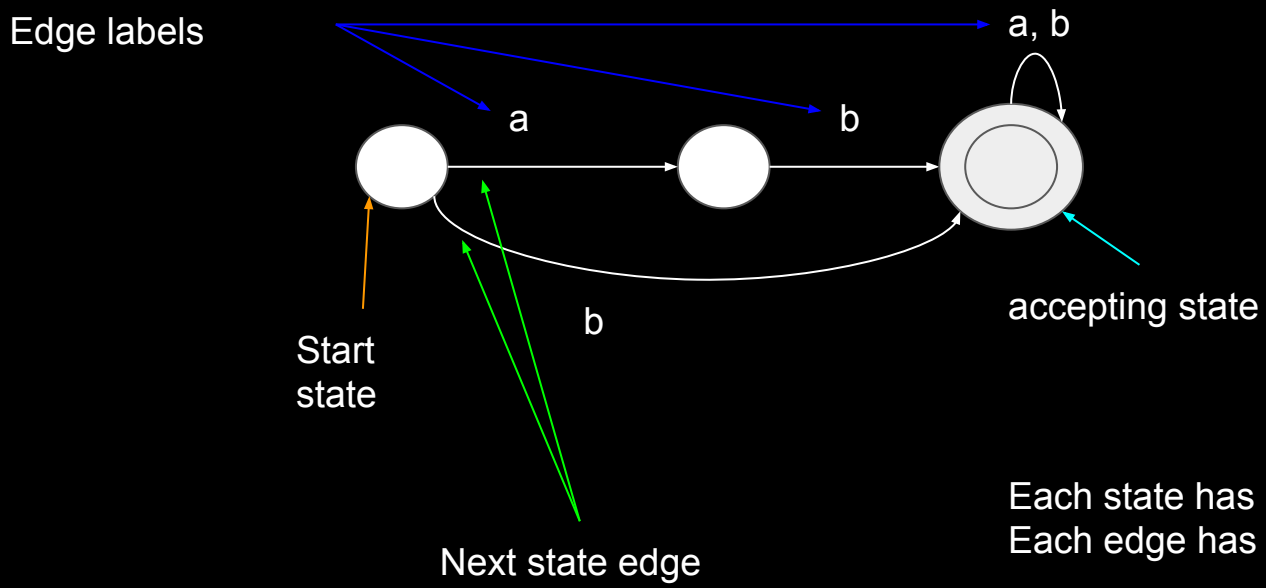
“Contextual State Charts”

David Tauraso
Graduate of Sonoma State University

10/15/19

Here is your regular state machine
It recognizes the regex (ab|b)(a|b)*



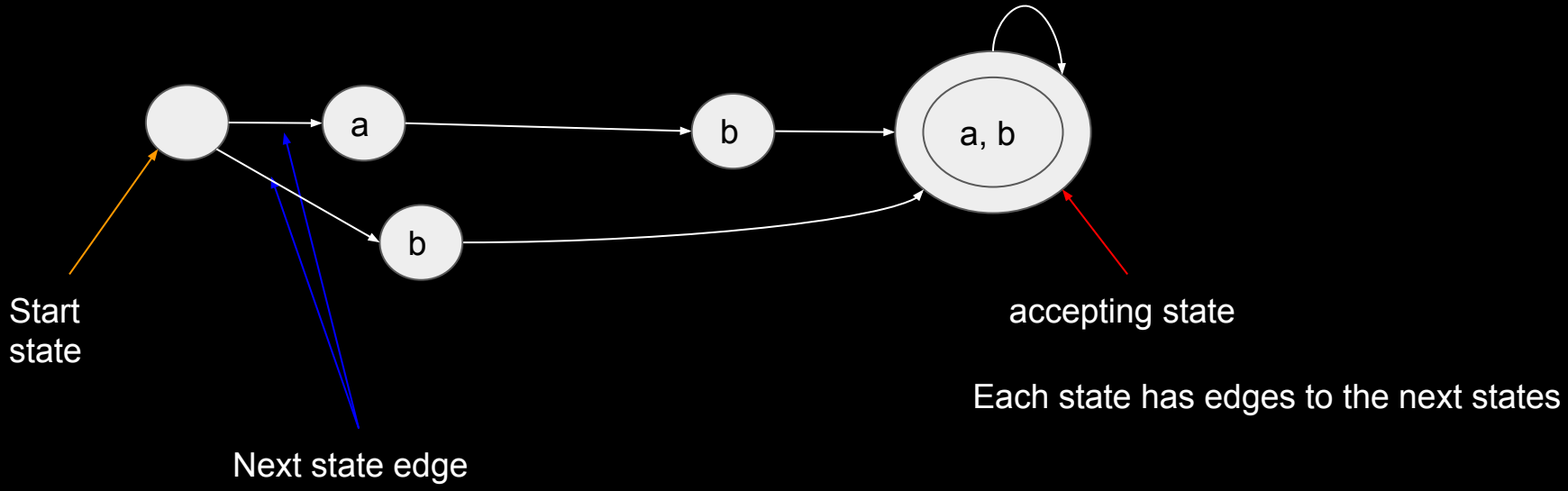
Each state has edges to the next states
Each edge has a label on it

New idea for a state

What if the edges are also states?

New idea for a state machine

What if the edges are also states?
It recognizes the regex (ab|b)(a|b)*



Each state has edges to the next states

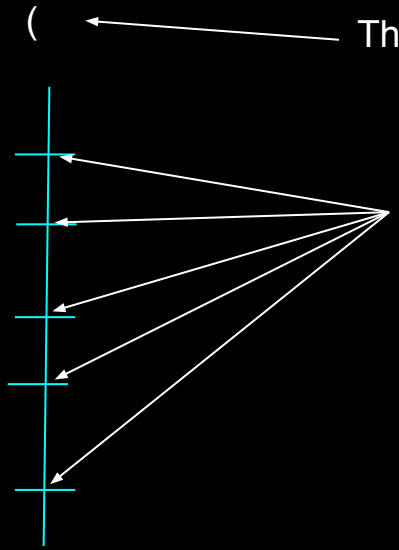
What if the edges are also states?

What if each state runs a function, and the function doesn’t necessarily have to do a character check?

What if you could visit the same state at different times and do something different each time?

New idea for a state

The state name is a sequence of strings(each string can have spaces)



The different possibilities of what the second string can be

You can make as many second strings as you want

Examples of first string, second string:

(, one
(, a different string
(, 3
(, the 4th string

Definition of the state

Name
- This is a sequence of strings

Next states
- Treat them like if else if else cases

Children
- States in the below level
- Treat them like if else if else cases in the below level

Parent
- Metadata for my state machine algorithm to work

Functions
- 1 function for each (first name, second name) pair
- The function doesn’t necessarily have to do a character match

Just as a note:
- The state machine is a graph defining the flow of control. You will run a function when you visit the state

New idea for a state

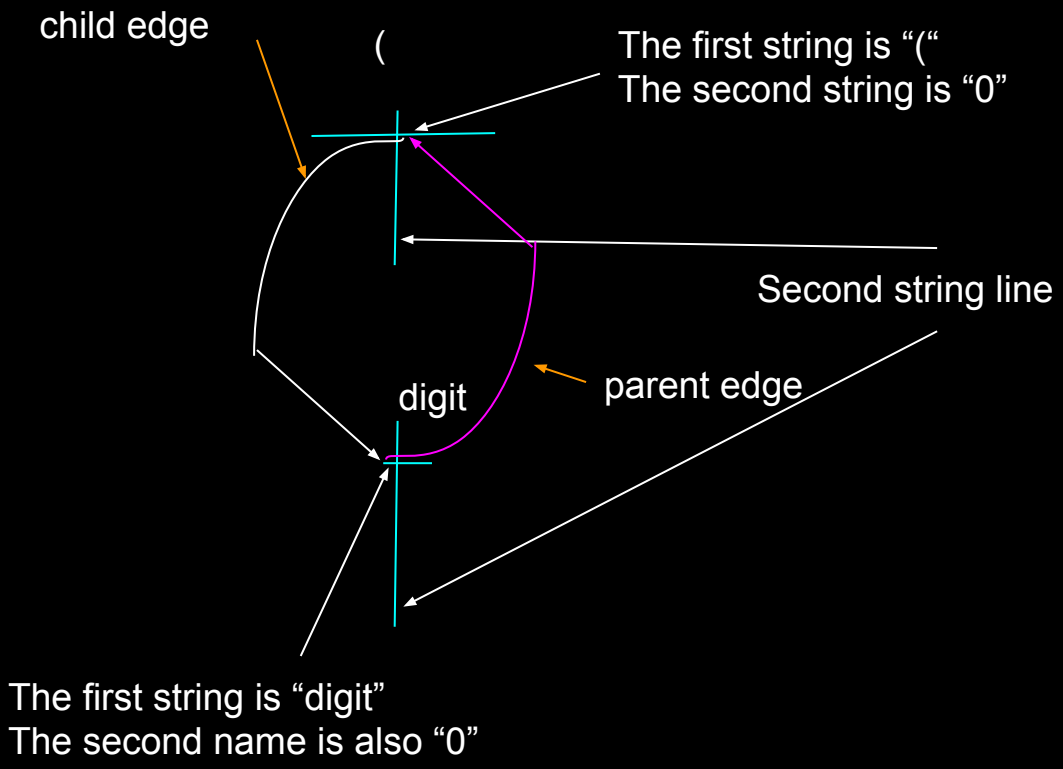
What if the edges are also states?

What if each state runs a function, and the function doesn’t necessarily have to do a character check?

What if you could visit the same state at different times and do something different each time?

What if you wanted a hierarchy of states?

New idea for a state(the functions are not shown)

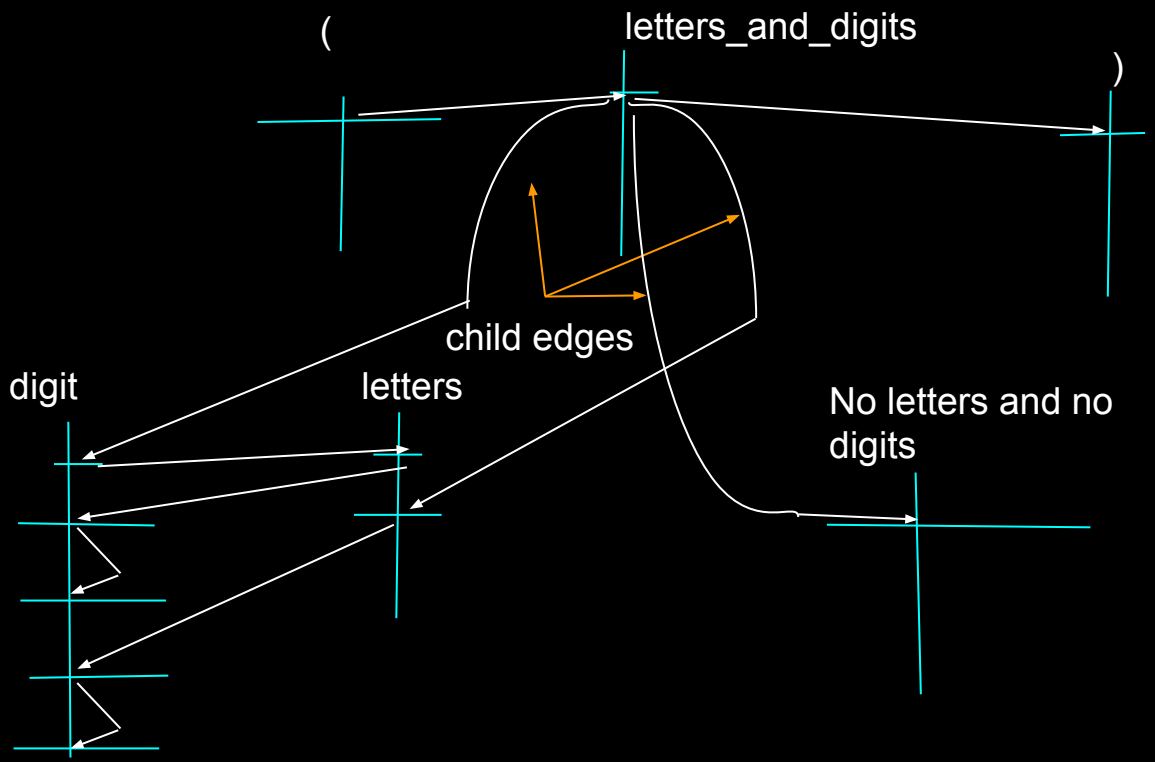


Each (first string, second string) can run a function

The function always returns true or false

The function can run any code before returning true or false

New idea for a state hierarchy



We want to recognize the below sequences
(digit letters digit digit)
(letters digit digit)
( )

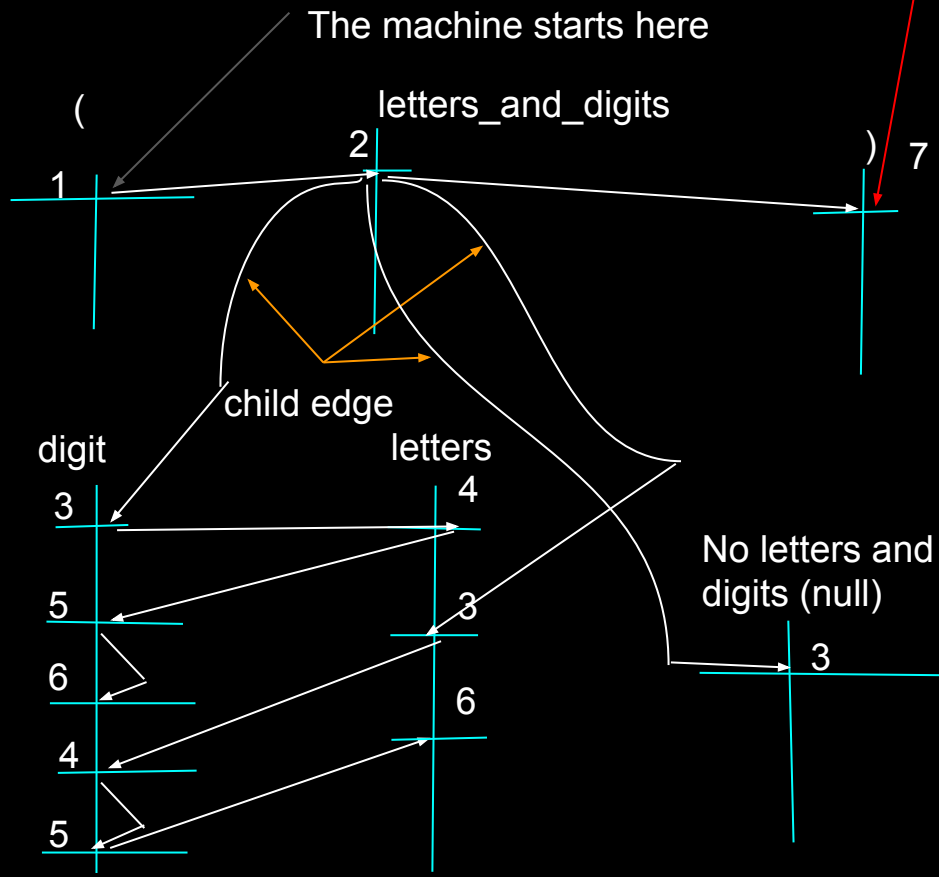
Let’s see an example.

Here is the order the states are run in

The machine ends at this state because it is the last topmost state run

We want to recognize the below sequences

(digit letters digit digit)
(letters digit digit letters)
( )



We treat the sequences like this:
(letters_and_digits)

letters_and_digits then turns into

digit letters digit digit or
letters digit digit letters or
null

Here is the layout of the code structure for a state

```
Name (first string)

      Next
            (second string)
            0
                  [‘another_first_string’, ‘another_second_string’], [‘another_first_string_2’,
            ‘another_second_string_2’]

            names\’s second context
                  [<a [state, case] would go here>]

      Children
            (second string)
            0
                   [‘a child’, ‘the child\’s case’]

      Functions
            (second string)
            0
                  functionName
            names\’s second context’
                  anotherFunctionName
```

Let’s see the actual code for 1 state

Quick notation for the below example: (first name, second name)

This is what the syntax looks like for 1 state with 2 cases and 1 child

```
['name', [
      ['next',
            [['0', [[‘first_name’, ‘second_name’], [‘first_name_2’, ‘second_name_2’]]]
            [‘names\’s second context’, [ <a [first name, second name] would go here> ]]       ]]],
      ['children',
            [['0', [ [‘the child’s first name’, ‘the child\’s second name’] ] ]]],
      ['functions',
            [['0', functionName ],   [‘names\’s second context’, anotherFunctionName]   ]]]],
```

Common mistakes for typing this
Having ‘[’ or ‘]’ in the wrong place

The actual functions are defined elsewhere

Let’s see the code for a function the state runs

```
def validOp(node, var_store):   # a main (state_string_1, state_string_2) function signature, not a helper
function
      i = var_store['i']
      input_ = var_store['input']
      if isOp(node, var_store):
            var_store['operation_vars']['a'] = input_[i - 1]
            return True  # because the function succeeded
      return False  # the function failed
```

Notice how the function returns true or false.

All of the state functions will return true or false.

If the state function succeeds, then it returns true.

The state function is not a helper function.

Make sure all the connections are set the way you expect before you test it

When you design the sequences write the states like this (string_1, string_2) -> (another_string_1, another_string_2)

That way you know what order you want

Link to github page for project materials

https://github.com/dtauraso/State-Machine-Tutorial

https://github.com/dtauraso/contextual-state-chart