

PEP Scala (4)

Email: christian.urban at kcl.ac.uk
Office: N7.07 (North Wing, Bush House)
Slides & Code: KEATS
Office Hours: Mondays 12:00 – 14:00

Somewhere Remote



This is a bit harsh...

...trying a new method because the fucking github reports dont tell me which test failed. It's not really helpful when the inline tests work and it compiles but all i get is 'one test failed'... really helpful my dude.

...Reverted back and finished part 5, this is ridiculous how one test works and all I get is 'ONE TEST FAILED'. Fix your reports before giving us assignments like this...


Needless to say I tried it out

```
> legal_moves(8, Nil, (2,2))  
= List((3,4), (4,3), (4,1), (3,0), (1,0), (0,1), (0,3), (1,4))
```


```
> legal_moves(8, Nil, (7,7))  
= List((6,5), (5,6))
```

```
> legal_moves(8, List((4,1), (1,0)), (2,2))  
= List((3,4), (4,3), (3,0), (0,1), (0,3), (1,4))
```


```
> legal_moves(1, Nil, (0,0))  
= List((-1,-2), (-2,-1))
```



```
> legal_moves(2, Nil, (0,0))  
= List((1,-2), (-1,-2), (-2,-1), (-2,1))
```



```
> legal_moves(3, Nil, (0,0))  
= List((1,2), (2,1), (2,-1), (1,-2), (-1,-2), (-2,-1), (-2,1))
```



```
def is_legal(dim: Int, p: Path, x: Pos): Boolean = {  
  if (.....some_really_long_condition.....) false  
  else true  
}
```

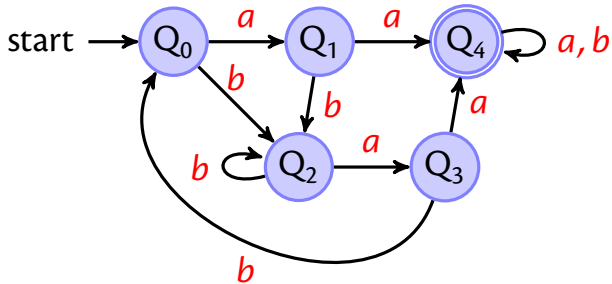
```
def is_legal(dim: Int, p: Path, x: Pos): Boolean = {  
  if (.....some_really_long_condition.....) false  
  else true  
}
```

```
def is_legal(dim: Int, p: Path, x: Pos): Boolean =  
  !.....some_really_long_condition.....
```

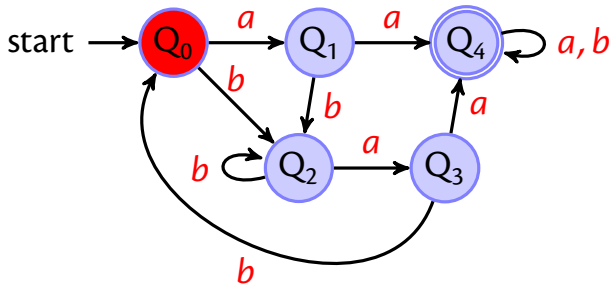
```
def is_legal(dim: Int, p: Path, x: Pos): Boolean = {  
  if (.....some_really_long_condition.....) false  
  else true  
}
```

```
def is_legal(dim: Int, p: Path, x: Pos): Boolean =  
  !.....some_really_long_condition.....
```

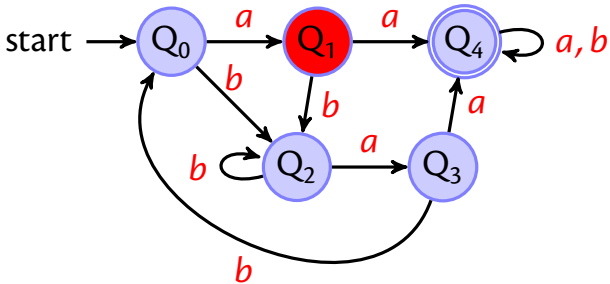
DFAs



DFAs

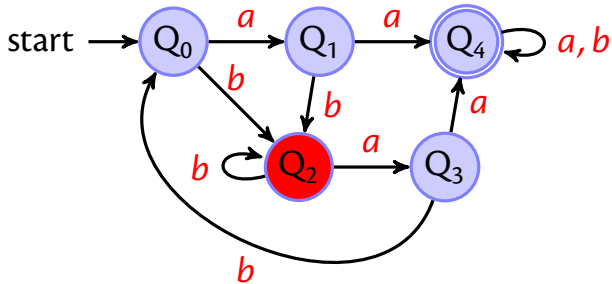


DFAs



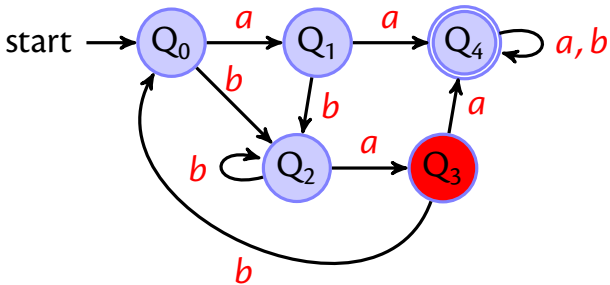
a

DFAs



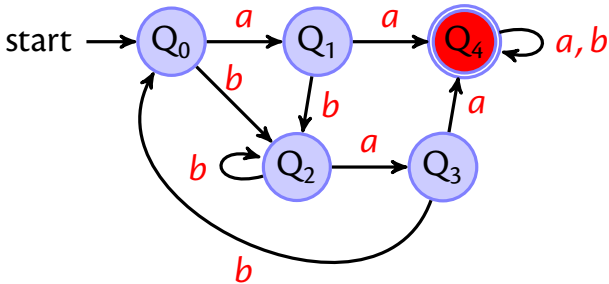
ab

DFAs



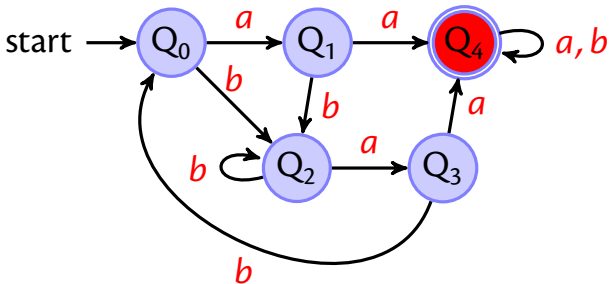
aba

DFAs



abaa

DFAs



abaaa \Rightarrow *yes*

DFAs

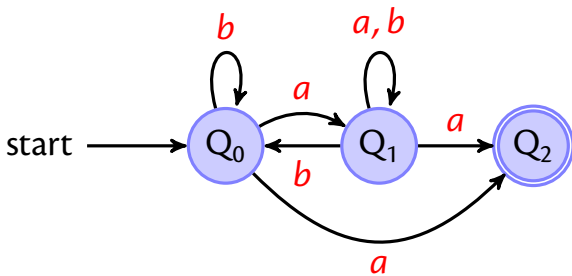
A **deterministic finite automaton**, DFA, consists of 5 things:

- an alphabet Σ
- a set of states Q_s
- one of these states is the start state Q_0
- some states are accepting states F , and
- there is transition function δ

which takes a state and a character as arguments and produces a new state; this function might not be everywhere defined

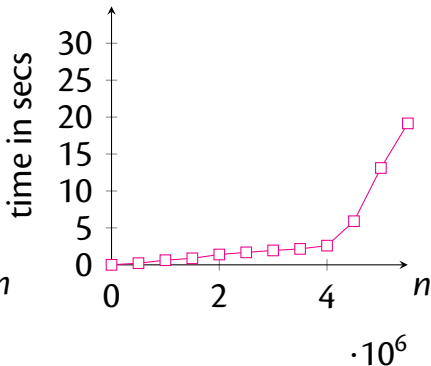
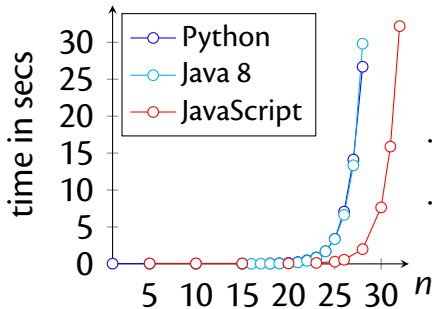
$$A(\Sigma, Q_s, Q_0, F, \delta)$$

NFAs



CW9 (1 Part): Regexes

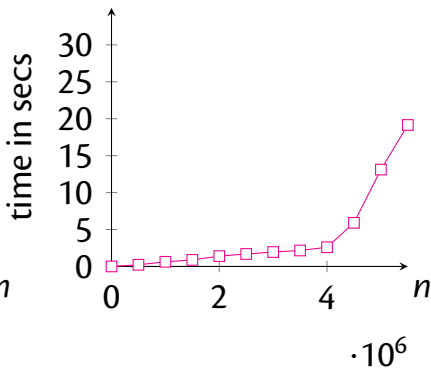
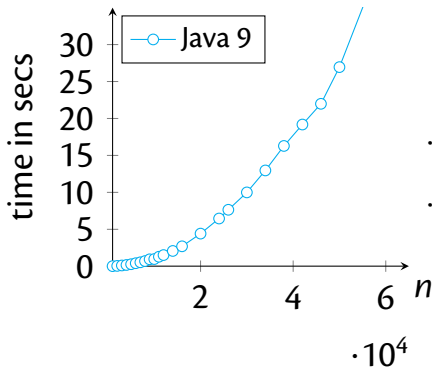
Graphs: $(a^*)^*b$ and strings $\underbrace{a \dots a}_n$



<https://vimeo.com/112065252>

CW9 (1 Part): Regexes

Graphs: $(a^*)^*b$ and strings $\underbrace{a \dots a}_n$



<https://vimeo.com/112065252>

Hint

Pattern-Matching

Questions?