CS 160 Compilers



# Lecture 1: Revisiting DFA & NFA



Yu Feng

Fall 2021

# Outline



- Today: Revisiting 

A & DFA
- High-level story: RegEx -> NFA -> DFA -> Table

# Finite automata



- Regular Expressions ⇐ [illegible]on
- Finite Automata ⇔ Implementation
- A finite automata formally consists of:



- An input alphabet Σ
- A set of states S
- A start state n
- A set of accepting states $F \subseteq S$
- A set of transitions state $\rightarrow^{\text{input}}$ state

# Finite automata



- Transition $S_1 \rightarrow^{\alpha} S$


- This means: In state $S_1$ and input character α, go to state $S_2$



- If end of input and in accepting state ⇒ accept



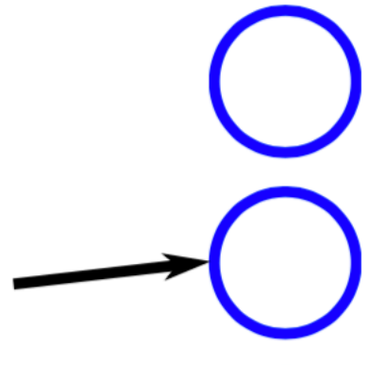
- Otherwise ⇒ reject

# Finite Automata as State Graphs

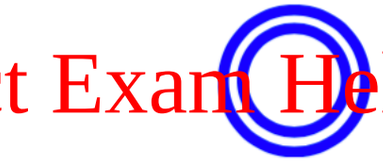


A state:

The start state:



An accepting state:

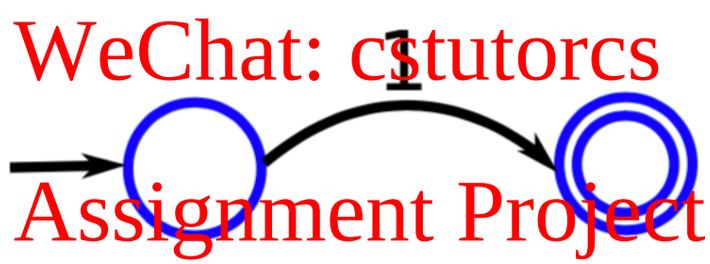


A transition:

# A simple example



- Here is an automa  nly accepts the string ”1”:



# Another simple example

- A finite automaton accepting any number of 1's followed by a single 0

- Alphabet: {0,1}



# Epsilon transitions



- A special kind of t ε-transitions
- Machine can move from state A to B without reading any input

# Deterministic and Nondeterministic Automata



- Deterministic Finite Automata (DFA)
  - At most one transition per input on any state
  - No ε moves
- Nondeterministic Finite Automate (NFA)
  - Can have multiple transitions for one input in a given state
  - Can have ε-moves

# RE to NFA

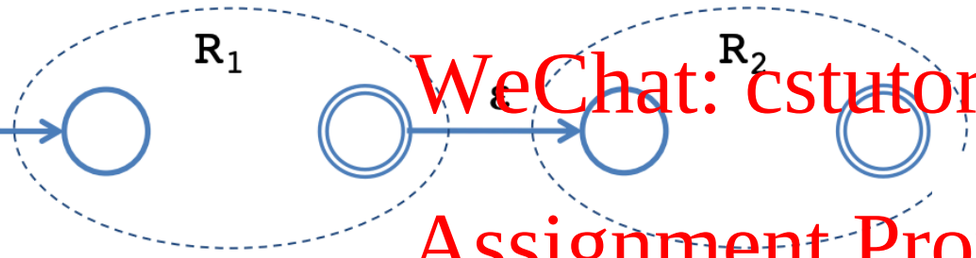


'a'


ε

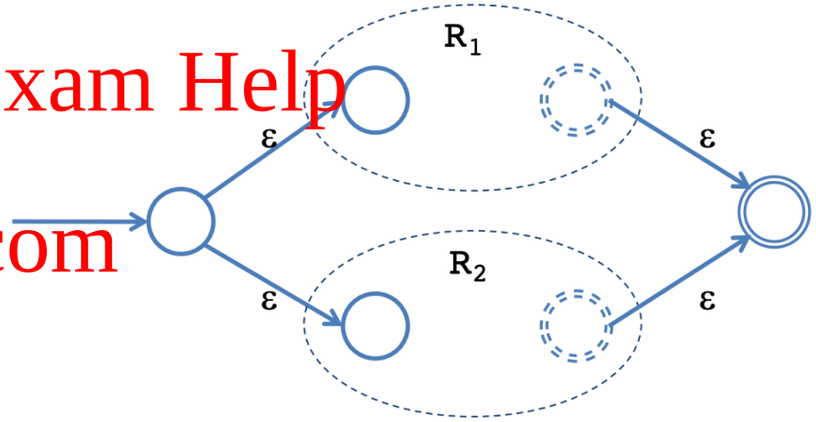


$R_1R_2$

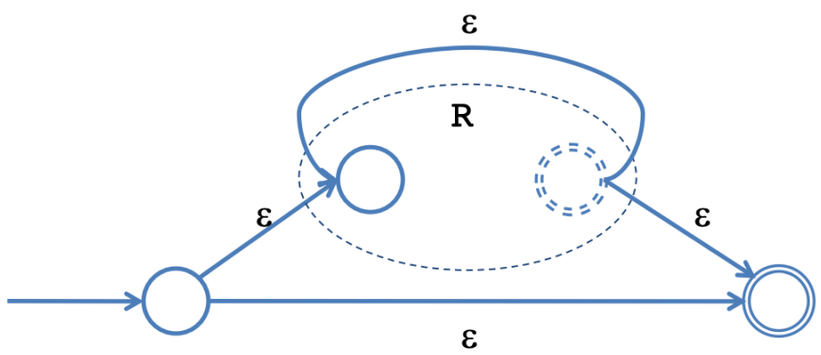





$R_1 | R_2$




$R^*$

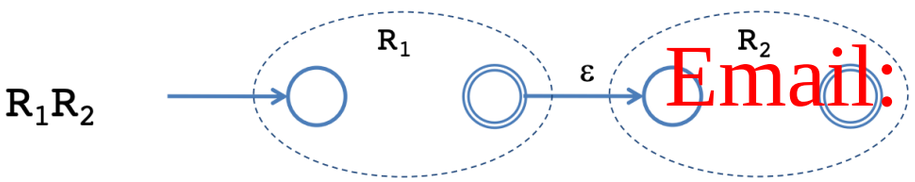


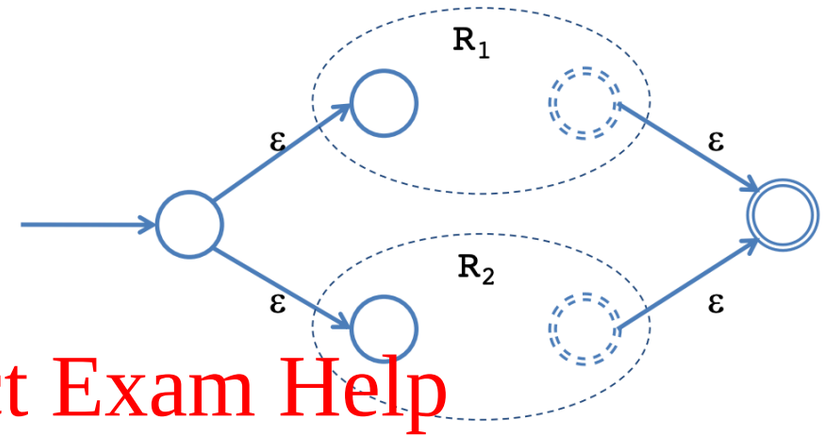
# In-class exercise

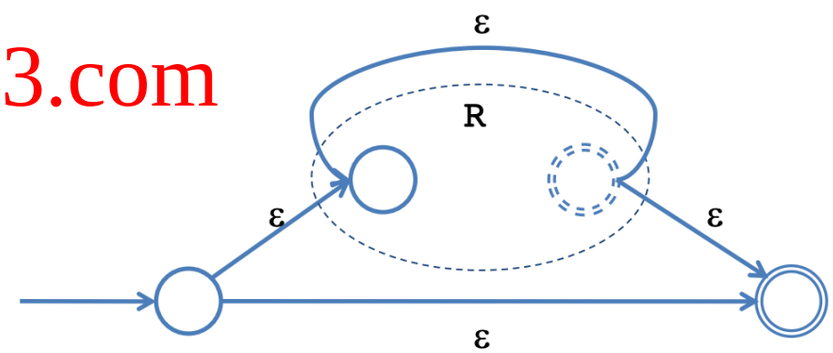


- Please draw the N(b | c)*





# NFA to DFA: The Algorithm

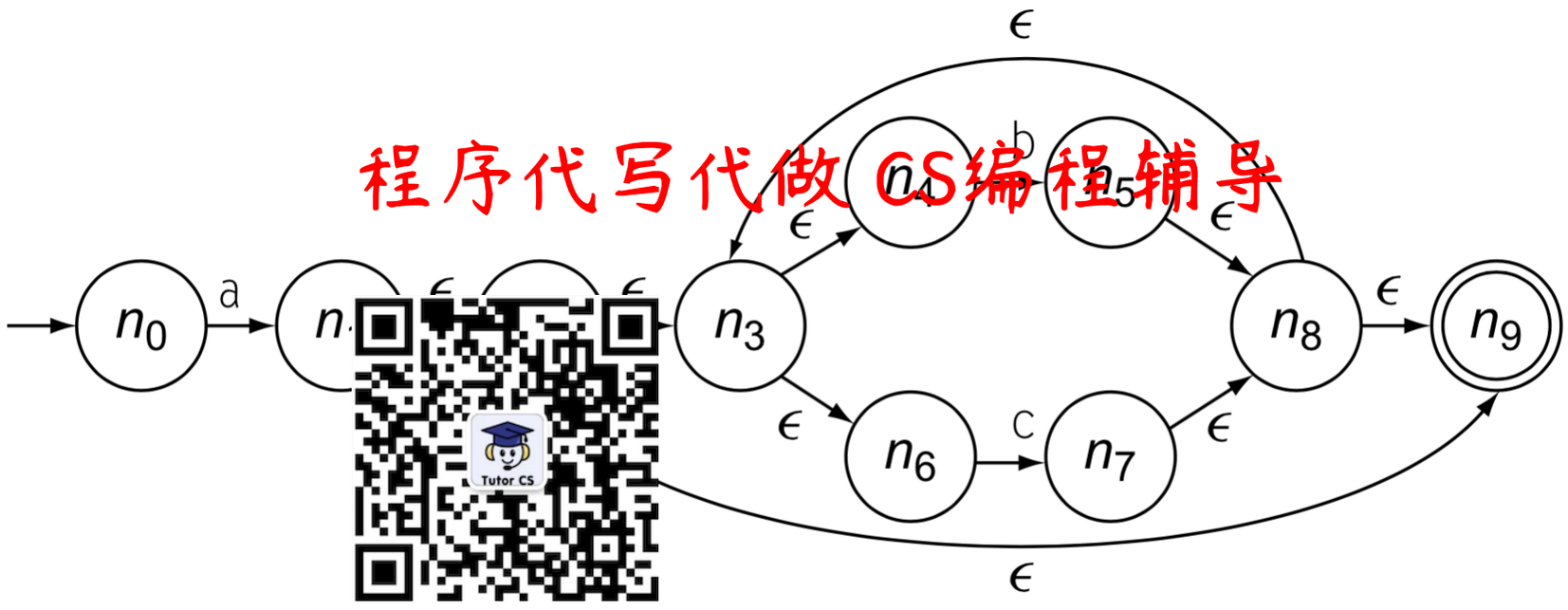


```
q0 ← ϵ-(n0});
Q ← q0;
WorkList

while (WorkList ≠ ∅) do
    remove q from WorkList;
    for each character c ∈ Σ do
        t ← ϵ-closure(Delta(q,c));
        T[q,c] ← t;
        if t ∉ Q then
            add t to Q and to WorkList;
    end;
end;
```



Apply NFA's transition function to each element of $q$



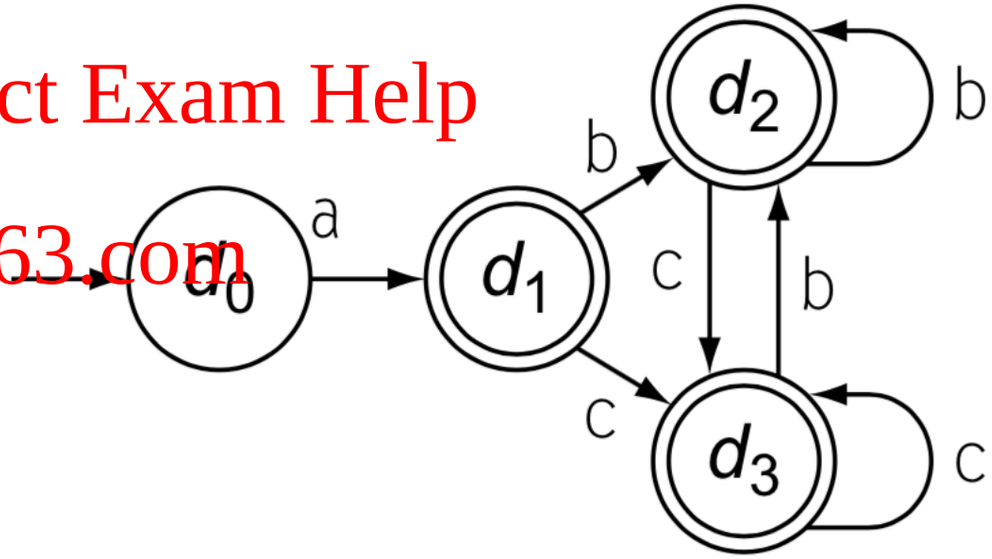

(a) NFA for "$a(b \mid c)^*$" (With States Renumbered)



| Set Name | DFA States | NFA States | ϵ-closure(Delta(q,*)) | | |
|---|---|---|---|---|---|
| | | | a | b | c |
| $q_0$ | $d_0$ | $n_0$ | $\{n_1, n_2, n_3, n_4, n_6, n_9\}$ | – none – | – none – |
| $q_1$ | $d_1$ | $\{n_1, n_2, n_3, n_4, n_6, n_9\}$ | – none – | $\{n_5, n_8, n_9, n_3, n_4, n_6\}$ | $\{n_7, n_8, n_9, n_3, n_4, n_6\}$ |
| $q_2$ | $d_2$ | $\{n_5, n_8, n_9, n_3, n_4, n_6\}$ | – none – | $q_2$ | $q_3$ |
| $q_3$ | $d_3$ | $\{n_7, n_8, n_9, n_3, n_4, n_6\}$ | – none – | $q_2$ | $q_3$ |

(b) Iterations of the Subset Construction






(a) Resulting DFA

*Engineering a compiler, C2.4*

# TODOs by next lecture



- Hw2 will be out. Get familiar with the Patina language


- Come to the discussion session if you have questions