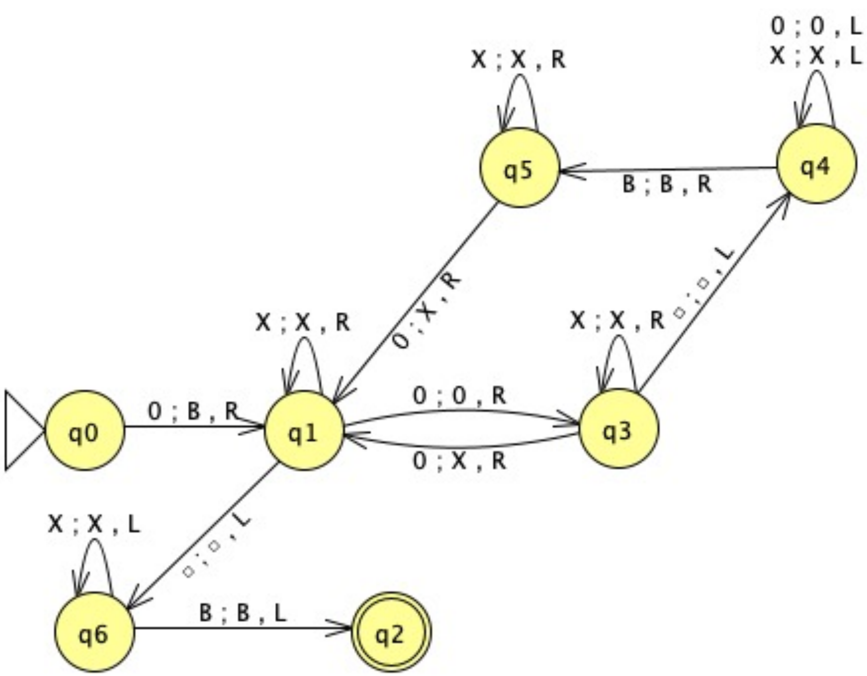
- Need an overall cycle to get back to the leftmost.
  - Something we've seen in other examples.

- Some transitions to skip the intermediate 0's and X's.

Example 1: $L = \{0^{2^n} \mid n \geq 0\}$

What we described:

- Exactly cuts length to half, but need to check at q6 before gets to q2. [Otherwise, 0^odd may gets accepted.]



- w = …B 000 B…

  B$q_0$000B → BB$q_1$00B

  → BB0$q_3$0B → BB0Xq_1B

  → BB0q_6XB → BB$q_6$0XB

Halt without accepting

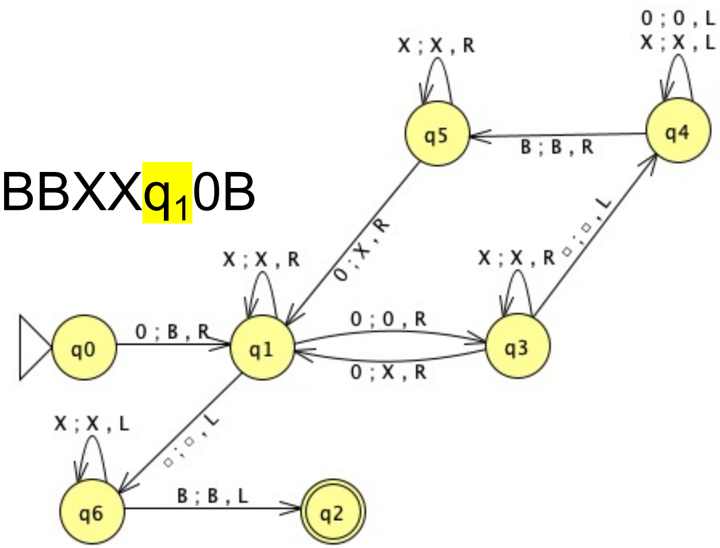
Example 1: $L = \{0^{2^n} \mid n \geq 0\}$

- w = …B 0000 B…

B$q_0$0000B → BB$q_1$000B → BB0$q_3$00B → BB0X$q_1$0B
→ BB0X0q_3B [First iteration]

BB0X0q_3B → BB0X$q_4$0B
…

Bq_4B0X0B → BB$q_5$0X0B → BBXq_1X0B → BBXX$q_1$0B
→ BBXX0q_3B [Second iteration]



BBXX0q_3B → BBXX$q_4$0B
…

Bq_4BXX0B → BBq_5XX0B → BBXq_5X0B
→ BBXX$q_5$0B → BBXXXq_1B [Third iteration]
→ BBXXq_6XB → … → Bq_6BXXXB
→ q_2BBXXXB

Halt and accepting

Example 2: $L = \{ww \mid w \in \{a, b\}^*\}$

- Not CFL, cannot be generated by CFG nor recognized by PDA.
  - Match symbols having same distances from the “mid-point”.
  - Or mid-point**s**, if we are dealing with, say, $\{a^i b^i c^j d^j\}$; etc.
  - red first, then blue, etc.

TOS

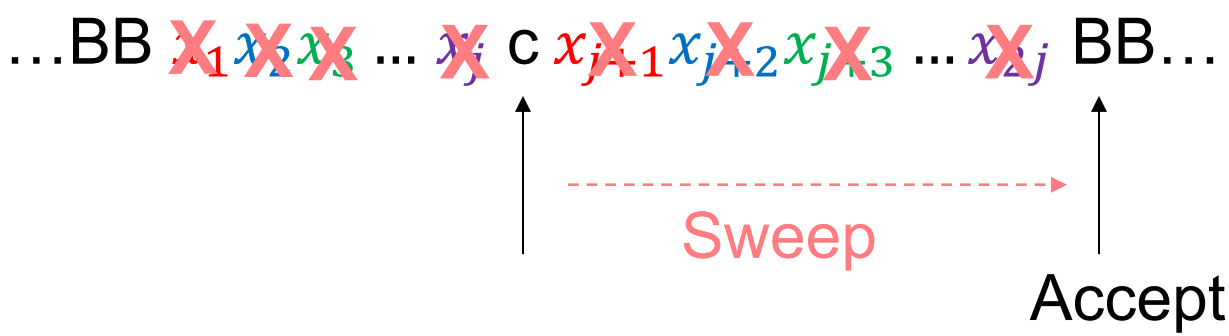
↓

… $x_1 x_2 x_3$ … x_j [“mid-point”] x_{j+1} … $x_{n-2} x_{n-1} x_n$ …

- $L_0 = \{ww^R \mid w \in \{a, b\}^*\}$
  - ✓ CFG and PDA
  - ✓ TM [Lecture and Video]: bounce back and forth using “same” approach
    - ➢ However, purple first, then green, etc., red last

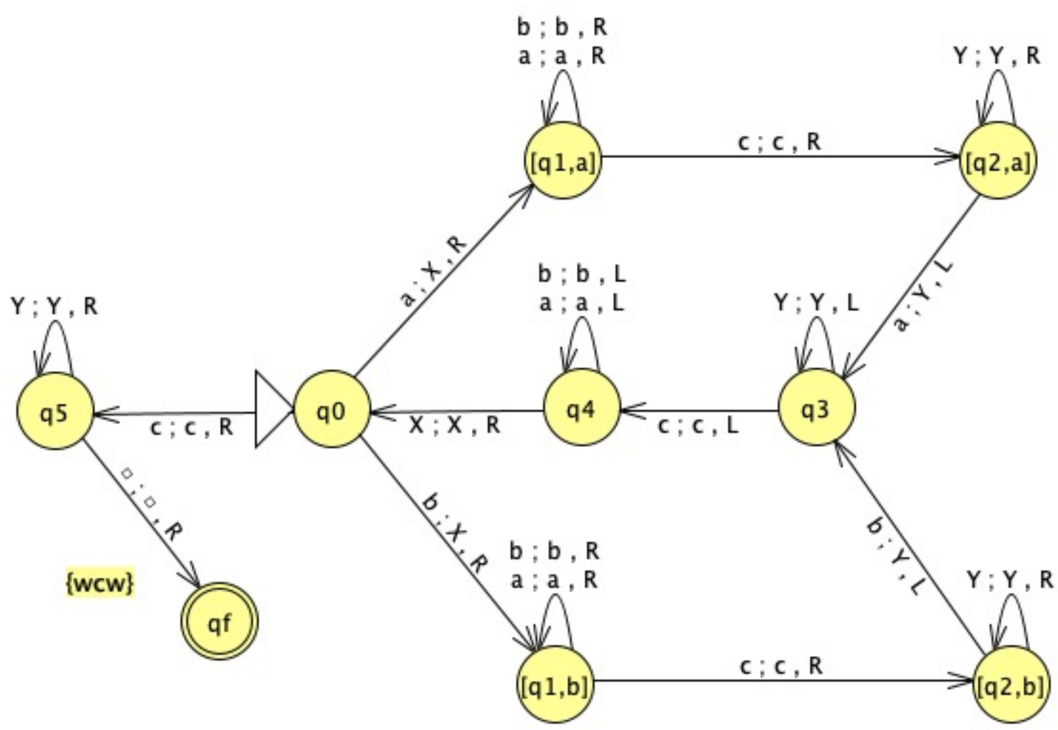
$L_1 = \{wcw \mid w \in \{a, b\}^*\}$

- For TM, we can match symbols through other ways.
  - Recall $L_1 = \{wcw \mid w \in \{a, b\}^*\}$: mark off current symbol with the first non-X symbol after the mid-point.



$L_1 = \{wcw \mid w \in \{a, b\}^*\}$

- Transition Diagram for the TM
  - Note that at q5 and qf in the diagram: need them to check unnecessary extra symbols in the second *w* portion.
  - Different from how to direct accept $\{ww^R\}$ from q0 to qf [Video].



Example 2: $L = \{ww \mid w \in \{a, b\}^*\}$

- Quite similar, but we do NOT know where the mid-point is.
- Any thought?

  1. Non-determinism? Will talk about NTM later.

  2. Using multiple tapes: lecture today.

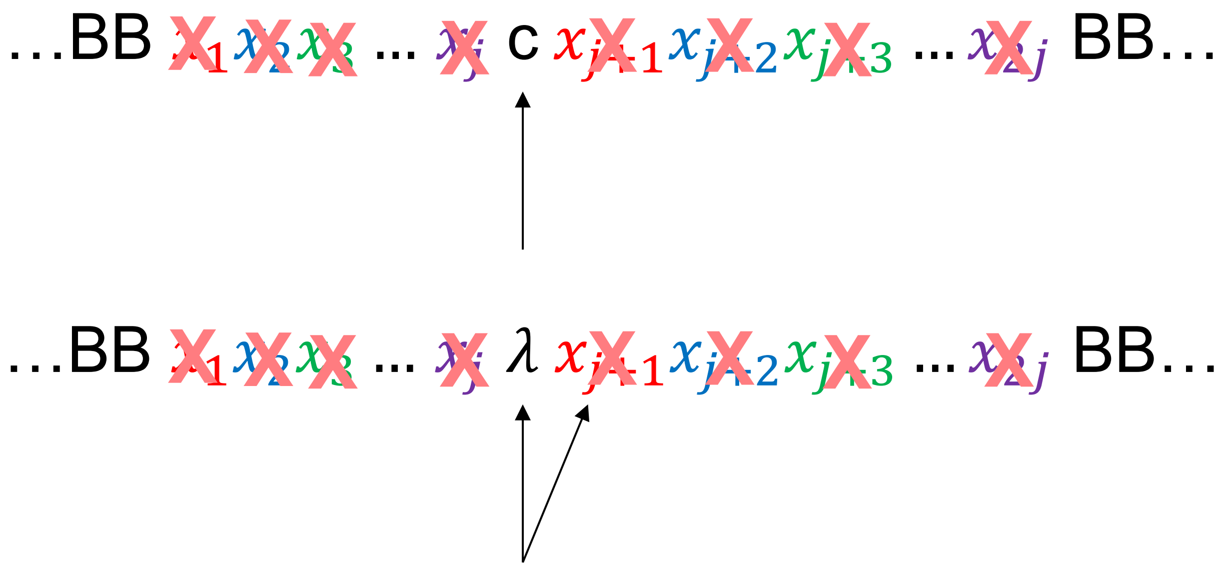
  3. Let's try to find the "mid-point", deterministically.

     But, rather, we differentiate the first and second w's.
     Then, we can apply similar matching approach.

- **Take-away**: Use a sequence of sub-TMs to "divide & conquer" the problem.

Example 2: $L = \{ww \mid w \in \{a, b\}^*\}$

- Differentiate the two portions



Not doable: again, no way to know any indicator to start with.
How can we achieve the same setup alternatively?

Example 2: $L = \{ww \mid w \in \{a, b\}^*\}$

- Differentiate the two portions and recover the first portion
- Example: ababaaababaa → XYXYXXXYXYXX

…BB XYXYXX λ XYXYXX BB…

Now identified the "mid-point"!

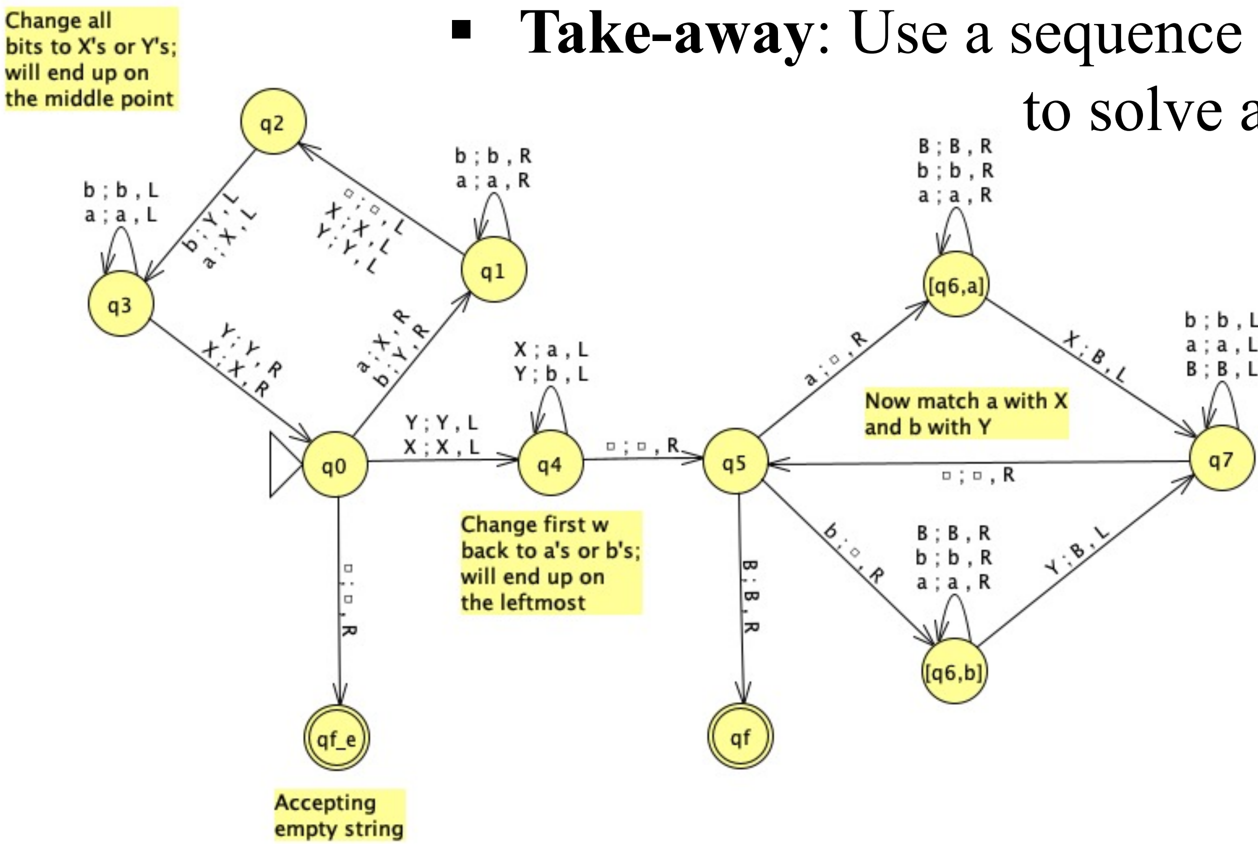
Can make the second portion different from the first by "recovering" the first portion.

I.e., ababaaXYXYXX

Then, we can try to match a with the first X we encounter, and b with the first Y we encounter, while skipping anything in between.

Example 2: $L = \{ww \mid w \in \{a, b\}^*\}$

- Try to come up with the "Algorithm" and its corresponding state and transition details.
- Here's the transition diagram, where B is diff. from *blank* (sq.)



- **Take-away**: Use a sequence of sub-TMs to solve a complex problem.

Example 2: $L = \{ww \mid w \in \{a, b\}^*\}$

- Again, using a different TM model, for instance with multi-track tape, the implementation can become easier and cleaner.

- Analogous to how changing the employed data structure (here tape structure) can make program more efficient and implementation more elegant.

- Will cover different TM models in lecture today.