PPL 2/17

Reagan Shirk

February 17, 2020

Midterm

- We can use any printed materials as reference on the exam
- No electronic devices

Parsing

Class	Direction of Scanning	Derivation Discovered	Parse Tree Construction	Algorithms Used
LL	Left-to-right	Left-most	Top-down	Predictive
LR	Left-to-right	Right-most	Bottom-up	Shift-reduce

Top-down and Bottom-up Parsing

- Going over the string "A, B, C;", with the grammar:

$$\begin{aligned}\text{id-list} &\rightarrow \text{id id-list-tail}\\ \text{id-list-tail} &\rightarrow \text{, id id-list-tail}\\ \text{id-list-tail} &\rightarrow \text{;}\end{aligned}$$

- This grammar is predictive because each right hand side of the rule starts with a new character, it is right recursive
  - I think that's what he said
- How do we make this left recursive?

$$\begin{aligned}\text{id-list} &\rightarrow \text{id-list}\\ \text{id-list} &\rightarrow \text{ID}\\ \text{id-list} &\rightarrow \text{ID, id-list}\end{aligned}$$

- If we move the recursion to the first rule, we have leftmost recursion. What is written above is rightmost recursion
- I was doing really well with paying attention until I got distracted by looking at new devices

LL Parsing

- At the end of the day, parsing is about reading one token and checking whether the token was expected or not
  - This is what parsing is doing for all compilers
- The differences we are talking about are different ways to know what to expect
- With LL Parsing, tokens are processed in a loop
  - It repeatedly looks up an action in a two-dimensional table
  - The table is based on the current leftmost non-terminal and the current input token

  - The actions are:
    * Matching a terminal
    * Predicting a production
    * Announcing a syntax error
- Why can't I focuuuussssss
- There are some issues with LL Parsing:
  - We want to avoid left-recursion because it translates to having recursive calls before even showing/defining your end case
  - We want to remove common prefixes like when one terminal has three variables that all start the same

$$\text{expr} \rightarrow \text{term + expr}$$
$$\text{expr} \rightarrow \text{term - expr}$$
$$\text{expr} \rightarrow \text{term}$$

  - We want to avoid dangling else statements
  - Important to note that getting rid of the left-recursion does **not** make a grammar LL