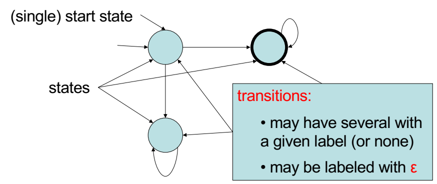
CS21
Decidability
and
Tractability

Lecture 3
January 9, 2023

1

## NFA diagrams




- At each step, several choices for next state
  - if *possible* to reach accept, then input accepted



2

## NFA formal definition

A nondeterministic FA … (Q, Σ, δ …

transit… labeled … alpha… symbols or ε

"powerset of Q": the set of all subsets of Q

- Q is a finite set called …
- Σ is a finite set called the alphabet
- δ:Q x (Σ ∪ {ε}) → $\mathcal{P}$(Q) is a function called the transition function
- $q_0$ is an element of Q called the start state
- F is a subset of Q called the accept states



3

## Formal description of NFA operation

NFA M = (Q, Σ, δ, $q_0$, F)
accepts a string w = $w_1w_2w_3\ldots w_n \in \Sigma^*$
if w can be written (by inserting ε's) as:

$$y = y_1y_2y_3\ldots y_m \in (\Sigma \cup \{\varepsilon\})^*$$

and ∃ sequence $r_0, r_1, \ldots, r_m$ of states for which
- $r_0 = q_0$
- $r_{i+1} \in \delta(r_i, y_{i+1})$ for i = 0,1,2, …, m-1
- $r_m \in F$



4

## Closures

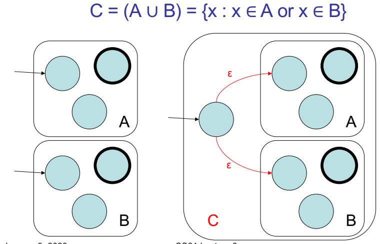
- Recall: want to show the set of languages recognized by NFA is closed under:
  - union "C = (A ∪ B)"
  - concatenation "C = (A ∘ B)"
  - star " C = A* "



5

## Closure under union

C = (A ∪ B) = {x : x ∈ A or x ∈ B}

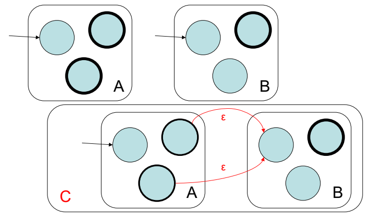





6

## Closure under concatenation

$C = (A \circ B) = \{xy : x \in A \text{ and } y \in B\}$

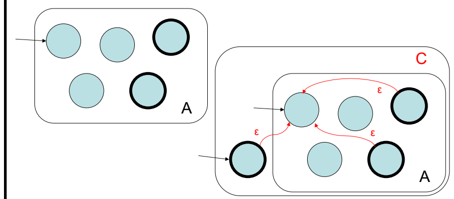







## Closure under star

$C = A^* = \{x_1x_2x_3\ldots x_k : k \geq 0 \text{ and each } x_i \in A\}$








## NFA, FA equivalence

**Theorem**: a language L is recognized by a FA if and only if L is recognized by a NFA.

Must prove *two* directions:

($\Rightarrow$) L is recognized by a FA implies L is recognized by a NFA.

($\Leftarrow$) L is recognized by a NFA implies L is recognized by a FA.

(usually one is easy, the other more difficult)





## NFA, FA equivalence

($\Rightarrow$) L is recognized by a FA implies L is recognized by a NFA

**Proof**: a finite automaton *is* a nondeterministic finite automaton that happens to have no ε-transitions, and for which each state has exactly one outgoing transition for each symbol.





## NFA, FA equivalence

($\Leftarrow$) L is recognized by a NFA implies L is recognized by a FA.

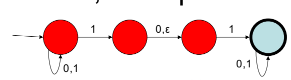
**Proof**: we will build a FA that *simulates* the NFA (and thus recognizes the same language).

- alphabet will be the same
- what are the states of the FA?



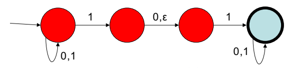


## NFA, FA equivalence




- given NFA $M = (Q, \Sigma, \delta, q_0, F)$
- construct FA $M' = (Q', \Sigma', \delta', q_0', F')$
- same alphabet: $\Sigma' = \Sigma$
- states are subsets of M's states: $Q' = \mathcal{P}(Q)$
- if we are in state $R \in Q'$ and we read symbol $a \in \Sigma'$, what is the new state?

## NFA, FA equivalence




– given NFA M = $(Q, \Sigma, \delta, q_0, F)$

– construct FA M' = $(Q', \Sigma', \delta', q_0', F')$

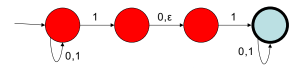
Helpful def'n: $E(S) = \{q \in Q$ : q reachable from S by traveling along 0 or more ε-transitions$\}$

– new transition fn: $\delta'(R, a) = \cup_{r \in R} E(\delta(r, a))$
= "all nodes reachable from R by following an a-transition, and then 0 or more ε-transitions"





## NFA, FA equivalence




– given NFA M = $(Q, \Sigma, \delta, q_0, F)$

– construct FA M' = $(Q', \Sigma', \delta', q_0', F')$

– new start state: $q_0' = E(\{q_0\})$

– new accept states:
$F' = \{R \in Q'$ : R contains an accept state of M$\}$





## NFA, FA equivalence

- We have proved (⇐) by construction.

Formally we should also prove that the construction works, by induction on the number of steps of the computation.

– at each step, the state of the FA M' is exactly the set of reachable states of the NFA M…





## So far…

**Theorem**: the set of languages recognized by NFA is closed under union, concatenation, and star.

**Theorem**: a language L is recognized by a FA if and only if L is recognized by a NFA.

**Theorem**: the set of languages recognized by FA is closed under union, concatenation, and star.





## Next…

- Describe the set of languages that can be built up from:
  – unions
  – concatenations
  – star operations
- Called "patterns" or regular expressions
- **Theorem**: a language L is recognized by a FA if and only if L is described by a regular expression.





## Regular expressions

- R is a regular expression if R is
  – a, for some $a \in \Sigma$
  – ε, the empty string
  – Ø, the empty set
  – $(R_1 \cup R_2)$, where $R_1$ and $R_2$ are reg. exprs.
  – $(R_1 \circ R_2)$, where $R_1$ and $R_2$ are reg. exprs.
  – $(R_1^*)$, where $R_1$ is a regular expression

A reg. expression R describes the language L(R).

## Regular expressions

- example: $R = (0 \cup 1)$
  - if $\Sigma = \{0,1\}$ then use "$\Sigma$" as shorthand for R
- example: $R = 0 \circ \Sigma^*$
  - shorthand: omit "$\circ$"   $R = 0\Sigma^*$
  - precedence: *, then $\circ$ then $\cup$, unless override by parentheses
  - in example $R = 0(\Sigma^*)$, not $R = (0\Sigma)^*$





## Some examples

alphabet $\Sigma = \{0,1\}$

- $\{w : w \text{ has at least one } 1\}$
  $= \Sigma^* 1 \Sigma^*$
- $\{w : w \text{ starts and ends with same symbol}\}$
  $= 0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$
- $\{w : |w| \leq 5\}$
  $= (\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)$
- $\{w : \text{every } 3^{rd} \text{ position of } w \text{ is } 1\}$
  $= (1\Sigma\Sigma)^*(\varepsilon \cup 1 \cup 1\Sigma)$





## Manipulating regular expressions

- The empty set and the empty string:
  - $R \cup \varnothing = R$
  - $R\varepsilon = \varepsilon R = R$
  - $R\varnothing = \varnothing R = \varnothing$
  - $\cup$ and $\circ$ behave like +, x;  $\varnothing$, $\varepsilon$ behave like 0,1
- additional identities:
  - $R \cup R = R$   (here + and $\cup$ differ)
  - $(R_1{}^*R_2)^*R_1{}^* = (R_1 \cup R_2)^*$
  - $R_1(R_2R_1)^* = (R_1R_2)^*R_1$





## Regular expressions and FA

- **Theorem**: a language L is recognized by a FA if and only if L is described by a regular expression.

Must prove *two* directions:

($\Rightarrow$) L is recognized by a FA implies L is described by a regular expression

($\Leftarrow$) L is described by a regular expression implies L is recognized by a FA.





## Regular expressions and FA

($\Leftarrow$) L is described by a regular expression implies L is recognized by a FA

**Proof**: given regular expression R we will build a NFA that recognizes L(R).

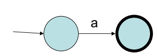
then NFA, FA equivalence implies a FA for L(R).



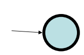


## Regular expressions and FA

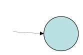
- R is a regular expression if R is
  - a, for some $a \in \Sigma$




  - $\varepsilon$, the empty string

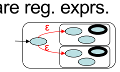


  - $\varnothing$, the empty set

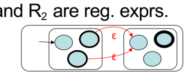


# Regular expressions and FA

– $(R_1 \cup R_2)$, where $R_1$ and $R_2$ are reg. exprs.



– $(R_1 \circ R_2)$, where $R_1$ and $R_2$ are reg. exprs.



– $(R_1{}^*)$, where $R_1$ is a regular expression