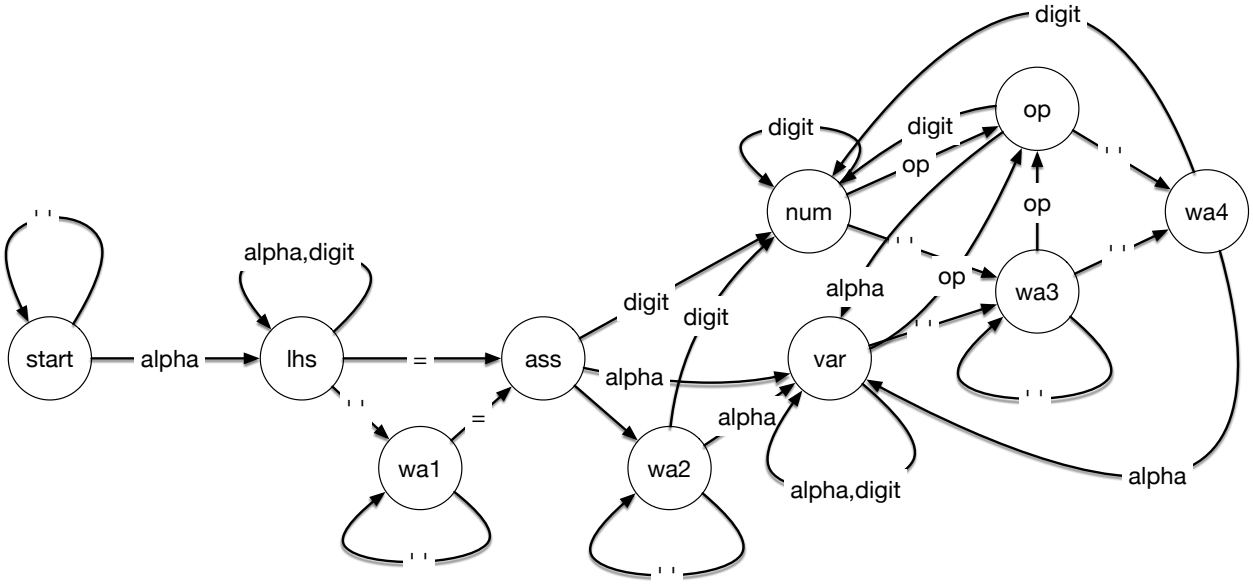
Programming Assignment

Finite automata can be used to "tokenize" input strings. We are using them to tokenize assignment statements in Python, where a variable (starts with alphabetic character followed by alphabetic characters or digits or the underscore character) are assigned to an expression consisting of variables, numbers, and binary operands.

The finite automaton processes, as is usual, one character at a time. In certain states, it emits the current token. Here is an automaton that captures these simple assignment operations.



You will need to figure out when to emit a token. Here is the beginning of my implementation:

```
digits = '0123456789'
alpha = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_'
ops = '+_/*'

def process_string(string):
    tokens = [ ]
    state = 'start'
    current_token = []
    for character in string:
        if state == 'start':
            if character in alpha:
                current_token.append(character)
                state = 'lhs'
            elif character in ' ':
                state = 'start'
        elif state == 'lhs':
            if character in alpha or character in digits:
                state = 'lhs'
```

```
            current_token.append(character)
        elif character in ' ':
            state = 'wa1'
            tokens.append(''.join(current_token))
            current_token = []
        elif character == '=':
            state = 'ass'
            tokens.append(''.join(current_token))
            current_token = []
        else:
            print('error')
            break


    return tokens
```

You will need to write a number of "unit tests", where you apply your tokenizer function on various simple assignment statements. Notice that the processing of blank spaces in this set-up is far from efficient.