1 CFLs

note 1: $\Sigma = \{a, b\}$
note 2: $n_x(a)$ means the number of a's in the string x.

1.1 common seen CFGs

- $\{x|n_x(a) = n_x(b)\}$
  corresponding CFG :
  $$S \to aSbS|bSaS|\epsilon$$
- $\{x|n_x(a) \geq n_x(b)\}$
  corresponding CFG:
  $$S \to SS|aSb|bSa|a|\epsilon$$
  not surely right version :
  $$\begin{aligned} S &\to SaS|T \\ T &\to aTbT|bTaT|\epsilon \end{aligned}$$
- $\{x|n_x(a) = 2n_x(b)\}$
  corresponding CFG:
  $$S \to SaSaSbS|SaSbSaS|SbSaSaS|\epsilon$$
  using similar method, we can easily generalize this into k.
  an important lemma needed in the proof: substring $a^x ba^y$, $(x + y = k)$ can always be found in any non-empty string s in the language.
- complement of $a^n b^n$
  corresponding CFG:
  $$\begin{aligned} S &\to aSb|bY|Ya \\ Y &\to bY|aY|\epsilon \end{aligned}$$
- $\left\{a^i b^j | i \neq j \wedge i \neq 2j\right\}$
  corresponding CFG(not surely right):
  $$\begin{aligned} S &\to A|B|N \\ E &\to aEb|\epsilon \\ A &\to Eb|aAb|Ab \\ B &\to aaabb|aBb|aaBb \\ M &\to aaMb|\epsilon \\ N &\to aM|aNb|aN \end{aligned}$$
- $\left\{a^i b^j | i \leq j \leq 2i\right\}$
  With the idea of previous problem, it's easy to construt a CFG to recognize this language.

  But it's also elegant to use PDA to solve this problem.

  While the PDA reads an 'a', it randomly pushes one 'a' or two 'a's into stack and pops two 'a's while reading 'b'.And PDA accepts when stack is empty.

  Using this idea, it's also easy to show $\{x|n_x(a) \leq n_x(b) \leq 2n_x(a)\}$ is CFL.
- $\{xyz|x,\ z \in \Sigma^\star,\ y \in \Sigma^\star b \Sigma^\star\ and\ |x| = |z| \geq |y|\}$
  corresponding CFG (not surely right):
  $$\begin{aligned} S &\to A|B|C \\ A &\to EEEbEE|EAE|EEAE \\ B &\to EEbEEE|EBE|EBEE \\ C &\to EbE|ECE \\ E &\to a|b \end{aligned}$$

1.2 others

1.2.1 convert NFA/DFA into CFG

let D = $(Q, \Sigma, \delta, q_0, F)$ be a NFA, and WLOG, $F = \{p\}$.
let C = (V, T, R, S) to be the resulting CFG, and

- $V = Q$
- $T = \Sigma$
- $S = q_0$
- $\forall x, y \in Q, c \in \Sigma, y \in \delta(x, c)$, insert "$x \to cy$" into R
- insert "$p \to \epsilon$" into R

Also, one can construt a CFG D = (V', T', R', S') and

- $V' = Q'$
- $T' = \Sigma$
- $S' = p$
- $\forall x, y \in Q, c \in \Sigma, y \in \delta(x, c)$, insert "$y \to xc$" into R'
- insert "$q_0 \to \epsilon$" into R'

extention

- Using similar idea, it's obvious that if the rules in a CFG contain only transitions of type $A \to cB$ where $A, B \in V, c \in T^*$or only transitions of type $A \to Bc$, this CFG generates regular language.

  However if the rules contain both transitions of type $A \to cB$ and type $A \to Bc$, this CFG might generate CFL but regular language.

  For example, consider CFG

  $$S \to Bb|\epsilon$$
  $$B \to aS$$

  The language of this CFG may not be regular(hint: use pumping lemma of regular language).

  Note that there are some CFLs that cannot be recognized by this type of CFGs.

  To show that, first we need to formalize this type of CFG, $G = (V, T, R, S)$ R contains transitions of **both** type $A \to cB$ and type $A \to Bc$, while it also contains type of $A \to c$ where $A, B \in V, c \in T^*$.

  Moreover we can simplify this type into "standard type" with same ability, that R contains transitions of **both** type $A \to cB$ and type $A \to Bc$, while it also contains type of $A \to c$ where $A, B \in V, c \in T$, especially, only S could yield ϵ. (hint: implement Chomsky normal form).

  Pumping lemma for "standard type", for any string in language L whose length is larger than p, can be rewrited as $xyuvw$, $x, y, u, v, w \in T^*$ such that.

  1. $|xyvw| \le p$
  2. $|yv| \ge 1$
  3. $\forall 0 \le i, xy^iuv^iw \in L$

  (hint: consider parsing tree of "standard type")

  With this lemma, it can be shown that $\{a^nb^na^mb^m|n, m \ge 0\}$ cannot be recognized by "standard type".