02/07/2024

Aesthetic Regular Expression Generation from Symbolic Finite Automata

Axel Gautier

Jim Newton

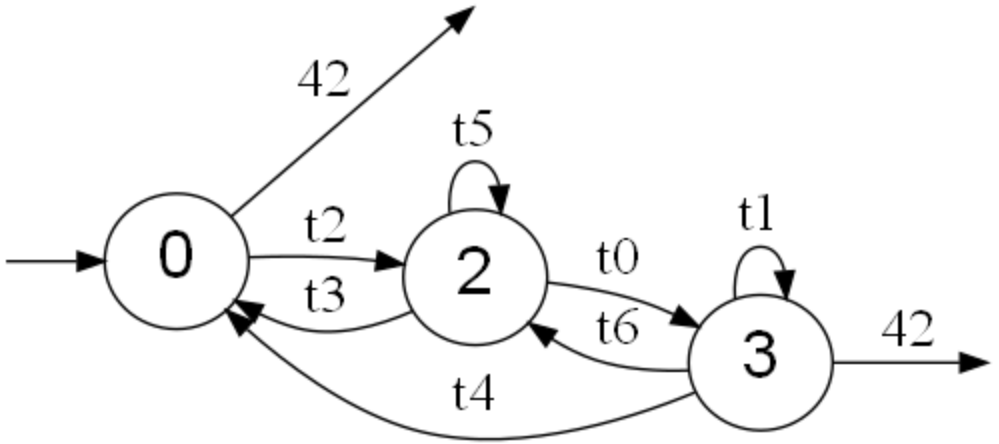
Summary

- Reminder on DFAs and RTEs
- A summary of the naïve state elimination extraction
- My work on implementing a state splitting algorithm

Symbolic Deterministic Finite Automata

```
0= SAnd(SAtomic:String,evenp?)
1= SOr(SAnd(SAtomic:Integer,evenp?),SAnd(SAtomic:String,evenp?))

2= SAtomic:Integer
3= SAnd(!SAtomic:String,evenp?)
4= SAnd(!SAtomic:Integer,!SAtomic:String,evenp?)
5= SAnd(SAtomic:String,!evenp?)
6= SOr(SAnd(SAtomic:Integer,!evenp?),SAnd(SAtomic:String,
    !evenp?))
```



Regular Type Expression representation

- Star(Cat(classOf[Int],Star(classOf[String]), evenp))

- Singleton(SAnd(Int, SOr(String, evenp)))

State Elimination Algorithm

- Add a new Initial state and add an EmptyWord transition from the new initial to the old initial state
- Add a new final state and add EmptyWord transitions from every old final state to the new final state

- While there are states remaining that aren't the new Initial and Final state:

- Select a state following a specific rule, for example the state with the least amount of transitions, and remove it from the graph while

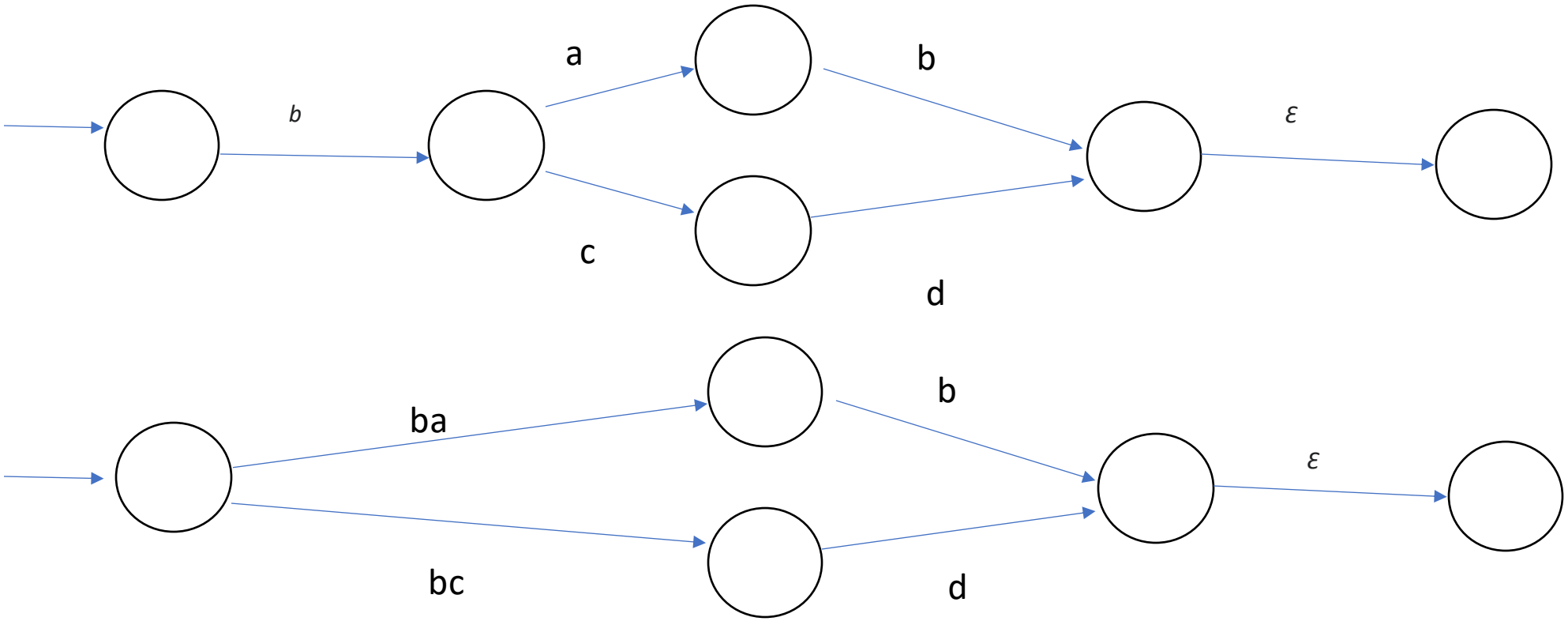
State Elimination Algorithm

- While there are states remaining that aren't the new Initial and Final state:

- Select a state following a specific rule, for example the state with the least amount of transitions.

- Remove the state from the graph while creating new transitions to make sure the graph keeps representing the same Regular Expression

State Elimination Algorithm

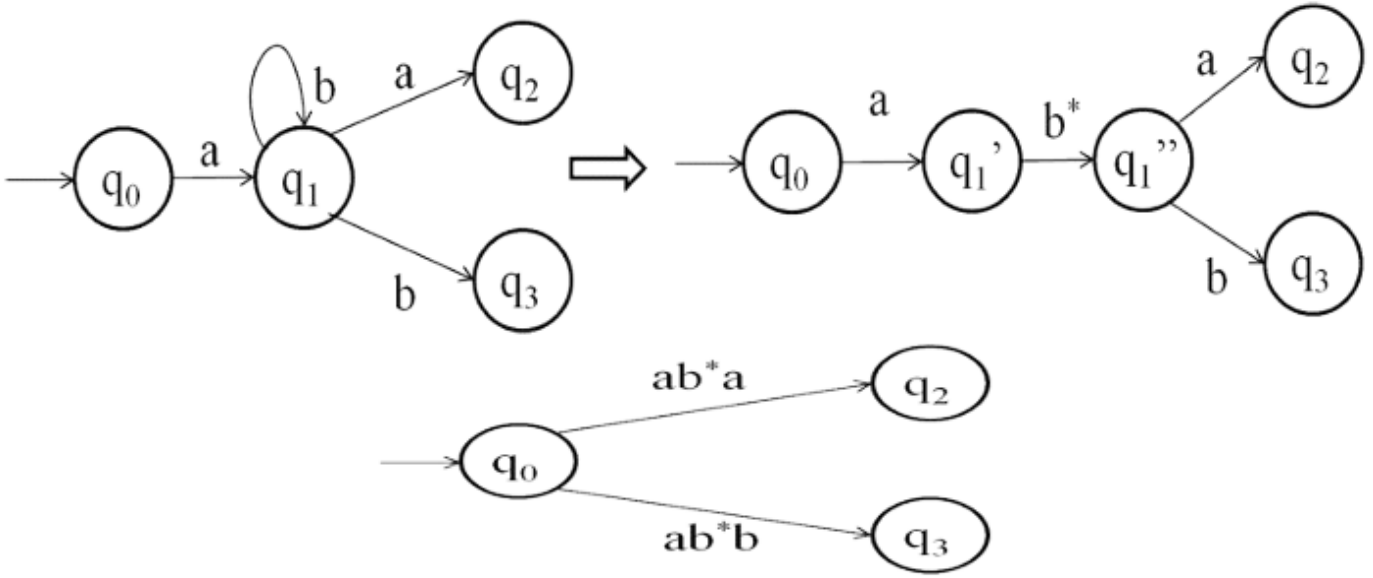


Until there are only the new Initial and Final states left

State splitting for self loops

- Loop State Splitting Algorithm as described in:

An elegant technique for obtaining shorter regular expressions, International Journal of Innovative Research & Studies, Dr. O. V. Shanmuga Sundaram



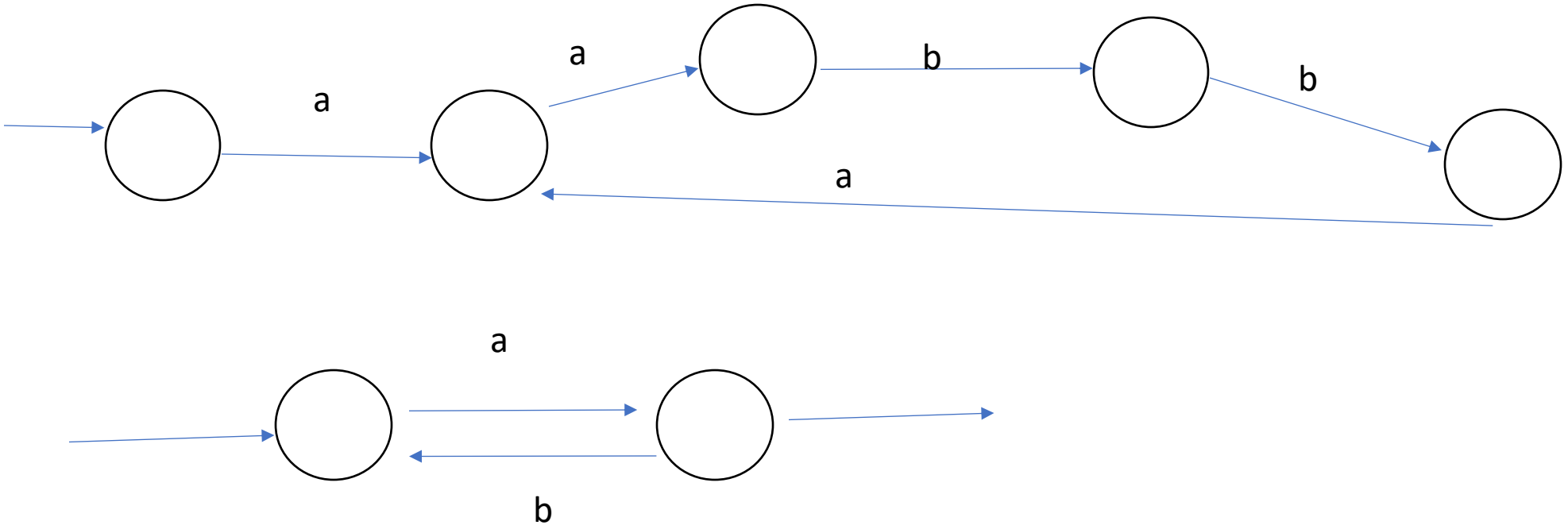
My work on a State splitting algorithm

Transform the graph into a general tree

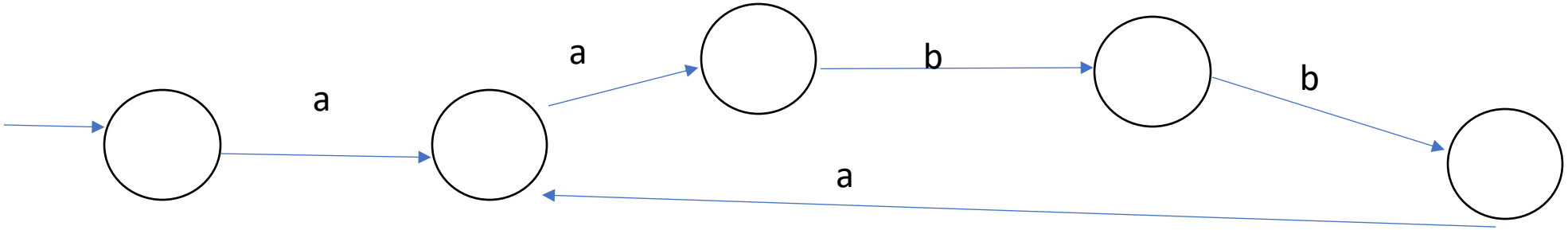
- Go through each state and eliminate self loops and then cycles

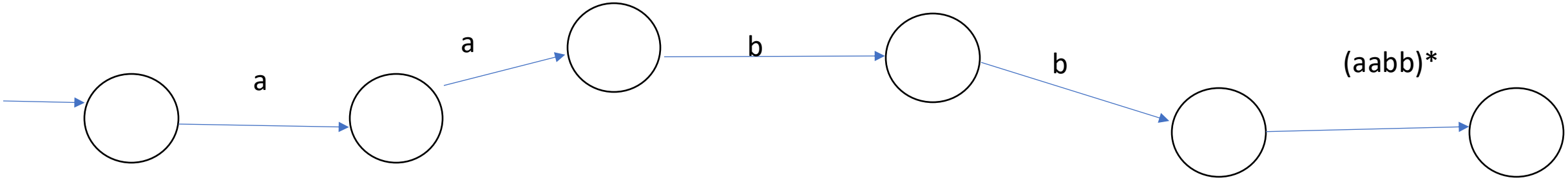
- Eliminate all parallel transitions

Backwards transitions and cycles

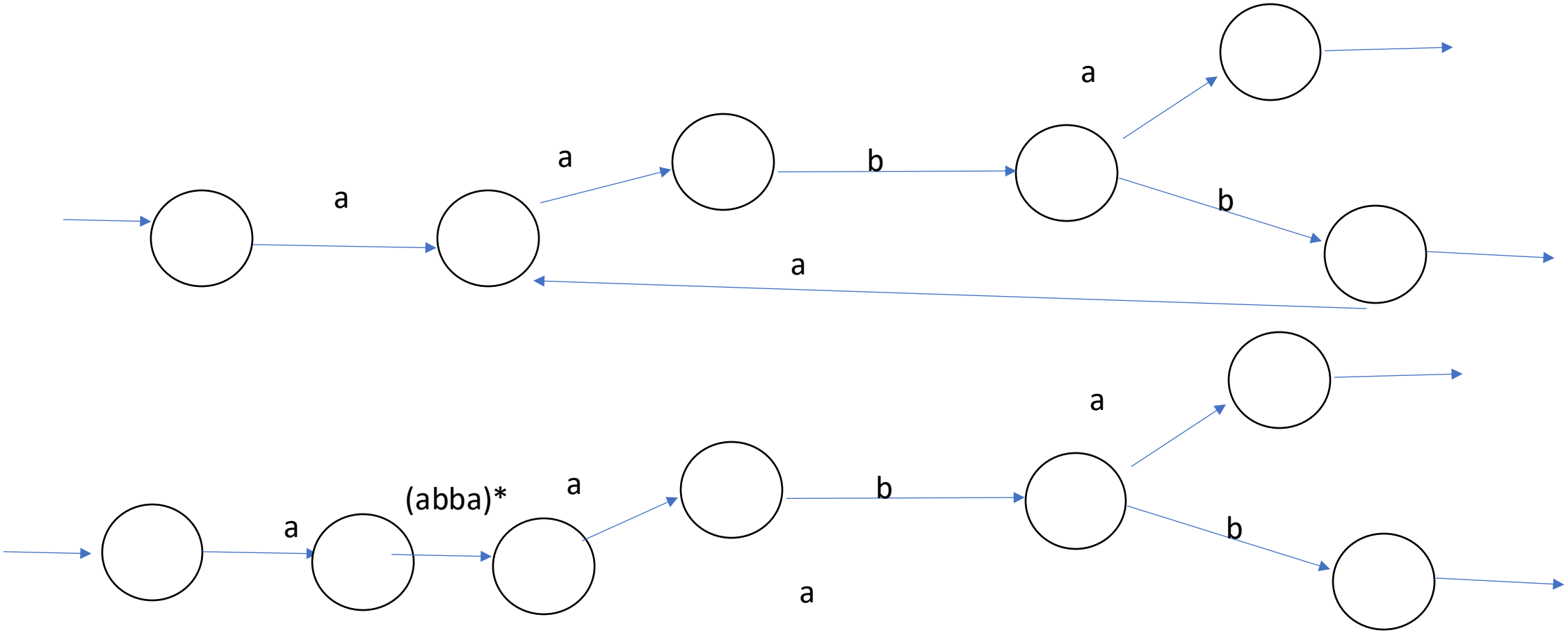


For example





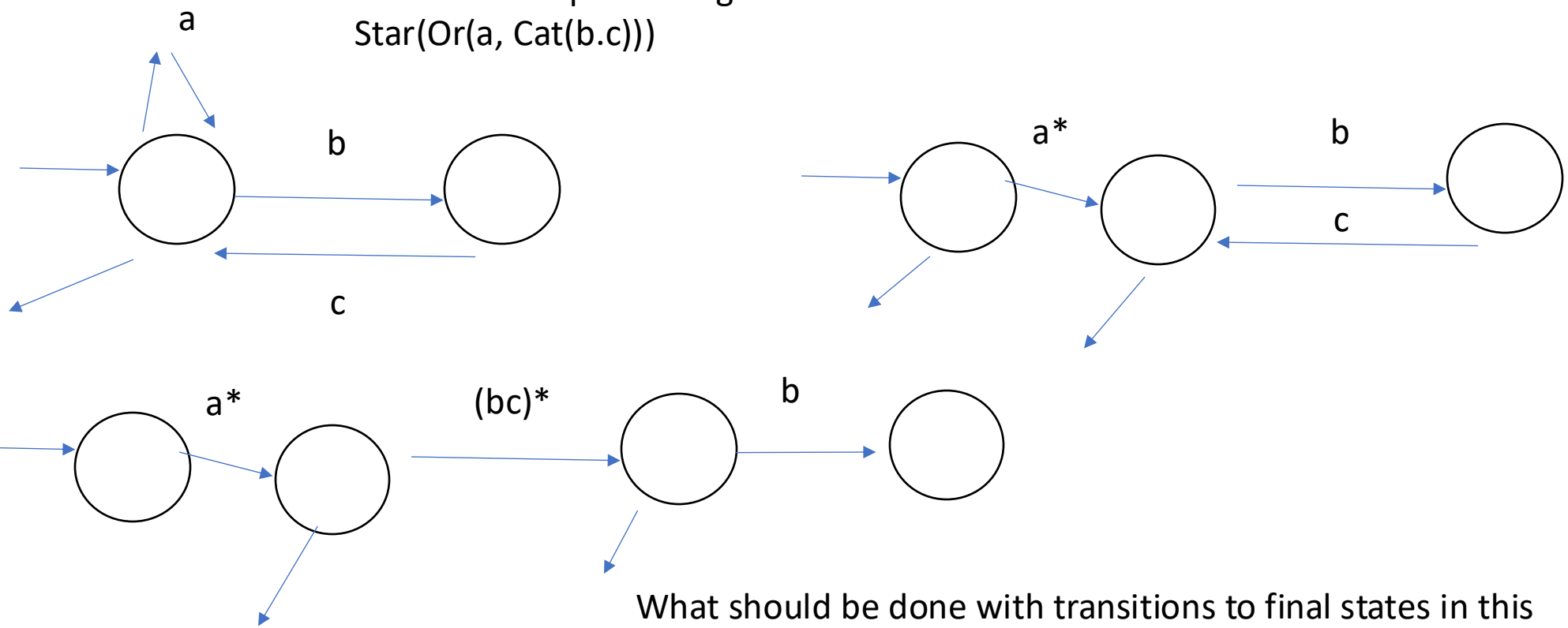
But what if there are outgoing transitions?



It can become more complicated when there are both incoming and outgoing transitions inside of the cycle

Cycles and self loops on the same state

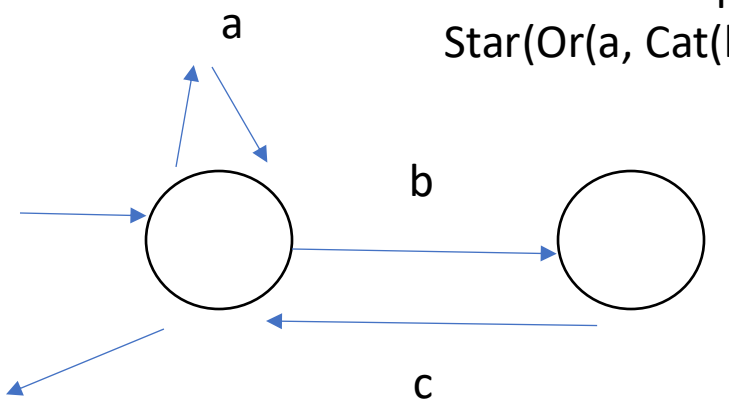
Automaton representing
Star(Or(a, Cat(b.c)))



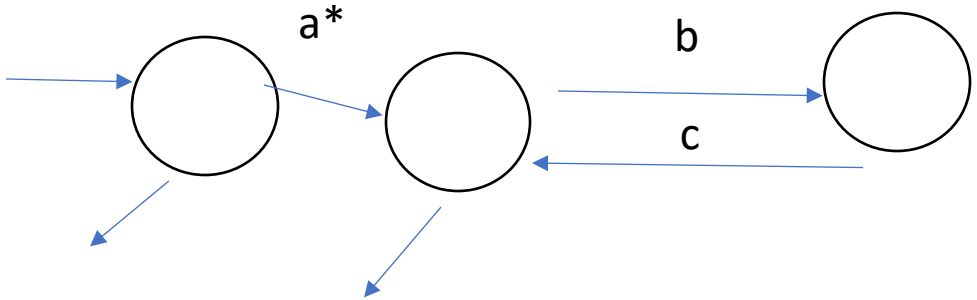
What should be done with transitions to final states in this situation is unclear Also the automaton now represents:
Or(Star(a), Cat(Star(a), Star(Cat(b,c)))

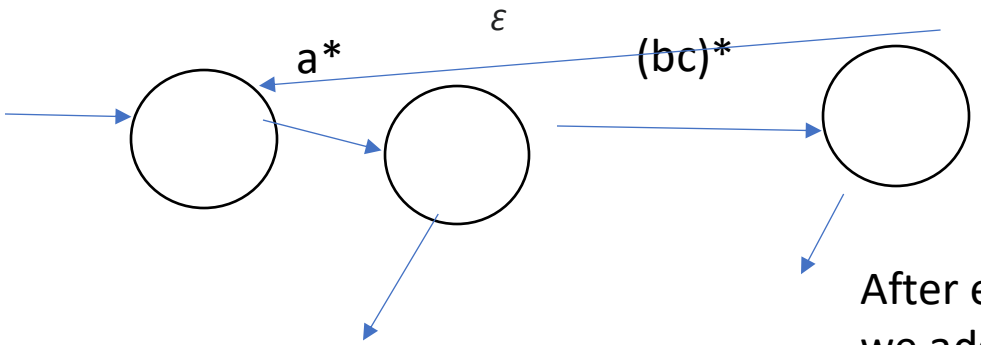
Solution to the problem

Automaton representing Star(Or(a, Cat(b.c)))

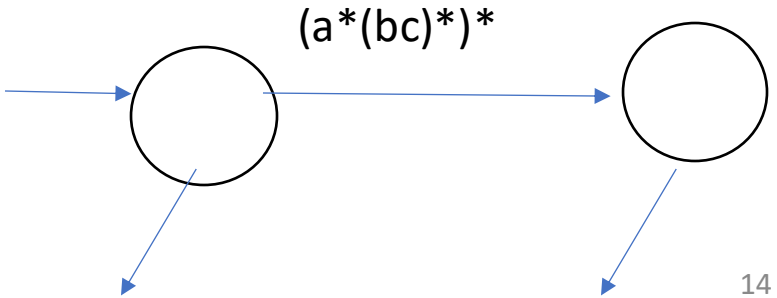


When there are two cycles or self loops on the same state, we mark it once we have eliminated the first loop





After eliminating the second loop we add an Epsilon transition in order to create another loop and end up with



Comparison of this case with the state elimination algorithm

- State Elimination Algorithm

Or(Cat(Star(Int), Boolean, Star(Cat(String, Int, Boolean), String), Star(Int), Star(Int))

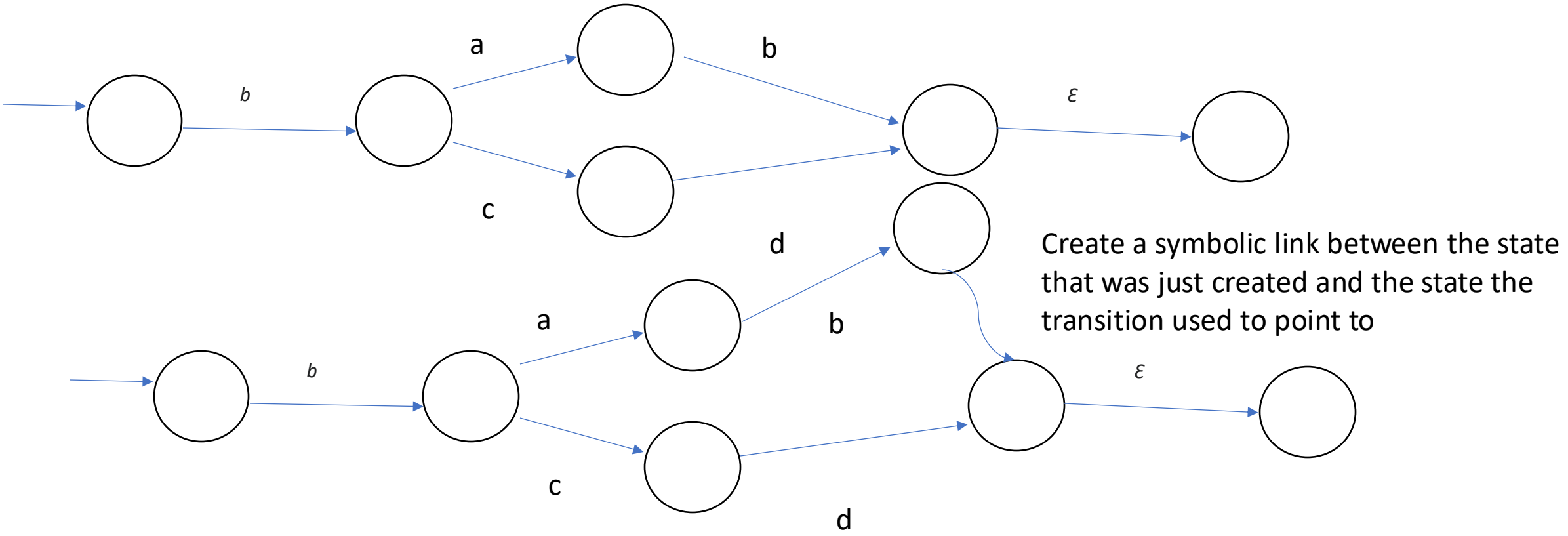
- RTE extracted using the presented algorithm:

Star(Cat(Star(Integer), Star(Cat(Boolean, String))))

Both RTEs are equivalent however the second one is smaller and shallower

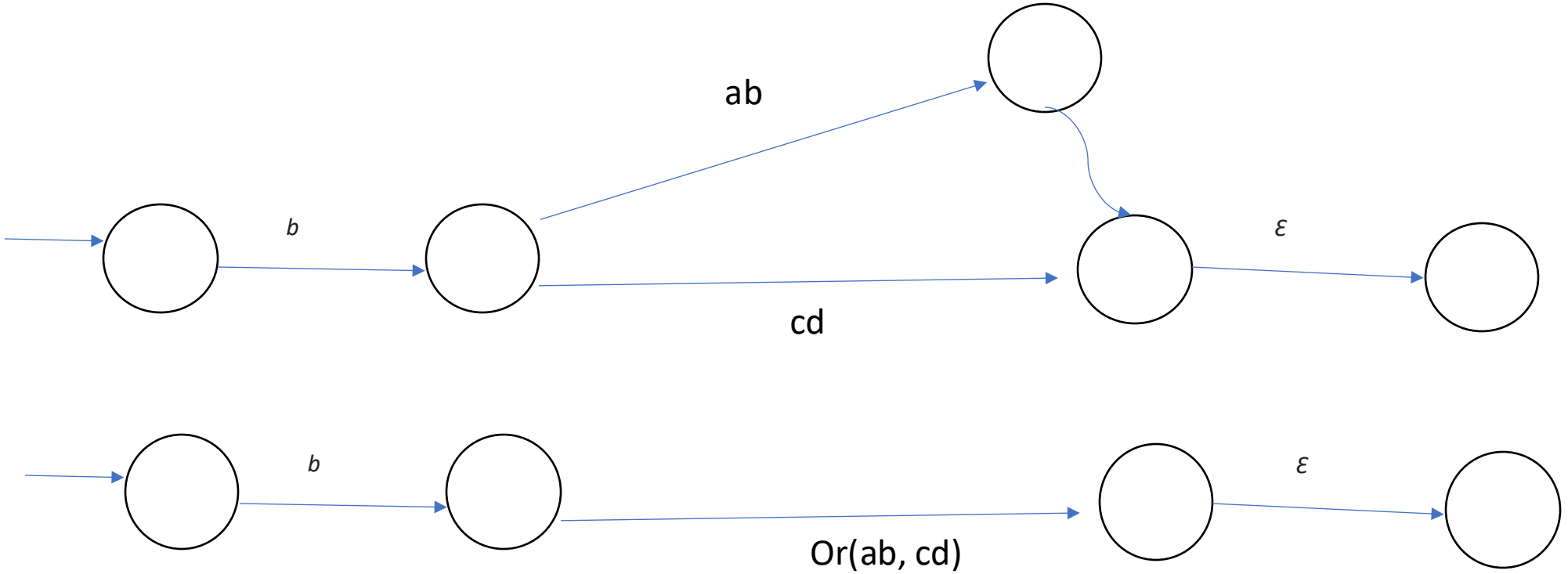
Parallel transitions

- Eliminate all parallel transitions:



Parallel transitions

- This makes it easier to calculate the regular expression in each path



What's next

Continue running tests and figuring out the best way to extract an RTE in each situation

Bibliography

- ScalaCheck, the definitive guide: Rickard Nilsson
- Scala programming language: Martin Odersky, EPFL
- A Portable, Simple, Embeddable Type System (Newton, Pommellet) ELS 2021
- Elements of Automata Theory, Jacques Sakarovitch
- An elegant technique for obtaining shorter regular expressions, International Journal of Innovative Research & Studies, Dr. O. V. Shanmuga Sundaram
- Approximation to the Smallest Regular Expression for a Given Regular Language. M Delgado, J Morais.