Konrad Bartlett

0580964

COIS 3050

Assignment 1

1. **Give a recursive definition for the language WEIRD which is defined over the alphabet**

**∑ = {a, b} and contains words that have an odd number of b’s and an even number of a’s.**

Rule 1: b ϵ WEIRD

Rule 2: if W ϵ WEIRD, then aaW, aWa, Waa, bbW, bWb, Wbb ϵ WEIRD

Rule 3: No other words belong to WEIRD

1. **Consider the following recursive definition of 4-PERMUTATION:**

**Rule 1: 1234 is a 4-PERMUTATION.  
Rule 2: If wxyz is a 4-PERMUTATION, then so are zxyw and yxzw.  
Rule 3: no other words belong to 4-PERMUTATION.  
Give all the words in the language 4-PERMUTATION.**

Rule 1: 1234 ϵ 4-PERMUTATION

Rule 2: Use 1234, w = 1, x = 2, y=3, z=4.

Then: 4231, 3241 ϵ 4-PERMUTATION

Rule 2: Use 4231, w = 4, x = 2, y = 3, z = 1.

Then: ~~1234,~~ 3214 ϵ 4-PERMUTATION

Rule 2: Use 3241, w = 3, x = 2, y = 4, z = 1.

Then: 1243, 4213 ϵ 4-PERMUTATION

Rule 2: Use 3214, w = 3, x = 2, y = 1, z = 4.

Then: ~~4213~~, ~~1243~~ ϵ 4-PERMUTATION

Rule 2: Use 1243, w = 1, x = 2, y = 4, z = 3.

Then: ~~3241~~, ~~4231~~ ϵ 4 -PERMUTATION

Rule 2: Use 4213, w = 4, x = 2, y = 1, z = 3.

Then: ~~3214~~, ~~1234~~ ϵ 4-PERMUTATION

Rule 3: No more words can be created by applying rule 1, rule 2, or rule 3.

Therefore: 1234, 1243, 3214, 3241, 4213, 4231 ϵ 4-PERMUTATION

1. **Construct a regular expression defining each of the following languages over the**

**alphabet {a, b}.**

#### (a) L = {aab, ba, bb, baab};

(aab + ba + bb + baab)

* Word 1 or Word 2 or Word3 or Word 4

#### (b) The language of all strings containing exactly two b's.

a\* b a\* b a\*

* Any amount of a’s, exactly two b’s, any amount of a’s

#### (c) The language of all strings containing at least one a and at least one b.

b\* (a\*a) (b\*b) a\*

* Any amount of b’s, any amount of a’s, at least one a, any amount of b’s, at least one b, any amount of a’s

#### (d) The language of all strings that do not end with ba.

(a+b)\* b

* Any amount of a or b, always end with b so that word does not end with ba

#### (e) The language of all strings that do not containing the substring bb.

a\* a\*+ b a\*

* Any amount of a’s, any amount of a’s or a b, any amount of a’s

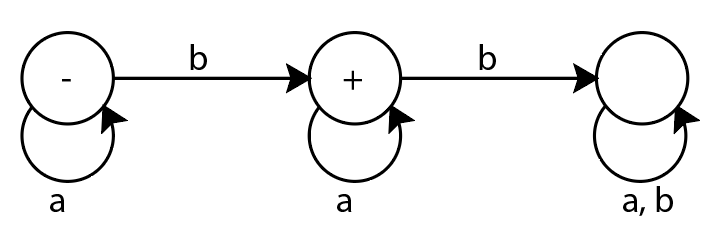
#### (f) The language of all strings in which every b is followed immediately by aa.

a\* baa\*

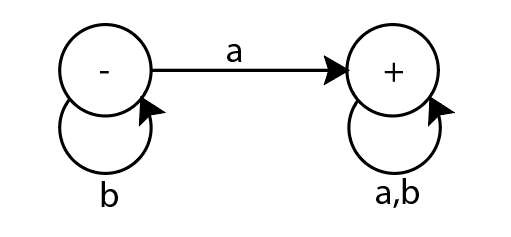
* Any amount of a’s, any amount of b’s that are followed immediately by aa

1. **For the alphabet S = {a, b}, construct an FA that accepts the following languages.**

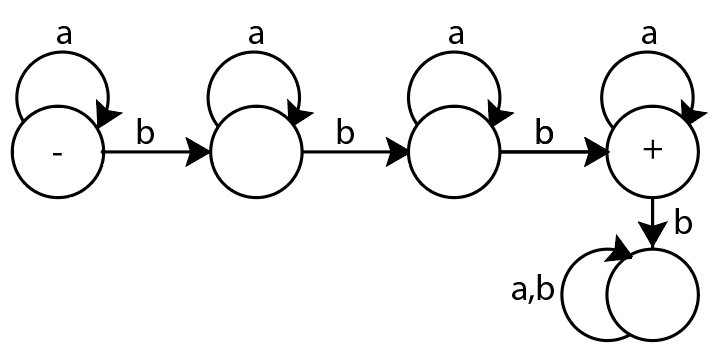
#### (a) L = {all strings with exactly one b}.



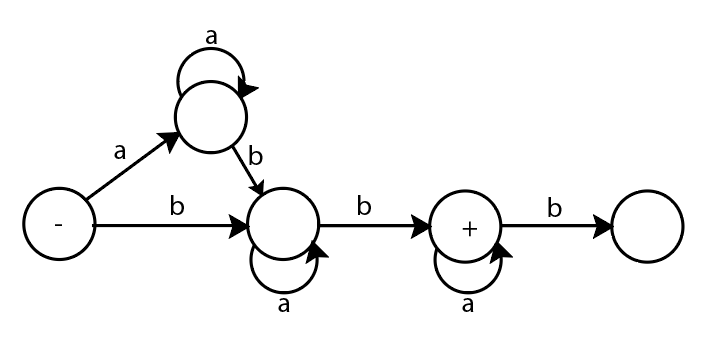
#### (b) L = {all strings with at least one a}.



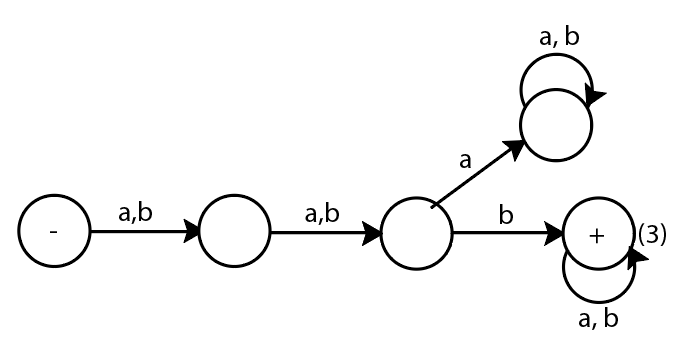
#### (c) L = {all strings with no more than three b's}.



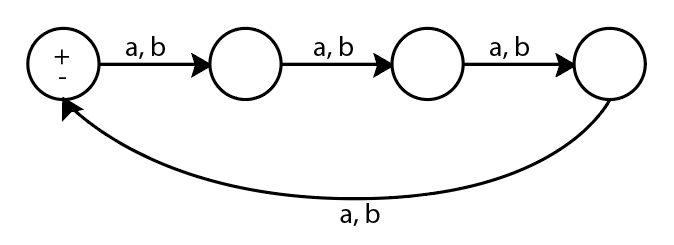
#### (d) L= {all strings with at least one a and exactly two b's}



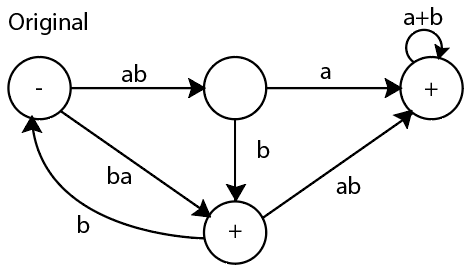
#### (e) L= {all strings with b as the third letter}

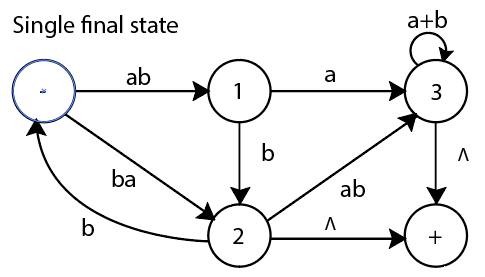


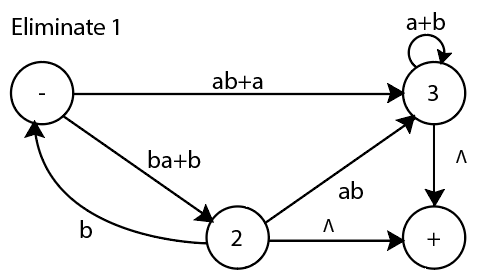
#### (f) L={w, |w| mod 4 = 0} // the cardinality of the word is a multiple of 4

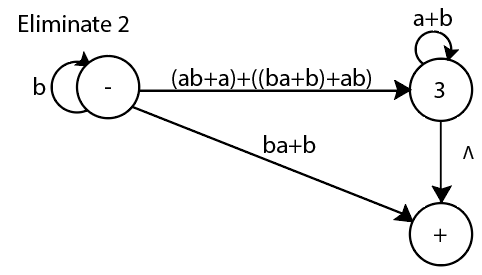


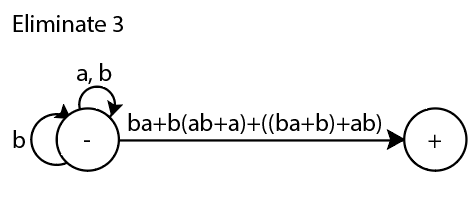
1. **Using the algorithm provided by Kleene’s Theorem, convert the following TG into a Regular Expression (show your work).**

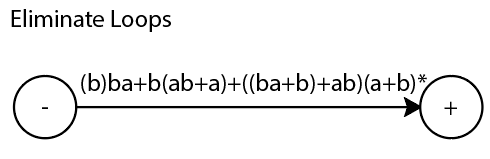












**Regular expression: (b) ba+b(ab+a)+((ba+b)+ab)(a+b)\***