CMSC 141 - Automata and Language Theory
Handout: Unrestricted Grammars
By: Gianina Renee V. Vergara

Unrestricted Grammars

- An unrestricted grammar, UG is also a 4-tuple (V, T, P, S)…
- Each production is of the form $\alpha \to \beta$, where $\alpha \in (V \cup T)^+$, $\beta \in (V \cup T)^*$
    - α must contain at least one variable

Example 1: Grammar for L = $\{a^n b^n c^n \mid n>0\}$

a. First, generate $a^n(BC)^n$

  S → aSBc
  S → aBc

b. Next, non-deterministically rearrange cB.

  cB → Bc

c. Finally, convert remaining variables to terminals, ensuring the string is in proper order.

  aB → ab
  bB → bb

Derivation	Production
S	
aSBc	S → aSBc
aaSBcBc	S → aSBc
aaaBcBcBc	S → aBc
aaaBBccBc	cB → Bc
aaaBBcBcc	cB → Bc
aaaBBBccc	cB → Bc
aaabBBccc	aB → ab
aaabbBccc	bB → bb
aaabbbccc	bB → bb

Example 2: Grammar for L = $\{a^{2^i} \mid i>0\}$

a. Use D to start duplication process.

  S → #aD

b. A will duplicate every terminal a.

  D → AD
  aA → Aaa

c. We get rid of A when it reaches the start of the string.

  #A → #

d. We get rid of D and # when we've completed all duplications we need.

  D → ε
  # → ε

Note: It is possible to find other UG for the same language.

Example 3: Grammar for L = $\{a^{2^i} \mid i>0\}$

  S → DaB
  D → DA
  Aa → aaA
  AB → B
  D → ε
  B → ε

Using Example 2:

Derivation	Production
S	
#aD	S → #aD
#aAD	D → AD
#AaaD	aA → Aaa
#aaD	#A → #
#aaAD	D → AD
#aAaaD	aA → Aaa
#AaaaaD	aA → Aaa
#aaaaD	#A → #
#aaaa	D → ε
aaaa	# → ε

Using Example 3:

Derivation	Production
S	
DaB	S → DaB
DAaB	D → DA
DaaAB	Aa → aaA
DaaB	AB → B
DAaaB	D → DA
DaaAaB	Aa → aaA
DaaaaAB	Aa → aaA
DaaaaB	AB → B
aaaaB	D → ε
aaaa	B → ε

Example 4: Grammar for L = { w∈{0,1}* | ww }

a. First, generate ww^R

S → WR
W → 0W0
W → 1W1
W → M

b. Carefully, reverse the order of characters in w^R. Create a reverse marker #

M → M#

c. The marker will take the terminal next to it towards R

#00 → 0#0
#01 → 1#0
#10 → 0#1
#11 → 1#1

d. We get rid of the marker when it reaches R. Move the terminal to the right of R.

#0R → R0
#1R → R1

e. Repeat (step b) until all terminals are to the right of R. Finally, get rid of M and R.

MR → ε

Example 5: Another Grammar for L = { w∈{0,1}* | ww }

a. We will use W to generate the string, and Z to mark the end of the string.

S → WZ

b. Mimic the first half of the input string, matching an A or B for every 0 or 1 respectively.

W → 0AW
W → 1BW
W → ε

c. Push all A's and B's to the right as 2nd half of the string.

A0 → 0A
A1 → 1A
B0 → 0B
B1 → 1B

d. When A's and B's are finally in order, use Z to convert each variable to a terminal.

AZ → Z0
BZ → Z1