Linguistics 570 Take Home Final

Dec 7, 2011

Deadline: Dec 14, 2010 5:15 PM

EBA 321

1 Groups

1.0.1 Consider multiplication mod 10 ($*_{10}$) and the algebra

$$\langle *_{10}, \{0, 1, 2, \ldots, 9\} \rangle$$

Show that this is not a group.

1.0.2 Find a 4-element subalgebra that is a group. Show the Cayley table and identify the inverses in each row by circling them. Verify the group axioms (you may assume multiplication mod 10 is associative.

1.0.3 Is this only 4-element subalgebra that is a group? What are the others if there are any?

1.0.4 Are there are smaller subalgebras that are groups?

2 Lattices

The questions in this section all consider the poset in Figure 2.

2.0.1 Is the poset a lattice? If not, is it a semi-lattice? If so, what kind, join or meet? For each no answer give a justification.

2.0.2 What are the upper bounds of p and q? What is the least of these upper bounds? Or is there a least upper bound?

2.0.3 What are the upper bounds of p and $p \vee q$? What is the least of these upper bounds? Or is there a least upper bound?

2.0.4 What is the least upper bound of p and $p \wedge q$?

2.0.5 What is the greatest lower bound of p and $p \vee q$?

2.0.6 What is the greatest lower bound of p and $p \wedge q$?

2.0.7 What is the greatest lower bound of p and q?

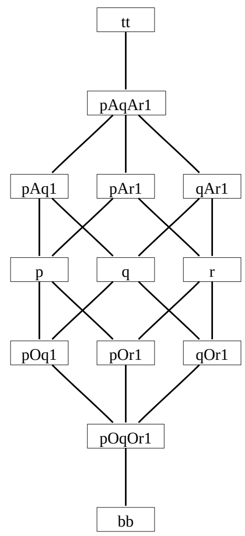

Figure 1: A logic poset

2.0.8 Define the ordering relation for this lattice using $\rightarrow$ (logical implication as in $p \rightarrow q$). The upper end of the diagram is greater, the lower end lesser. Your definition should be in the following form:

$$a > b \text{ iff } \text{ some expression involving } a,\, b, \text{ and } \rightarrow$$

Your definition should be consistent with the definition of the poset in the diagram. For example, it should make $p \vee q \vee r$ be greater than $q \vee r$, because that's what the diagram says. Your interpretation of $\rightarrow$ should be consistent with logical implication. Here are some examples of logical truth that use $\rightarrow$:

$$\begin{array}{l} p \rightarrow p \\ p \rightarrow (p \vee q) \\ (p \wedge q) \rightarrow p \\ (p \wedge q) \rightarrow q \end{array}$$

2.0.9 Assume that if $p \rightarrow q$, then p is more informative than q. Which direction in the poset represents getting more information, going upward or going downward? Why?

2.0.10 Which point in the lattice is better thought of as representing contradiction, $\perp$ or $\top$? Why?

3 Finite State Languages

Warm ups first

3.1 Regular expression, fsa's, and Type 3 grammars

3.1.1 Draw state diagrams for fsa's corresponding to the following regular expressions:

(a) aba^*b

(b) $(aba^*b)^*$

(c) $(aba^*b)^*b$

(d) $b(aba^*b)^*$

(e) $b(aba^*b)^*\{a, b\}^*$. Note: This is just the concatenation of $b(aba^*b)^*$ with the Kleene closure of $\{a,\ b\}$, but there is a simpler machine than this description suggests.

3.1.2 Consider the automaton in Figure 2 (Green meas start state, red meqns stop state). Call the language accepted by this automaton L.

(a) Which of the following strings is in L?

bbbb,abaabab,abaabb,bbbaabbbb

(b) What is the smallest string x in L such that the number of b's in x is evenly divisible by 3? How many b's does the next smallest such string contain?

Hint: Be sure and read the intro to the Pumping Lemma in your textbook, starting p. 470.

3.2 Pumping Strings

The Pumping Lemma for finite-automaton languages (**fal**s) states:

> If L is an infinite fal (Finite Automaton Language, synonym of regular language) over the alphabet Σ, then there are strings $x, y, z \in \Sigma^*$ such that $y \neq e$ and $xy^nz \in$ L for all $n > 0$. These strings are called **pumping strings**. (Note: I've changed the formulation in the book very slightly, using $>$ instead of $\geq$; both are right).

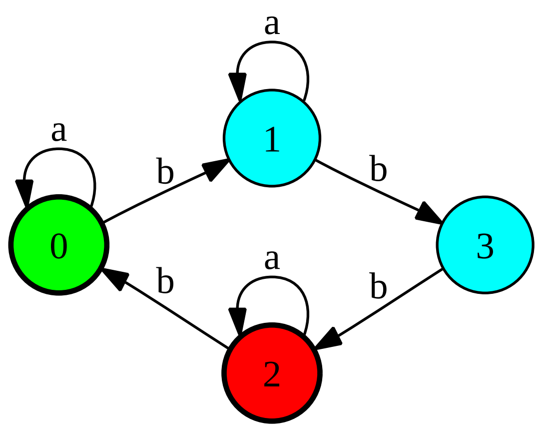

Figure 2: A sample FSA

Here's an example of an fal and a pumping string. Let:

$$\text{L} = b^{+}c^{+}d$$

Then

$$bcd$$

is an example of a pumping string for L, where:

e	b	cd
x	y	z
prefix	repeating part	suffix

Why? Because we can increase the number of bs as much as we want, and we still get a member of L:

$$\left.\begin{array}{ll} b & cd \\ bb & cd \\ bbb & cd \\ bbb\ldots bbb & cd \end{array}\right\} \in \text{L}$$

Thus bcd satisfies the requirements of the lemma:

$$b^{n}cd \in \text{L for all } n > 0.$$

Note that in choosing a pumping string, either the prefix or the suffix or neither or both can be an empty string. In this case the prefix is. The only part the lemma says can't be an empty string is the repeating part, y.

Note also that bcd is a pumping string in another way:

b	c	d
x	y	z
prefix	repeating part	suffix

because:

$$bc^n d \in \text{L for all } n > 0.$$

Problem: Produce pumping strings for the following infinite regular languages. In each case identify the prefix, the suffix, and the repeating part.

3.2.1 $(ab?a)^+$

3.2.2 $a^*ba^*(ba^*ba^*)^*$

3.2.3 a^*ba^*b

3.3 Finite-state Pumping Lemma and Copy language

Consider the **copy language** L:

$$\Sigma = \{a,\, b,\, c\,\}$$
$$\text{L} = \{xx \mid x \in \Sigma^*\}$$

That is, L consists of strings such that the second half is an exact copy of the first half. Examples of strings in L:

abcabc
abbacabbac
cabbbcabbb
cbbacbba

Examples of strings NOT in L:

abbacabbbac Note: 3 b's on right, 2 on left
acbbbcabbb Note: Second half has *ca* not *ac*

3.3.1 Show that L is not an fal using the pumping lemma.

Wait! You better read this hint! You can't do this directly. Why not? Because there ARE perfectly good pumping strings in L. For example:

$$bb$$

is a pumping string, because for every value of n,

$$(bb)^n \in \mathrm{L}.$$

In this case both the prefix and suffix are empty. The Pumping Lemma says, If it's fal, it has a pumping string. It doesn't say: if it has a pumping string, it's an fal. In fact, L isn't an fal, but you can't prove by saying there no pumping strings.

So you need another strategy. We'll call it the intersection strategy. Instead of proving L has no pumping strings you're going to prove a related language L' has no pumping strings. This proves L' is not an fal. Yay! Wait, we're not done yet. The problem is to prove L is not an fal! So how did this little digression into L' help?

What you need to do is choose L' carefully. We choose it so there is a known fal R whose **intersection** with L is L'. That is:

$$\mathrm{L'} = \mathrm{R} \cap \mathrm{L}$$

Why? Because the fals are closed under intersection. And if L' is not an fal, then since R is an fal, L is not an fal.

Damn! This just got a whole lot harder.

Strategy hint. L' will contain only copy strings since all its elements are in L. To get an L' without pumping strings, the idea is to contrive R so as to yield as tightly constrained an L' as possible (while still being infinite!). Notice for example that if we use R_0

$$\mathrm{R}_0 = a^*b^*a^*b^*$$

the intersection with L is still too big because it still leaves us pumping strings like the one above, $(bb)^n$. Notice, too, when applying the pumping lemma, that you've reduced the problem to proving that L' is not regular. So when pumping your candidate pumping strings, the question is not whether they stay in L; the question is whether they stay in L'. But now try

$$\mathrm{R}_1 = a^+b^+a^+b^+$$

Notice that R_1's intersection with L is:

$$\mathrm{L'}_1 = \mathrm{R}_1 \cap \mathrm{L} = a^n b^m a^n b^m$$

Does $\mathrm{L'}_1$ have any pumping strings? If not, then the outline of the proof that L is not an fal is clear:

(a) Consider R as above, which is clearly an fal language.

(b) Show that $\mathrm{R} \cap \mathrm{L} = \mathrm{L'}_1$ has no pumping strings.

(c) It follows that $\mathrm{L'}_1$ is not an fal.

(d) But then it follows that L is not an fal, since the fal's are closed under intersection and R is an fal and

$$\mathrm{L'}_1 = \mathrm{R} \cap \mathrm{L}$$

Flesh out this outline of a proof in prose.

3.4 Mirror language

Consider the mirror language L:

$$\Sigma = \{a,\ b,\ c\}$$
$$\mathrm{L} = \{x \mid x \in \Sigma^* \text{ and } x = x^{\mathrm{R}}\}$$

Examples of strings in L:

abccba
abbaccabba
cabbbbbbbbac
abcccba

Examples of strings NOT in L:

abbaccabbba Note: 3 b's on right, 2 on left
acbbbbbbbbac Note: First two letters are not the mirror image of last 2

3.4.1 Find a grammar that generates exactly the mirror language L. The grammar does not have to be a finite-state grammar. That is, it can use center embedding.

3.4.2 Show that L is not an fal using the pumping lemma.