CS 3100, Models of Computation, Spring 2020, Lec 2

Ganesh Gopalakrishnan
School of Computing
University of Utah
Salt Lake City, UT 84112

https://bit.ly/3100S20Syllabus

Lecture 2, Covering Chapter 3

(Please remind me to turn on recording before the lecture begins)

Pattern to expect for all future lectures

- Required reading to be done before class
  - Class is for review + detailed examples + Q/A
- Watch posted videos for lecture
  - This helps you plan your reading + try out things beforehand
- Pace will pick up
  - Also, non-trivial material will appear – will give you sufficient notice
- Asg-1 has been posted
  - Mainly “worked out” problems
  - Fast-track your learning of basic Python (just 4-5 simple ideas), colab (how to use it) and reading Ch 2-3
  - Easy, but get started soon, as there will be no deadline extensions in this course
  - Submit early, Submit Often, ask questions!

Recap: Match these in all possible ways

- Symbol : string-of-length-one
- String : sequence of Symbols
- Alphabet : set of Symbols
- Language : set of Strings

- Set : an arbitrary set (no restrictions)

$\{\}$

$\{a\}$

$\{0, 1\}$

$\{\varepsilon\}$

$\{\emptyset\}$

$\{a, aa\}$

$\{a, 1, \{\}, \{\{aa\}\}\}$

Operations on objects (of various types)

- Avoid these
  - type error!

$$\emptyset \; \varepsilon = \varepsilon \; \emptyset = nonsense$$

- This one is OK

$$\emptyset \; \{\varepsilon\} = \{\varepsilon\} \; \emptyset = (makes \; sense)$$

Think of all operations as having types

- Language union and intersection have type Lang x Lang -> Lang
- Language concatenation has type Lang x Lang -> Lang
- Zero of language concatenation:
  - What that language – call it Zero – such that
    - Zero L = L Zero = Zero ?
- One of language concatenation
  - What is that language – call it One – such that
    - One L = L One = L ?

Assessment: Language Concatenation

$$L_1 = \{x : x \in 0^{2i+1},\ i \geq 0\}$$

$$L_2 = \{x : x \in 0^{2i},\ i \geq 0\}$$

Express $L_1 L_2$ in English

$$L_2 L_2 = ?$$

$$L_1 L_1 = ?$$

$$L_1 L_2 = ?$$

Is this true ??

$$L_1 L_1 = L_2 ?$$

Assessment 1: Find the difference

What is this language

$$L_1 = \{ (^n)^n \ : n \geq 0 \}$$

- How is the above language different from the following?

L_balPar = { x : x is a string of “well balanced parentheses” } ?

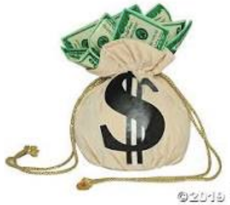
- Is L1 included in L_balPar or vice-versa ??
  - If so, which way does the inclusion hold?

Assessment 2: Same or different?

$$\{\ (^i\ )^i\ :\ i \geq 0\} =? = \{\ (^i : \ i \geq 0\}\ \{\ )^i\ :\ i \geq 0\}$$

If concatenation is like multiplication, then doing concatenation many times is like …. ?

If concatenation is like multiplication, then doing concatenation many times is like …. Exponentiation



Choose the right answer below (write it…)

If we now view

language concatenation

as multiplication, then

Exponentiation is

Repeated multiplication

$$L^n = LL^{n-1}$$

Which of these two must be true ??

$$L^0 = \emptyset?$$

$$L^0 = \{\varepsilon\}?$$

Answer for Language Concatenation Definition

$$L^n = LL^{n-1}$$

$$L^0 = \{\varepsilon\} \ !$$

Much like with numbers, 33 raised to 0 is 1

Likewise, a “language raised to Zero is the One language”

Interesting facts : let alphabet Sigma = {0,1}

- {‘’} = all strings of length 0
- {0,1} = all strings of length 1
- {0,1} {0,1} = all strings of length 2
- {0,1} {0,1} {0,1} = all strings of length 3
- …

Defining lexp in Python (`language exp')

```python
def exp(L,n):
    """Exponentiate a language.
       If A = set(['ab', 'bc']) is a language, then
       exp(A,2) --> set(['abab', 'bcab', 'bcbc', 'abbc'])
    """
    return Unit() if n == 0 else cat(L, exp(L, n-1))
```

$$L^0 = \{\varepsilon\}$$

$$L^n = LL^{n-1}$$

```python
def Unit():
    """This is the UNIT language for concatenation,
       when concatenation is viewed as multiplication.
    """
    return {""} # Set with epsilon
```

Now…. for the ‘star’ of the show !!

i.e. Kleene Star

Purpose of Kleene-Star (or “star”)

- Divide-up a language design problem into manageable pieces
- Design one language (and its machine) for the BLOCK
- Put a WHILE around the BLOCK
  - ZERO or More Identifiers declared in C
  - Zero or more files whose names begin with ’a’
  - ”rm a*” → You are using Kleene-star or Star
    - Remove zero or more files beginning with ‘a’

Suppose you want to say
"I want ALL strings of length 0 OR
ALL strings of length 1 OR
ALL strings of length 2 OR

…" Here is a start – finish it !!!

{0,1}

{0,1} {0,1}

{0,1} {0,1} {0,1}

… ?

Suppose you want to say
"I want ALL strings of length 0 OR
ALL strings of length 1 OR
ALL strings of length 2 OR
…"

Does this do it ?

$\{0,1\} \qquad \cup$

$\{0,1\}\,\{0,1\} \qquad \cup$

$\{0,1\}\,\{0,1\}\,\{0,1\} \qquad \cup$

… ?

Suppose you want to say
“I want ALL strings of length 0 OR
ALL strings of length 1 OR
ALL strings of length 2 OR
..."

NO, Don’t forget the Unit Language for Concatenation !! i.e. the set with Epsilon

{‘’} $\cup$

{0,1} $\cup$

{0,1} {0,1} $\cup$

{0,1} {0,1} {0,1} $\cup$

… !!

Definition of Star (three equivalent ones)

Star, Definition 1: $L^* = L^0 \cup L^1 \cup L^2 \cup \ldots$

Star, Definition 2: $L^* = \bigcup_{i=0}^{\infty} L^i$

Star, Definition 3: $L^* = \{x : \exists k \in Nat, x \in L^k\}$

"Pick your poison..."

Star as a limit (handy for coding, understanding)

$$L_n^* = L^n \cup L_{n-1}^*$$

$$L_0^* = \{\varepsilon\}$$

```
def lunion(L1,L2):
    """Language union
    """
    return L1 | L2
```

```
def star(L,n):
    """Star a language, bounding the iteration to the given n.
    If A = set(['ab', 'bc']) is a language, then
    star(A,2) --> set(['abab', 'bcbc', 'ab', 'abbc', '',
                       'bc', 'bcab']).
    """
    return Unit() if n == 0 else lunion(exp(L,n), star(L,n-1))
```

The star of ANY language CONTAINS ε

$$\{0\}^* = \{0\}^0 \cup \{0\}^1 \cup \{0\}^2 \ldots$$

$$\{0,1\}^* = \{0,1\}^0 \cup \{0,1\}^1 \cup \{0,1\}^2 \ldots$$

$$\{\}^* = \{\}^0 \cup \{\}^1 \cup \{\}^2 \ldots$$

CRUCIAL OBSERVATION !!!!!

- What is the universal language over an alphabet, say Sigma?
  - E.g. Sigma = {0,1} ; what is the universal language over it?
    - Express it in terms of Star!
- So now, what does it mean to complement a language L, given a Sigma?
  - What language are we creating?
  - Express in terms of Sigma, Star, and set subtraction

CRUCIAL OBSERVATION !!!!!

- For a language L and a given alphabet Σ, the complement of L , written $\overline{L}$ is nothing but $\Sigma^* - L$

- In Lecture-1 we had not introduced the star operator; there we called this universal set "U". WE SHALL NO LONGER USE THAT NOTATION, NOW THAT WE HAVE DEFINED STAR!

The star of ANY language CONTAINS ε

$$0^* = \{\varepsilon, 0, 00, 000, \ldots\}$$

$$\{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, \ldots\}$$

$$\{\}^* = \{\varepsilon\} \cup \{\} \cup \{\} \ldots = \{\varepsilon\}$$

One minute quiz: Property of Star

- There are just two languages L1 and L2 such that
  - L1* and L2* are finite
  - L1 = ?
  - L2 = ?
- The Star of any other language is an infinite language

On listing strings in a language

- We often need to list strings from a language
  - To feed the strings to some tool
  - To show someone what the language contains (by way of example)
- How do we ”smartly list” strings from an infinite language

How do you list a language “smartly”?

- WRONG WAY:
  - Choose a method that guarantees that you will NEVER list some string

- RIGHT WAY:
  - Choose a method that guarantees that every string will be EVENTUALLY listed

  - Such methods of listing are called **enumeration**

How do you list a language “smartly”?

- WRONG WAY:
  - Choose a method that guarantees that you will NEVER list some string
    - Called Lexicographic Order of listing
    - This is NOT an enumeration
- RIGHT WAY:
  - Choose a method that guarantees that every string will be EVENTUALLY listed
    - Called Numeric Order of enumeration

Example of enumeration and non-enumeration

Numeric order : This is an enumeration (gets to any string eventually, i.e. in a FINITE NUMBER of steps)

$$\{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, \ldots\}$$

Lexicographic order : NOT an enumeration. Does not guarantee to EVER get to some strings (anything containing 1 won't be listed !!)

$$\{0,1\}^* = \{\varepsilon, 0, 00, 000, 0000, 00000, \ldots\} \text{ DONT DO THIS!}$$

See the book for code that enumerates

See the book for code that enumerates

- Code to enumerate in numeric order
  - See book
- Code to list in lexicographic order
  - It is easy to list strings in lexicographic order
    - But this is useless for most purposes (not an enumeration)

Language Reversal

- Reverse each string in the language

String and Language Reversal

$$(abc)^R = cba$$

$$\{a, ab, aa, abc\}^R = \{a, aa, ba, cba\}$$

```
def revs(S):
    """Reverse a string.
      revs('ab') --> 'ba'
    """
    return S[::-1]


def revl(L):
    """Reverse a language.
      revl(set(['ab', 'bc'])) --> set(['cb', 'ba'])
    """
    return set(map(lambda x: revs(x), L))
```

1-min Exercise
Set Comprehension for Palindromes

- Write below a set comprehension listing all palindromes over {0,1}
  - Allowed to use notations are
    - w^R for the reverse of w
    - w1 w2 for concatenation
- Call it Pal

1-min Exercise
Set Comprehension for Palindromes

- Is star(Pal) = Pal ?
  - Justify your answer
- Is Pal Pal = Pal ?

More exercises

- Pal Sigma* = Sigma* Pal ??
- Pal U Sigma* = Sigma* U Pal ??

What this course is about: STRUCTURE of information in strings

- Is Pal Sigma* = Sigma* Pal ?
  - Answer:

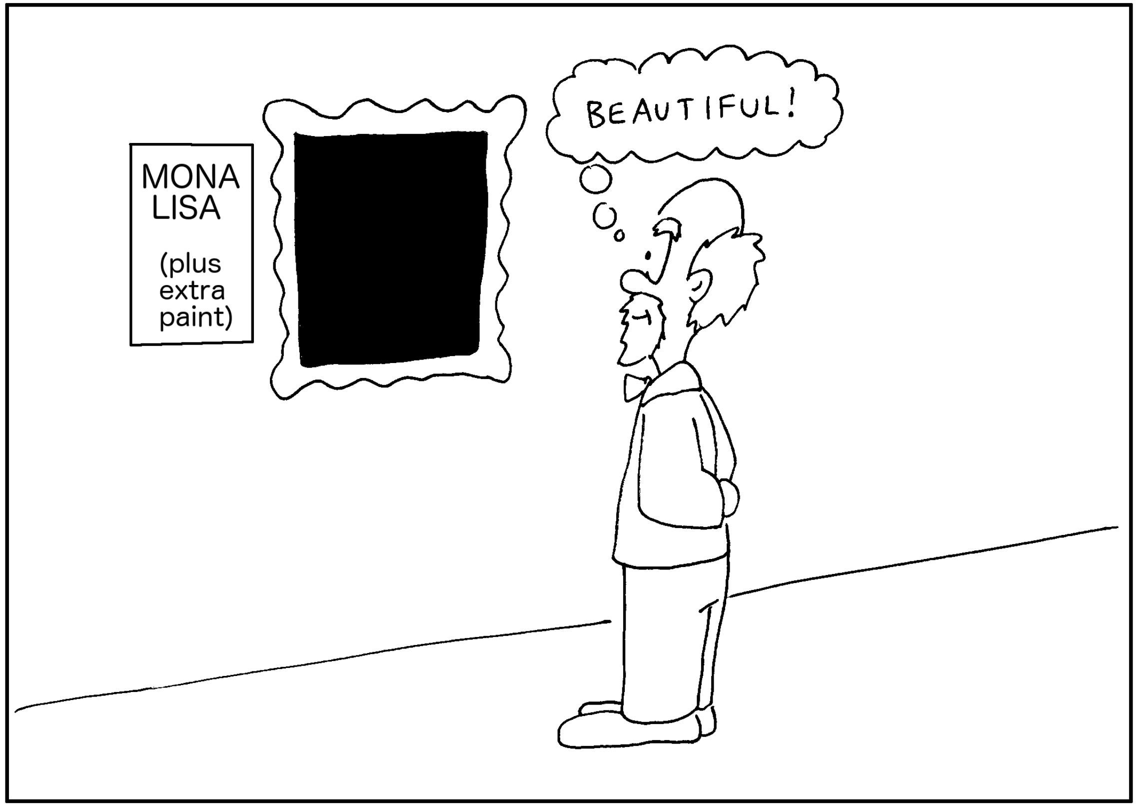
- Is Pal U Sigma* = Sigma* U Pal ?
  - Answer:

What this course is about: STRUCTURE of information in strings

-

- Pal U Sigma* = Sigma* U Pal
  - Yes!
  - Union with Sigma* blends structure away !!

  (destroys information)



Compare sizes |{a,ab}{a,ab}| vs. |{‘’,a}{‘’,a}|

Which is bigger?

Ex: (({a,ab}{a,ab})^R)^*

Here, juxtaposition is concatenation, ^R is reverse, and ^* is Kleene star
If this is an infinite set, then

- write 6 elements in numeric order
- Write 6 elements in lexicographic order
- Space for answer below:

Drill Problems to try soon

Ex1 Show that $L^* = L^{**}$

$L^* \subseteq L^{**}$ because L^{**} has $(L^*)^1$.

$$L^{**} = \{x : \exists k \cdot x \in (L^*)^k\}$$

show that $\forall k \geq 1 \cdot (L^*)^k = L^*$

for $y \in (L^*)^k$

$$y \in L^{0^k} \vee L^{1^k} \vee L^{2^k} \vee .. L^{p^k}$$

$$L^{p^k} = L^{pk}$$

$$y \in L^{pk} \Rightarrow y \in L^*.$$

Drill Problems to try soon

Ex 2

Let $M = \{\varepsilon, a, b\}$

Show that

$$M \neq MM$$

Ex 3

Show that

(a) If $\varepsilon \notin L$

then $L \neq LL$

(b) If $\{\varepsilon, a, b\} \subseteq L$ then

$L = LL$ iff $L = \{a, b\}^*$

Extra Slides

Plans for using Jove which helped during F'19
(feedback from F'19 hopefully will improve things)

- Students did face issues wrt the Windows platform
- Mac install was smooth
- Colab is given to you to help accelerate w/o any installation
- We will give you practice during Asg-1 (based on feedback F'19)