

CS21
Decidability
and
Tractability

Lecture 3-4
January 14,
2025

1

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)

January 14, 2025 CS21 Lecture 3-4 2

2

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.

January 14, 2025 CS21 Lecture 3-4 3

3

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?

January 14, 2025 CS21 Lecture 3-4 4

4

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?

January 14, 2025 CS21 Lecture 3-4 5

5

NFA, FA equivalence

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

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

– 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"

January 14, 2025 CS21 Lecture 3-4 6

6

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

January 14, 2025 CS21 Lecture 3-4 7

7

NFA, FA equivalence

- We have proved (\Leftarrow) 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 ...

January 14, 2025 CS21 Lecture 3-4 8

8

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.

January 14, 2025 CS21 Lecture 3-4 9

9

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.

January 14, 2025 CS21 Lecture 3-4 10

10

Regular expressions

- R is a regular expression if R is
 - a , for some $a \in \Sigma$
 - ϵ , the empty string
 - \emptyset , 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** $L(R)$.

January 14, 2025 CS21 Lecture 3-4 11

11

Regular expressions

- example: $R = (0 \cup 1)$
 - if $\Sigma = \{0,1\}$ then use “ Σ ” 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 $R = (0\Sigma)^*$

January 14, 2025 CS21 Lecture 3-4 12

12

Some examples

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

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

January 14, 2025 CS21 Lecture 3-4 13

13

Manipulating regular expressions

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

January 14, 2025 CS21 Lecture 3-4 14

14

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.

January 14, 2025 CS21 Lecture 3-4 15

15

Regular expressions and FA

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

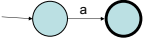


Proof: given regular expression R we will build a NFA that recognizes L(R).

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

January 14, 2025 CS21 Lecture 3-4 16

16

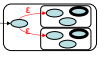
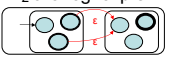
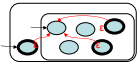
Regular expressions and FA

- R is a regular expression if R is
 - a , for some $a \in \Sigma$ 
 - ϵ , the empty string 
 - \emptyset , the empty set 

January 14, 2025 CS21 Lecture 3-4 17

17

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 

January 14, 2025 CS21 Lecture 3-4 18

18

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.

January 14, 2025

CS21 Lecture 3-4

19

19

Regular expressions and FA

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

Proof: given FA M that recognizes L , we will

1. build an equivalent machine “Generalized Nondeterministic Finite Automaton” (GNFA)
2. convert the GNFA into a regular expression

January 14, 2025

CS21 Lecture 3-4

20

20

Regular expressions and FA

- GNFA definition:

– it is a NFA, but may have **regular expressions** labeling its transitions

– GNFA accepts string $w \in \Sigma^*$ if can be written
 $w = w_1w_2w_3 \dots w_k$

where each $w_i \in \Sigma^*$, and there is a path from the start state to an accept state in which the i^{th} transition traversed is labeled with R for which $w_i \in L(R)$

January 14, 2025

CS21 Lecture 3-4

21

21

Regular expressions and FA

- Recall step 1: build an equivalent GNFA

- Our FA M is a GNFA.

- We will require “**normal form**” for GNFA to make the proof easier:

- *single* accept state q_{accept} that has all possible incoming arrows
- every state has all possible outgoing arrows; exception: start state q_0 has no self-loop

January 14, 2025

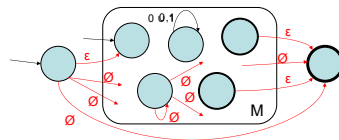
CS21 Lecture 3-4

22

22

Regular expressions and FA

- converting our FA M into GNFA in normal form:



January 14, 2025

CS21 Lecture 3-4

23

23

Regular expressions and FA

- On to step 2: convert the GNFA into a regular expression

– if normal-form GNFA has two states:



the regular expression R labeling the single transition describes the language recognized by the GNFA

January 14, 2025

CS21 Lecture 3-4

24

24

Regular expressions and FA

– if GNFA has more than 2 states:

– select one “ q_{np} ”; delete it; repair transitions so that machine still recognizes same language.
– repeat until only 2 states.

January 14, 2025 CS21 Lecture 3-4 25

25

Regular expressions and FA

– how to repair the transitions:
– for every pair of states q_i and q_j do

January 14, 2025 CS21 Lecture 3-4 26

26

Regular expressions and FA

– summary:
FA $M \rightarrow k$ -state GNFA $\rightarrow (k-1)$ -state GNFA
 $\rightarrow (k-2)$ -state GNFA $\rightarrow \dots \rightarrow 2$ -state GNFA $\rightarrow R$
– want to *prove* that this procedure is correct, i.e. $L(R) = \text{language recognized by } M$

- FA M equivalent to k -state GNFA ✓
- i -state GNFA equivalent to $(i-1)$ -state GNFA (we will prove...)
- 2-state GNFA equivalent to R ✓

January 14, 2025 CS21 Lecture 3-4 27

27

Regular expressions and FA

– **Claim:** i -state GNFA G equivalent to $(i-1)$ -state GNFA G' (obtained by removing q_{np})

– **Proof:**

- if G accepts string w , then it does so by entering states: $q_0, q_1, q_2, q_3, \dots, q_{\text{accept}}$
- if none are q_{np} then G' accepts w (see slide)
- else, break state sequence into runs of q_{np} :
 $q_0 q_1 \dots q_i q_{np} q_{i+1} \dots q_j q_{np} \dots q_k q_{l+1} \dots q_{\text{accept}}$
- transition from q_i to q_j in G' allows all strings taking G from q_i to q_j using q_{np} (see slide)
- thus G' accepts w

January 14, 2025 CS21 Lecture 3-4 28

28

Regular expressions and FA

January 14, 2025 CS21 Lecture 3-4 29

29

Regular expressions and FA

January 14, 2025 CS21 Lecture 3-4 30

30

Regular expressions and FA

– Proof (continued):

- if G' accepts string w , then every transition from q_i to q_j traversed in G' corresponds to
 - either
 - a transition from q_i to q_j in G
 - or
 - transitions from q_i to q_j via q_{ip} in G
- In both cases G accepts w .
- Conclude: G and G' recognize the same language.

January 14, 2025

CS21 Lecture 3-4

31

31

Regular expressions and FA

- **Theorem:** a language L is recognized by a FA iff L is described by a regular expr.
- Languages recognized by a FA are called **regular languages**.
- Rephrasing what we know so far:
 - regular languages closed under 3 operations
 - NFA recognize exactly the regular languages
 - regular expressions describe exactly the regular languages

January 14, 2025

CS21 Lecture 3-4

32

32

Limits on the power of FA

- Is every language describable by a sufficiently complex regular expression?
- If someone asks you to design a FA for a language that seems hard, how do you know when to give up?
- Is this language regular?
{ $w : w$ has an equal # of "01" and "10" substrings}

January 14, 2025

CS21 Lecture 3-4

33

33

Limits on the power of FA

- Intuition:
 - FA can only remember finite amount of information. They cannot **count**
 - languages that "entail counting" should be non-regular...
- Intuition not enough:
{ $w : w$ has an equal # of "01" and "10" substrings}
 $= 0\Sigma^*0 \cup 1\Sigma^*1$
but { $w : w$ has an equal # of "0" and "1" substrings} is not regular!

January 14, 2025

CS21 Lecture 3-4

34

34

Limits on the power of FA

How do you **prove** that there is **no** Finite Automaton recognizing a given language?

January 14, 2025

CS21 Lecture 3-4

35

35

Non-regular languages

- Pumping Lemma:** Let L be a regular language. There exists an integer p ("pumping length") for which every $w \in L$ with $|w| \geq p$ can be written as $w = xyz$ such that
1. for every $i \geq 0$, $xy^iz \in L$, and
 2. $|y| > 0$, and
 3. $|xy| \leq p$.

January 14, 2025

CS21 Lecture 3-4

36

36

Non-regular languages

- Using the Pumping Lemma to prove L is not regular:
 - assume L is regular
 - then there exists a pumping length p
 - select a string $w \in L$ of length at least p
 - argue that for every way of writing $w = xyz$ that satisfies (2) and (3) of the Lemma, pumping on y yields a string not in L.
 - contradiction.

January 14, 2025 CS21 Lecture 3-4 37

37

Pumping Lemma Examples

- Theorem: $L = \{0^n 1^n : n \geq 0\}$ is not regular.
- Proof:
 - let p be the pumping length for L
 - choose $w = 0^p 1^p$
$$w = \underbrace{000000000\dots}_{p} \underbrace{0111111111\dots}_{p} 1$$
 - $w = xyz$, with $|y| > 0$ and $|xy| \leq p$.

January 14, 2025 CS21 Lecture 3-4 38

38

Pumping Lemma Examples

– 3 possibilities:

$$w = \underbrace{000000000\dots}_{x} \underbrace{0111111111\dots}_{y} \underbrace{\dots}_{z} 1$$

$$w = \underbrace{000000000\dots}_{x} \underbrace{0111111111\dots}_{y} \underbrace{\dots}_{z} 1$$

$$w = \underbrace{000000000\dots}_{x} \underbrace{0111111111\dots}_{y} \underbrace{\dots}_{z} 1$$

– in each case, pumping on y gives a string not in language L.

January 14, 2025 CS21 Lecture 3-4 39

39

Pumping Lemma Examples

- Theorem: $L = \{w : w \text{ has an equal \# of 0s and 1s}\}$ is not regular.
- Proof:
 - let p be the pumping length for L
 - choose $w = 0^p 1^p$
$$w = \underbrace{000000000\dots}_{p} \underbrace{0111111111\dots}_{p} 1$$
 - $w = xyz$, with $|y| > 0$ and $|xy| \leq p$.

January 14, 2025 CS21 Lecture 3-4 40

40

Pumping Lemma Examples

– 3 possibilities:

$$w = \underbrace{000000000\dots}_{x} \underbrace{0111111111\dots}_{y} \underbrace{\dots}_{z} 1$$

$$w = \underbrace{000000000\dots}_{x} \underbrace{0111111111\dots}_{y} \underbrace{\dots}_{z} 1$$

$$w = \underbrace{000000000\dots}_{x} \underbrace{0111111111\dots}_{y} \underbrace{\dots}_{z} 1$$

– first 2 cases, pumping on y gives a string not in language L; 3rd case a problem!

January 14, 2025 CS21 Lecture 3-4 41

41

Pumping Lemma Examples

- recall condition 3: $|xy| \leq p$
- since $w = 0^p 1^p$ we know more about how it can be divided, and this case cannot arise:

$$w = \underbrace{000000000\dots}_{x} \underbrace{0111111111\dots}_{y} \underbrace{\dots}_{z} 1$$
- so we do get a contradiction.
- conclude that L is not regular.

January 14, 2025 CS21 Lecture 3-4 42

42

Pumping Lemma Examples

- Theorem: $L = \{0^i1^j : i > j\}$ is not regular.
 - Proof:
 - let p be the pumping length for L
 - choose $w = 0^{p+1}1^p$
- $w = \underbrace{000000000}_{p+1} \dots \underbrace{011111111}_{p} \dots 1$
- $w = xyz$, with $|y| > 0$ and $|xy| \leq p$.

January 14, 2025

CS21 Lecture 3-4

43

43

Pumping Lemma Examples

- 1 possibility:
 $w = \underbrace{00000}_{x} \underbrace{0000}_{y} \dots \underbrace{011111111}_{z} \dots 1$
- pumping on y gives strings in the language (?)
- this seems like a problem...
- Lemma states that for every $i \geq 0$, $xy^iz \in L$
- xy^0z not in L . So L not regular.

January 14, 2025

CS21 Lecture 3-4

44

44