CSci 435: Formal Languages and Automata

Instructor: Dr. M. E. Kim Name: Sam Dressler

**Home Assignment 4: 126/150 points + 10 points (optional)**

\*Q1. [20] For a given language L = {*anb****2n*** | *n* ≥ 0 is even}.

1. [8] Give a CFG that accepts L.

S -> A, A -> aaAbbbb | λ S🡪A is a unit production, redundant. S 🡪 aaAbbbb | λ is enough.

1. [6] Show the sequence of derivations for the acceptance of aaaabbbbbbbb by G in (1).

S-> aaAbbbb -> aaaaAbbbbbbbb -> aaaabbbbbbbb

* Is a symbol used to define a production rule, LHD 🡪 RHD.

To derive a sentential form from ‘S’, use ⇒ symbol,

i.e. S ⇒ aaAbbbb-⇒ aaaaAbbbbbbbb ⇒ aaaabbbbbbbb

1. A picture containing diagram

   Description automatically generated[6] Draw a derivation tree for aaaabbbbbbbb.

Q2. [20/ 30] Construct a CFG for the following languages where *n*, *m, k* ≥ 0.

1. [10] L1 = { *anbn* | *n* is a multiple of *3* }

S -> aaaSbbb | λ

1. [2/ 10] L2= { *anbmck* | *k* = *n+m* }

S -> A | B | λ A -> aABc | λ B -> bBc | λ

S ⇒ A ⇒ aABc ⇒ a(aABc)Bc ⇒ aaBcBc ⇒ aa(bBc)cBc ⇒ aabccBc ⇒ aabcc(bBc)c ⇒aabccbcc ∉L

1. [2/10] L3 = { *anbm* | *n =* *m –*1 }

S-> B | λ A-> aB | λ B-> Ab | λ S → *a*Sb | b

S ⇒ B ⇒ Ab ⇒ aBb ⇒ aAbb ⇒abb ∉L

1. [6/ 10, optional] L4 = { *anbmck* | *n=m* or *m* ≤ *k* }

S-> AB | C

A -> aAb | λ *anbn*

B -> cB | λ c\*

C -> aCc | D *ancn  From the rules of C and D, it generates anb\*cn, but it doesn’t guarantee that #(b)* ≤ *#(c )*

D -> bD | λ b\*

\*Q3. [10] Give the language L that is generated by the given grammar, in a formal expression.

S → *aa*S*bb* | SS |λ.

e.g.) L = { *w* ∈ {*a, b*}\* | *na*(*w*) = 2*nb*(*w*) }

From EX 5.4

L = {w ∈ (aa, bb) \* | i.e. |w| is even, na (w) = nb ­(w) and na(v) >= n­b(v) where v is any prefix of w}

\*Q4. [0/ 10] Find an s-grammar for L = {*anb****2n*** | *n* ≥ 2}.

S-> A A -> aAB B -> Bbb S🡪 A is a unit production, redundant.

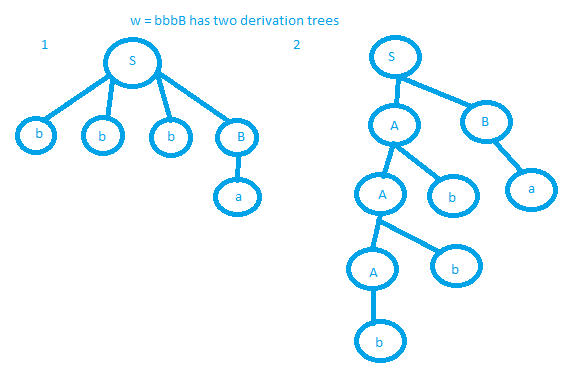
No terminal condition in any rule, i.e. your grammar never derives a string.

S → *a*S1B, S1 → *a*ABB, A → *a*ABB | b, B → *b*

\*Q5. [18/ 20] For a grammar G with the productions where G = ( {S, A, B}, {*a, b*}, S, P ) with productions

S → AB | *bbbB*, A → *b* | A*b*, B → *a..*

1. [8] Show that the grammar G is ambiguous.

example with string w = bbba has 2 derivation trees with the current grammar, G

1. [4/ 6] Give language L that is generated by G, L = L(G), in a formal expression (including a regular expression).

L = {(bna) | n >= ~~0~~ 1} i.e. bb\**a*

1. [6] Can you construct an unambiguous grammar that is equivalent to G? Otherwise, show that G is inherently ambiguous.

Unambiguous grammar:

S -> AB A -> b | Ab B -> a

Q6. [35] In the given grammar G, generate the simplified equivalent grammar by eliminating the following productions through (1) – (3).

G = ( {S, A, B, C}, {*a, b*}, S, P ) with productions

S →*b*AA | *b*B, A → *a*A| *aaC* , B → *bb*B | *λ,* C → A

1. [10] Eliminate the λ-productions

Vn = {B} – the set of productions containing lambda productions

S -> bAA | bB | b

A -> aA | aaC

B -> bbB | bb

C -> A

1. [10] Eliminate the Unit-productions from (1)

Only 1 unit-production in the form A-> B where A, B ∈ V: C -> A, productions involving C->A are not needed.

S -> bAA | bB | b

A -> aA | aaA

B -> bbB | bb

1. [10] Eliminate the useless productions (2), so that give the simplified equivalent grammar.

1 useless production since a terminal cannot be reached by productions of A, we can remove it.

S -> bB | b

B -> bbB | bb

1. [5] Give the language L that is generated by this grammar, L = L(G), in a formal expression (including a regular expression).

L = {b(bb)\*}

i.e. w = {b, bbb, bbbbb, bbbbbbb, …}

Q7. [15] Convert the given grammar into Chomsky Normal Form (CNF).

S → AB | *a*B, A → *abb* | *λ* , B → *bb*A

Hint: Eliminate the λ-productions and/or any unit-production prior to their conversion into CNF.

S->AB | aB

A-> aab | λ

*Remove λ-productions*

S-> AB | aB | B

A -> aab

B -> bbA | bb

*Remove unit productions*

S -> AB | aB | bb | bbA

A -> aab

B-> bbA | bb

*Convert to CNF*

S-> AB | XB | YY | YW

A-> XQ

B -> XZ | YY

Q -> XY

X -> a

Y -> b

Z -> YA

Q8. [8/10] Convert the given grammar into Greibach normal form.

S → *a*S*b* | *ab* | *bb*

S -> aSB | aB | bB

~~A -> a~~ Your ‘A’ is useless because it’s never reachable from S.

B -> b