

CS21 Decidability and Tractability

Lecture 3
January 11, 2008

January 11, 2008

CS21 Lecture 3

1

Outline

- Regular Expressions
- FA and Regular Expressions
- Pumping Lemma

January 11, 2008

CS21 Lecture 3

2

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 11, 2008

CS21 Lecture 3

3

Coming up...

- 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 11, 2008

CS21 Lecture 3

4

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 11, 2008

CS21 Lecture 3

5

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 11, 2008

CS21 Lecture 3

6

Some examples

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

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

January 11, 2008

CS21 Lecture 3

7

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 $+$, x ; \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 11, 2008

CS21 Lecture 3

8

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 11, 2008

CS21 Lecture 3

9

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 11, 2008

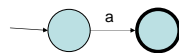
CS21 Lecture 3

10

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



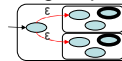
January 11, 2008

CS21 Lecture 3

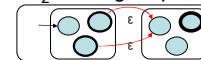
11

Regular expressions and FA

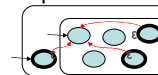
– $(R_1 \cup R_2)$, where R_1 and R_2 are reg. exprs.



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



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



January 11, 2008

CS21 Lecture 3

12

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 11, 2008

CS21 Lecture 3

13

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 it can be written

$$w = w_1 w_2 w_3 \dots w_k$$

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 11, 2008

CS21 Lecture 3

14

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

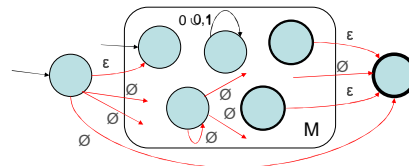
January 11, 2008

CS21 Lecture 3

15

Regular expressions and FA

- converting our FA M into GNFA in normal form:



January 11, 2008

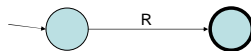
CS21 Lecture 3

16

Regular expressions and FA

- Onto 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

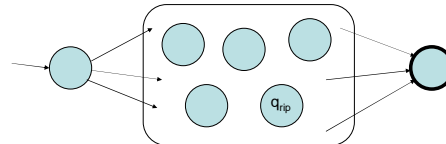
January 11, 2008

CS21 Lecture 3

17

Regular expressions and FA

– if GNFA has more than 2 states:



- select one " q_{rip} "; delete it; repair transitions so that machine still recognizes same language.
- repeat until only 2 states.

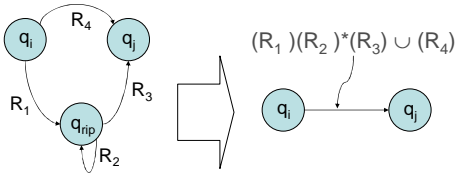
January 11, 2008

CS21 Lecture 3

18

Regular expressions and FA

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



January 11, 2008

CS21 Lecture 3

19

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. $L(R) =$ 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 11, 2008

CS21 Lecture 3

20

Regular expressions and FA

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

– **Proof:**

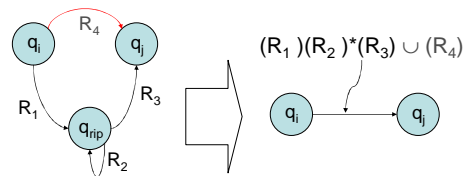
- if G accepts string w , then it does so by entering states: $q_0, q_1, q_2, q_3, \dots, q_{accept}$
- if none are q_{rip} then G' accepts w (see slide)
- else, break state sequence into runs of q_{rip} :
 $q_0 q_1 \dots q_i q_{rip} q_{rip} \dots q_{rip} q_j \dots q_{accept}$
- transition from q_i to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

January 11, 2008

CS21 Lecture 3

21

Regular expressions and FA

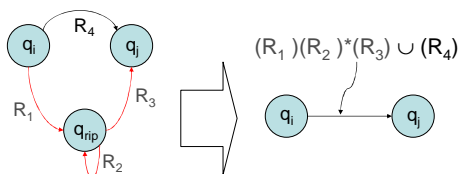


January 11, 2008

CS21 Lecture 3

22

Regular expressions and FA



January 11, 2008

CS21 Lecture 3

23

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_{rip} in G
- In both cases G accepts w .
- Conclude: G and G' recognize the same language.

January 11, 2008

CS21 Lecture 3

24

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 11, 2008

CS21 Lecture 3

25

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 11, 2008

CS21 Lecture 3

26

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 11, 2008

CS21 Lecture 3

27

Limits on the power of FA

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

January 11, 2008

CS21 Lecture 3

28

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

$$w = xyz \quad \text{such that}$$

1. for every $i \geq 0$, $xy^iz \in L$, and
2. $|y| > 0$, and
3. $|xy| \leq p$.

January 11, 2008

CS21 Lecture 3

29

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 11, 2008

CS21 Lecture 3

30

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 $w = 0^p 1^p$

$$w = \underbrace{000000000}_{p} \dots \underbrace{011111111}_{p} \dots 1$$

- $w = xyz$, with $|y| > 0$ and $|xy| \leq p$.

January 11, 2008

CS21 Lecture 3

31

Pumping Lemma Examples

- 3 possibilities:

$$w = \underbrace{000000000}_x \dots \underbrace{011111111}_{y} \dots \underbrace{1}_{z}$$

$$w = \underbrace{000000000}_x \dots \underbrace{011111111}_{y} \dots \underbrace{1}_{z}$$

$$w = \underbrace{000000000}_x \dots \underbrace{011111111}_{y} \dots \underbrace{1}_{z}$$

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

January 11, 2008

CS21 Lecture 3

32

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 $w = 0^p 1^p$

$$w = \underbrace{000000000}_p \dots \underbrace{011111111}_p \dots 1$$

- $w = xyz$, with $|y| > 0$ and $|xy| \leq p$.

January 11, 2008

CS21 Lecture 3

33

Pumping Lemma Examples

- 3 possibilities:

$$w = \underbrace{000000000}_x \dots \underbrace{011111111}_{y} \dots \underbrace{1}_{z}$$

$$w = \underbrace{000000000}_x \dots \underbrace{011111111}_{y} \dots \underbrace{1}_{z}$$

$$w = \underbrace{000000000}_x \dots \underbrace{011111111}_{y} \dots \underbrace{1}_{z}$$

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

January 11, 2008

CS21 Lecture 3

34

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:

$$w = \underbrace{000000000}_x \dots \underbrace{011111111}_{y} \dots \underbrace{1}_{z}$$

- so we do get a contradiction.

- conclude that L is not regular.

January 11, 2008

CS21 Lecture 3

35