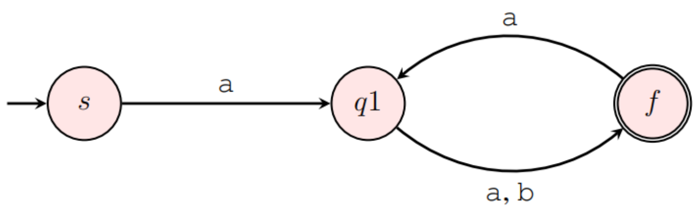
Lab 4

You are fine to work together with others.

Consider a deterministic finite automata (DFA) represented by the following digraph:



DFA's are one of the simplest computation models that can be designed to accept languages with words matching description with regular expressions. For example, the above DFA would accept "words" in the set:

$$\{\text{aa, ab, aaaa, aaab, abaa, abab, …}\}$$

Computation begins at the state node labelled "s". An arc can only be traversed if the next character in the word matches the label on the arc. Each character in a word is consumed as an arc is traversed. Characters are consumed from left to right. If the last state node that is reached is labelled with "f", then the word is an accepted word and belongs to the language generated by the DFA. If computation does not have any next arc that can be transitioned, or the input word is consumed and the last state node is not labelled "f", then the word is rejected.

Do not worry about programming the above. I have already programmed DFAs in general with the code provided in `dfa.hs`. The example DFA above is also encoded with variable `m` and final state `f`. You have the function `genLanguage` that takes DFA `m` (or any other) and integer number, say 4, and generates the set of words accepted by `m` of length 4.

1) Write a function that replaces each instance of character 'a' with the word "bark " (with a space at the end) for a given input string. Also, write a function that replaces each

instance of the character ‘b’ with “meow “ (with a space at the end) for a given input string. You are welcome to write one function that does the job of both of the above.

2) Write an expression using a combination of Monoids and mapping applied to the set of strings that belong to the given DFA of length 10, so that each word is converted to a String with “bark ” and “meow “ substrings. The result should be a list of strings again.

Do not modify the code given. So, sometimes with a DFA, you may recognize a pattern for strings you would like to create, but simplify the DFA so that it is less complex to work with. Perhaps there are many functions on languages like this that could be reused instead of designing a different DFA for each language you want.

This lab is out of 5 marks total. There are 2 marks for each of Question 1, and 3 marks for Question 2. Submit a `lab4.hs` file to Blackboard in the Assignments and Tests section under Lab 4 submission. Deadline for submission is end of class, Nov 2, before 2:20 pm.