CSE 311 Section 08

Structural Induction, Recursively Defined Sets, Regular Expressions, CFGs

Administrivia

Announcements & Reminders

- Midterm grades will be released shortly
- Homework 6 was due Wednesday (2/26)
- Homework 7 will be due Wednesday (3/5)
- Check your section participation grade on gradescope
    - If it different than what you expect, let your TA know

Structural Induction

Idea of Structural Induction

Every element is built up recursively…

So to show $P(s)$ for all $s \in S$…

Show $P(b)$ for all base case elements b.

Show for an arbitrary element not in the base case, if $P()$ holds for every named element in the recursive rule, then $P()$ holds for the new element (each recursive rule will be a case of this proof).

Structural Induction Template

Let $P(x)$ be "<predicate>". We show $P(x)$ holds for all $x \in S$ by structural induction.

Base Case: Show $P(x)$
[Do that for every base cases x in S.]

Inductive Hypothesis: Suppose $P(x)$ for an arbitrary x
[Do that for every x listed as in S in the recursive rules.]

Inductive Step: Show $P()$ holds for y.
[You will need a separate case/step for every recursive rule.]

Therefore $P(x)$ holds for all $x \in S$ by the principle of induction.

Problem 2 – Structural Induction on Trees

Definition of Tree:
Basis Step: • is a Tree.
Recursive Step: If L is a Tree and R is a Tree then Tree(•, L, R) is a Tree

Definition of leaves():
leaves(•) = 1
leaves(Tree(•, L, R)) = leaves(L) + leaves(R)

Definition of size():
size(•) = 1
size(Tree(•, L, R)) = 1 + size(L) + size(R)

Prove that leaves(T) ≥ size(T)/2 + 1/2 for all Trees T

Work on this problem with the people around you.

Problem 2 - Structural Induction on Sets

For x ∈ S, let P(x) be “”.
We show P(x) holds for all x ∈ S by structural induction on x.

<u>Base Case:</u> Show P(x) (for all x in the basis rules)

<u>Inductive Hypothesis:</u> Suppose P(x) (for all x in the recursive rules),
i.e. (IH in terms of P(x))

<u>Inductive Step:</u> Goal: Show that P(?) holds. (IS goal in terms of P(?))

<u>Conclusion:</u> Therefore P(x) holds for all x ∈ S by the principle of induction.

Problem 2 - Structural Induction on Trees

For a tree T, let P(T) be “leaves(T) ≥ size(T)/2 + 1/2”.
We show P(x) holds for all x ∈ S by structural induction on x.

Base Case: Show P(x) (for all x in the basis rules)

Inductive Hypothesis: Suppose P(x) (for all x in the recursive rules),
i.e. (IH in terms of P(x))

Inductive Step: Goal: Show that P(?) holds. (IS goal in terms of P(?))

Conclusion: Therefore P(x) holds for all x ∈ S by the principle of induction.

Problem 2 - Structural Induction on Trees

For a tree T, let P(T) be "leaves(T) ≥ size(T)/2 + 1/2".
We show P(T) holds for all trees T by structural induction on T.

Base Case: Show P(x) (for all x in the basis rules)

Inductive Hypothesis: Suppose P(x) (for all x in the recursive rules),
i.e. (IH in terms of P(x))

Inductive Step: Goal: Show that P(?) holds. (IS goal in terms of P(?))

Conclusion: Therefore P(x) holds for all x ∈ S by the principle of induction.

Regular Expressions

Regular Expressions

ε matches only the **empty string**

a matches only the one-character string ***a***

A ∪ **B** matches all strings that either **A** matches or **B** matches (or both)

AB matches all strings that have a first part that **A** matches followed by a second part that **B** matches

A* matches all strings that have any number of strings (even 0) that **A** matches, one after another (**ε** ∪ **A** ∪ **AA** ∪ **AAA** ∪ **...**)

Definition of the *language* matched by a regular expression

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

c) Write a regular expression that matches all binary strings that contain the substring "111", but not the substring "000".

We will do (a) together, then work on b-c

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

base-10 numbers:

Our everyday numbers! Notice we have 10 symbols (0-9) to represent numbers.

256: $(\mathbf{2} * 10^2) + (\mathbf{5} * 10^1) + (\mathbf{6} * 10^0)$

base-2 numbers: (binary)

10: $(\mathbf{1} * 2^1) + (\mathbf{0} * 2^0)$

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$$

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>101" or "<u>0</u>91" **are not** Base-10 numbers

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>101" or "<u>0</u>91" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>101" or "<u>0</u>91" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

$(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>101" or "<u>0</u>91" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

$(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>" **is** a Base-10 number not considered

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ “$\underline{0}$101” or “$\underline{0}$91” **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

$(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ “$\underline{0}$” **is** a Base-10 number not considered

All possible *strings* using numbers 0-9 that never start with 0 *or* is 0

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)*$

⚠️ "0101" or "091" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

$(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)*$

⚠️ "0" **is** a Base-10 number not considered

All possible *strings* using numbers 0-9 that never start with 0 *or* is 0

$0 \cup ((1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)*)$

Problem 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "$\underline{0}$101" or "$\underline{0}$91" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

$(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "$\underline{0}$" **is** a Base-10 number not considered

All possible *strings* using numbers 0-9 that never start with 0 *or* is 0

$0 \cup ((1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*)$

✅Generates only all possible Base-10 numbers

Problem 4 – Regular Expressions

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

Write a regular expression that matches all base-3 numbers

Problem 4 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

all binary strings that contain the substring “111”

Context-Free Grammars

Context-Free Grammars

A context free grammar (CFG) is a finite set of production rules over:

- An alphabet Σ of “terminal symbols”
- A finite set V of “nonterminal symbols”
- A start symbol (one of the elements of V) usually denoted S

Always think back to Regex!

- CFG to match RE **A ∪ B**

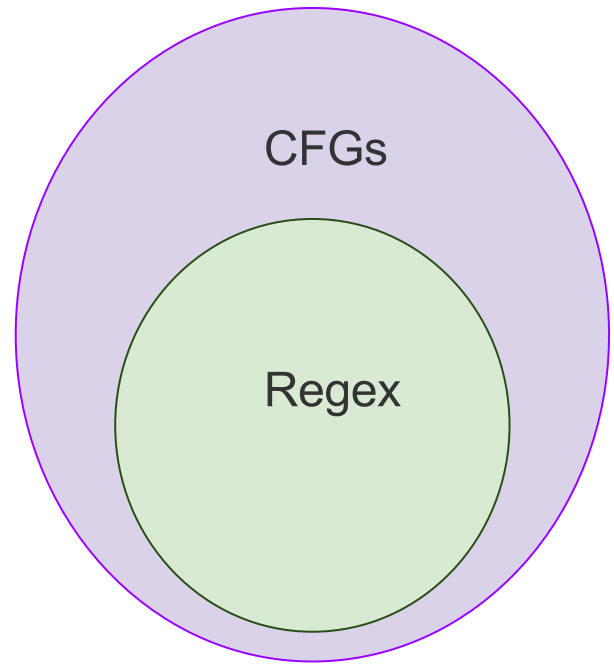
  $\mathbf{S \rightarrow S_1 \mid S_2}$ + rules from original CFGs

- CFG to match RE **AB**

  $\mathbf{S \rightarrow S_1 S_2}$ + rules from original CFGs

- CFG to match RE $\mathbf{A^*}$ (= $\varepsilon \cup \mathbf{A} \cup \mathbf{AA} \cup \mathbf{AAA} \cup \ldots$)

  $\mathbf{S \rightarrow S_1 S} \mid \varepsilon$ + rules from CFG with $\mathbf{S_1}$



Always think back to Regex!

- CFG to match RE **A ∪ B**

  $\mathbf{S} \rightarrow \mathbf{S_1} \mid \mathbf{S_2}$ + rules from original CFGs

- CFG to match RE **AB**

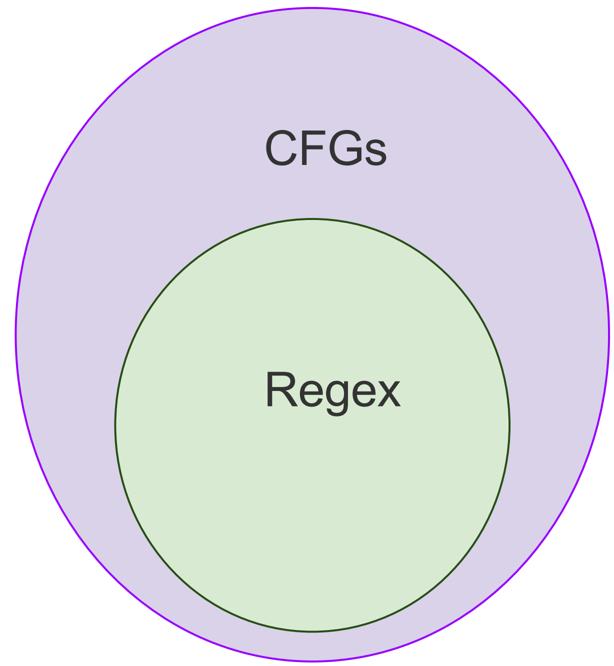
  $\mathbf{S} \rightarrow \mathbf{S}_1 \mathbf{S}_2$ + rules from original CFGs

- CFG to match RE **A*** (= ε ∪ **A** ∪ **AA** ∪ **AAA** ∪ ...)

  $\mathbf{S} \rightarrow \mathbf{S_1} \mathbf{S} \mid \varepsilon$ + rules from CFG with $\mathbf{S_1}$

CFG or Regex?

"equal number of 0's and 1's" (ex. 011010)



Always think back to Regex!

- CFG to match RE **A ∪ B**

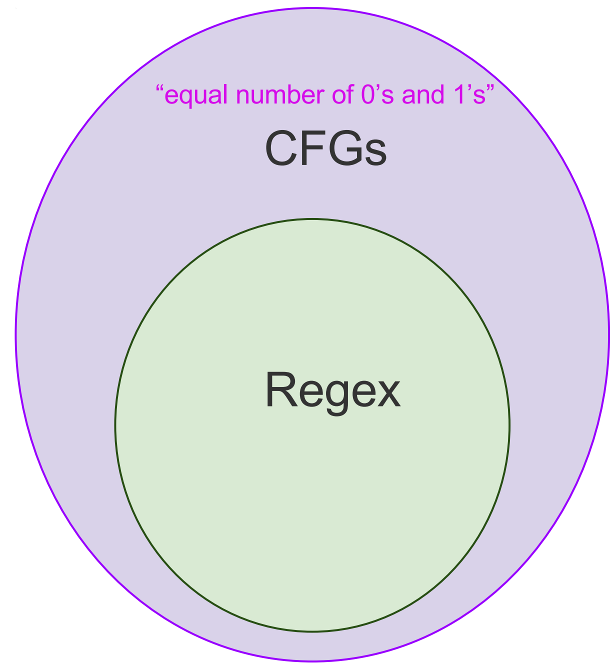
  $S \rightarrow S_1 \mid S_2$ + rules from original CFGs

- CFG to match RE **AB**

  $S \rightarrow S_1 S_2$ + rules from original CFGs

- CFG to match RE **A*** (= $\varepsilon \cup$ **A** $\cup$ **AA** $\cup$ **AAA** $\cup$ …)

  $S \rightarrow S_1 S \mid \varepsilon$ + rules from CFG with S_1



Problem 5 – CFGs

Write a context-free grammar to match each of these languages.

a) All binary strings that start with 11.

b) All binary strings that contain at most one 1.

Work on this problem with the people around you.

Problem 5 – CFGs

a) All binary strings that start with 11.

Problem 5 – CFGs

a) All binary strings that start with 11.

Thinking back to regular expressions…

Problem 5 – CFGs

a) All binary strings that start with 11.

Thinking back to regular expressions…

$11\ (0 \cup 1)^*$

Problem 5 – CFGs

a) All binary strings that start with 11.

Thinking back to regular expressions…

$11\ (0 \cup 1)^*$

Now generate the CFG…

Problem 5 – CFGs

a) All binary strings that start with 11.

Thinking back to regular expressions…

$11\ (0 \cup 1)^*$

Now generate the CFG…

$\mathbf{S} \rightarrow 11\mathbf{T}$
$\mathbf{T} \rightarrow 1\mathbf{T} \mid 0\mathbf{T} \mid \boldsymbol{\varepsilon}$

Problem 5 – CFGs

b) All binary strings that contain at most one 1.

Problem 5 – CFGs

b) All binary strings that contain at most one 1.

Thinking back to Regular expressions…

Problem 5 – CFGs

b) All binary strings that contain at most one 1.

Thinking back to Regular expressions…

$0^* (1 \cup \boldsymbol{\varepsilon})\, 0^*$

Problem 5 – CFGs

b) All binary strings that contain at most one 1.

Thinking back to Regular expressions…

$0^* \; (1 \cup \boldsymbol{\varepsilon}) \; 0^*$

Now generate the CFG…

Recursively Defined Sets

Recursive Definition of Sets

Define a set S as follows:

Basis Step:
Describe the basic starting elements in your set
ex: $0 \in S$

Recursive Step:
Describe how to derive new elements of the set from previous elements
ex: If $x \in S$ then $x + 2 \in S$.

Exclusion Rule: Every element of S is in S from the basis step (alone) or a finite number of recursive steps starting from a basis step.

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

b) Binary strings not containing 10.

c) Binary strings not containing 10 as a substring and having at least as many 1s as 0s.

d) Binary strings containing at most two 0s and at most two 1s.

Work on this problem with the people around you.

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Generate accepted and rejected strings first!

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Generate accepted and rejected strings first!

Step 1: Write out basic cases and more intricate cases

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Generate accepted and rejected strings first!

Step 1: Write out basic cases and more intricate cases

Accepted Strings	Rejected Strings
ε	0
11	1
10101010	101010**1**
10101011	**0**10101011

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Generate accepted and rejected strings first!

Step 1: Write out basic cases and more intricate cases

Step 2: Find a pattern!

Accepted Strings	Rejected Strings
ε	0
11	1
10101010	101010**1**
10101011	**0**10101011

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Generate accepted and rejected strings first!

Step 1: Write out basic cases and more intricate cases

Accepted Strings	Rejected Strings
ε	0
11	1
10101010	101010**1**
10101011	**0**10101011

Step 2: Find a pattern!

All even-length strings can be generated from **a series of substrings** of **length 2!**

All possible substrings of length 2 are:
10, 01, 11, 00

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Step 3: Write out Basis and Recursive step

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Step 3: Write out Basis and Recursive step

Basis: $\varepsilon \in S$

Recursive Step: If $x \in S$, then $x00, x01, x10, x11 \in S$

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Step 3: Write out Basis and Recursive step

Step 4: check that you cannot build the rejected strings and only build accepted strings with the recursive step

Basis: $\varepsilon \in S$

Recursive Step: If $x \in S$, then $x00, x01, x10, x11 \in S$

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

Step 3: Write out Basis and Recursive step

Basis: $\varepsilon \in S$

Recursive Step: If $x \in S$, then $x00, x01, x10, x11 \in S$

Step 4: check that you cannot build the rejected strings and only build accepted strings with the recursive step

Accepted Strings	Rejected Strings
ε	0
11	1
10101010	101010**1**
10101011	**0**10101011

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

b) Binary strings not containing 10.

Problem 3 – Recursively Defined Sets

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

c) Binary strings not containing 10 as a substring and having at least as many 1s as 0s.

That’s All, Folks!

Thanks for coming to section this week!
Any questions?