Relationships between language classes

Claim: If $L$ is recursive, then $L$ is RE.

Claim: Let $L = \{ww \mid w \in \{0,1\}^*\}$. Then $L$ is recursive.

Claim: There is an RE language $L \subseteq \{0,1\}^*$ that is not recursive.

For $|\Sigma| \geq 2$, we have the following:

\[
\text{regular} \subset \text{CFL} \subset \text{recursive} \subset \text{RE}
\]

High-level TM Constructs

- Storage in finite-state control
- Multiple tracks
- Subroutines
- High-level language constructs: if, while

Example: There is a 2-track TM to accept $\{ww \mid w \in \{0,1\}^*\}$. 
while symbol on track 2 is $B$
  write 0 on track 2, move right
  scan right to $B$
  scan left across $1^*$ on track 2
  replace $B$ with 1 on track 2, move left
  scan left across $B^*$ on track 2, move right
while not reading $B$
  $c \leftarrow$ symbol on track 1
  write 0 on track 2, move left
  scan left to $B$, move right
  if symbol on track 1 is not $c$
    reject
  write $B$, move right
  scan right across $0^*$ on track 2
accept

Multitape Turing Machines

- $\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L,R\}^k$
- IDs are $(k + 1)$-tuples: $Q \times (\Gamma^* \#^*)^k$
- $\vdash$ is defined in the natural way
- $L(M)$ is the set of all strings $w$ such that
  
  $\vdash (q_0, \#w, \#, \ldots, \#) \vdash^* (q, \alpha_1 \# \beta_1, \ldots, \alpha_k \# \beta_k)$

  for some $q \in F$, $\alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_k \in \Gamma^*$. 
To simulate a $k$-tape TM with a TM:

- Use a $k$-track TM
- Record the state in finite control
- Record each $\alpha\#\beta$ on a separate track
- To simulate a single transition:
  - assume the head is to the left of every $#$
  - scan right, recording in finite control the symbol following each $#$ until all $k$ symbols have been recorded
  - scan left, updating each track

Nondeterministic TMs

- $\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}}$

To simulate an NTM with a 2-tape TM:

- Do a breadth-first search of all computations
- Use one tape as a FIFO queue containing IDs beginning and ending with $*$
- $\delta$ is stored in finite control
while the queue is not empty
        find state $q$ and current symbol $X$ in first ID
        foreach $(p, Y, D) \in \delta(q, X)$
            if $p \in F$
                accept
            copy first ID from queue to second tape,
                applying transition $(q, X) \rightarrow (p, Y, D)$
            move ID from second tape to end of queue
            overwrite first ID in the queue with blanks

Claim: Every CFL is recursive.

Proof sketch: Let $L$ be a CFL.

- Let $G$ be a CNF grammar such that $L(G) = L - \{\epsilon\}$.  
- We will construct a 2-track NTM $M$ such that $L(M) = L(G)$, and $M$ is guaranteed to halt.
- It is easy to modify $M$ to accept $\epsilon$ if $\epsilon \in L$.
- $M$ will nondeterministically find a derivation of input $w$.  

write \( S \) on track 2, move right

**while** current symbol on track 1 is nonblank

move left

**while** the current symbol is nonblank

nondeterministically move left or break

nondeterministically select production \( A \rightarrow \alpha \),

where \( A \) is current symbol on track 2

apply \( A \rightarrow \alpha \) to track 2, leaving tape head to the right of last nonblank on track 2

move left

**while** the current symbol is nonblank

verify that track symbols match or that there is a production \( A \rightarrow a \), where \( a \) is on track 1, \( A \) on track 2

move left

\( M \) accepts \( w \) iff \( w \in L(G) \).

Each iteration of the first loop either increases the number of nonblank symbols on track 2 or increases the number of terminals.

The first loop must eventually terminate.

Therefore, \( M \) always terminates.
Restricted Turing Machines

- Semi-infinite tapes
  - We can simulate a TM tape with a semi-infinite tape by “folding over”.

- Multistack TMs
  - We can simulate a TM tape with two stacks
  - The TM tape is “split” at the tape head position

- Counter machines