Quiz 2 on Tuesday Nov 12

Context free grammars

CFGs

1. Predict parse tree
2. Ambiguous? Or not?
3. Difference b/w grammars

Regular expressions

4. Does w match r?
5. Write down r.
6. Draw the board games

state machines

NFA

Interpreter design

Regular expressions — CFG

ocamllex — menhir

NFA — LR(1)

Q: Grammar: $e ::= INT \mid e + e$

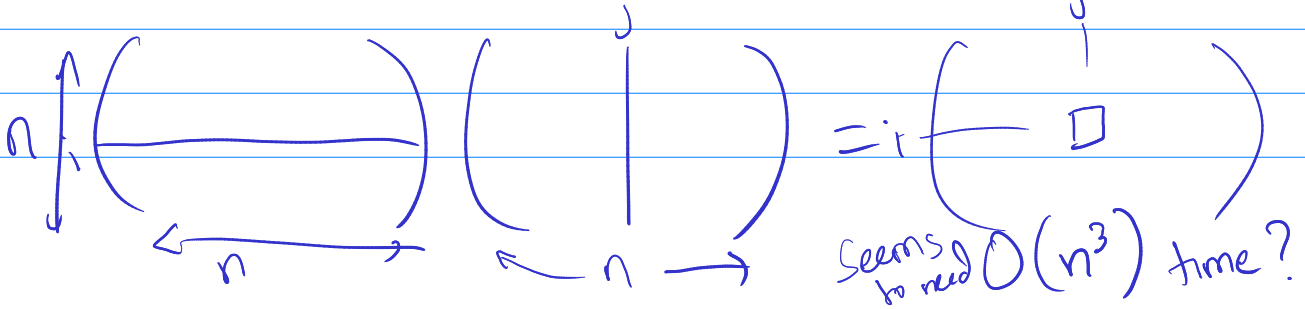
Is "34 + +" a legal production of this grammar?

Fix a CF grammar G.

I give you a string w.

→ You will need $\Omega(|w|^2)$ time to check if w matches G.

→ Q: What is the fastest alg for matrix multiplication?



Stupid claim: There cannot be a mat mul alg that runs in $o(n^2)$ time.

(little oh)

Strassen's result: Mat mul can be calculated in time $O(n^{\log_2 7})$

Q: How fast can you multiply matrices?

Nobody knows $O(n^\omega)$ $2 \leq \omega < \log_2 7$

CFG Parsing $\xrightarrow{\text{Valiant (1967)}}$ Boolean matrix multiplication $O(n^\omega)$

$\xleftarrow{\text{Lee (2002)}}$

$e ::= Int$

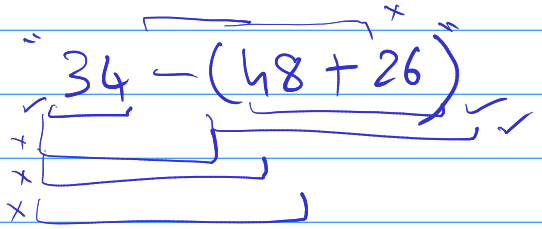
$\mid e_1 + e_2$

$\mid e_1 - e_2$

$\mid (e)$

Arbitrary CFG

$O(|G|^2)$ → Chomsky normal form



$r ::= \varepsilon \mid a \mid r_1 + r_2 \mid r_1 \cdot r_2 \mid r^*$

$\frac{\partial (ab)^*}{\partial a}$ (Brzozowski) Derivative

(Ultra-) Basic Type Theory

static kind.

↓

Define the shape of data.

Calculators with conditionals — L_0 : Untyped expressions

$$e ::= \overline{Int} \mid \overline{e_1 + e_2} \mid \overline{e_1 - e_2} \mid \underline{true} \mid \underline{false} \mid \underline{e_1 \leq e_2} \mid \text{if } e_1 \text{ then } e_2 \text{ else } e_3$$

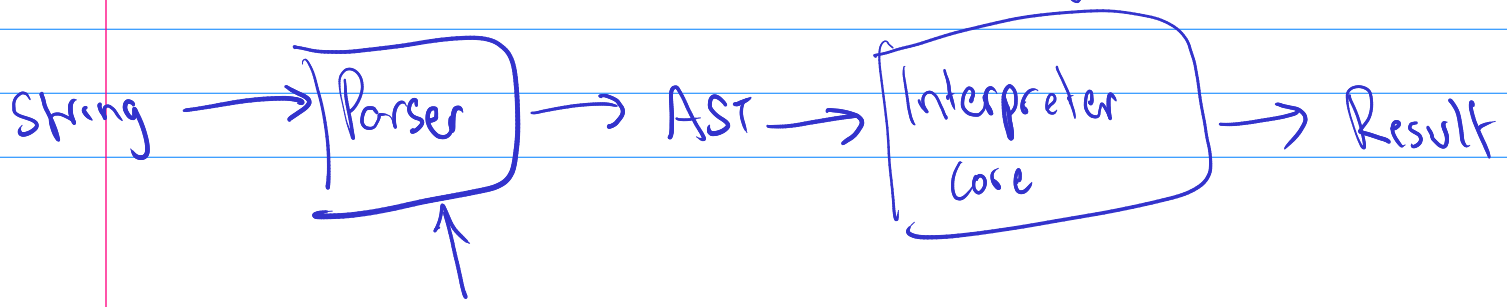
"if $3 \leq 4$ then $8+9$ else $9+16$"

"if 3 then $8+9$ else $9+16$"

L_1 (Separate ints & bools)

$$e ::= Int \mid e_1 + e_2 \mid e_1 - e_2 \mid \text{if } b \text{ then } e_1 \text{ else } e_2$$

$$b ::= true \mid false \mid e_1 \leq e_2 \mid \text{if } b_1 \text{ then } b_2 \text{ else } b_3$$



L_2: Language of lists

$e ::= int \mid e_1 + e_2 \mid e_1 - e_2 \mid len(e)$
$\mid [\,] \mid e :: e \mid hd(e) \mid tl(e)$

$len([\,])$ ✓ $\quad 2 + len([\,])$ ✓

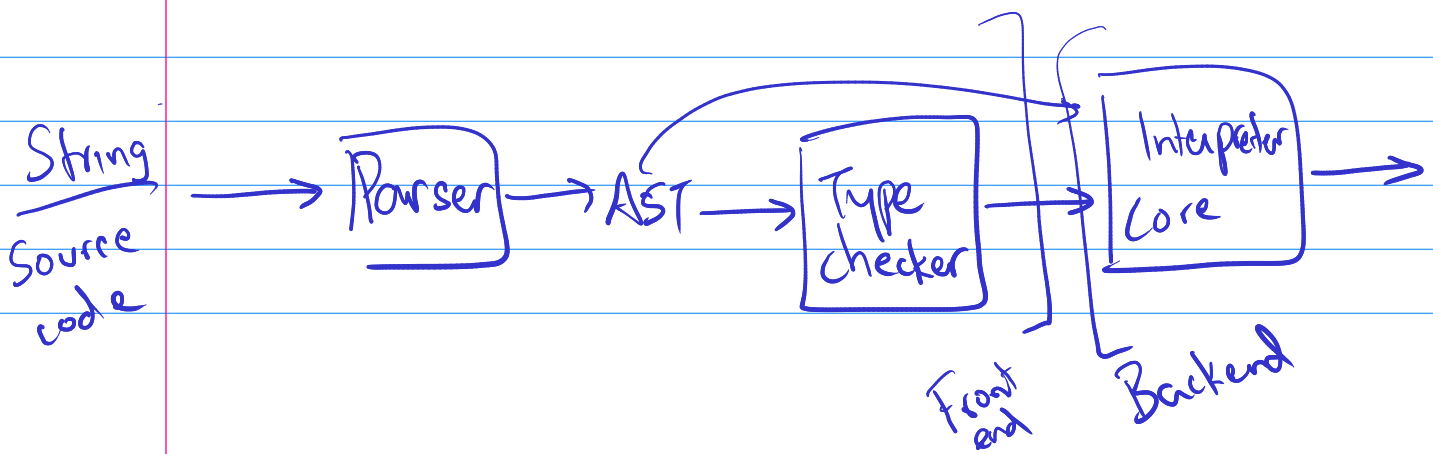
$len(3 :: [\,])$ ✓

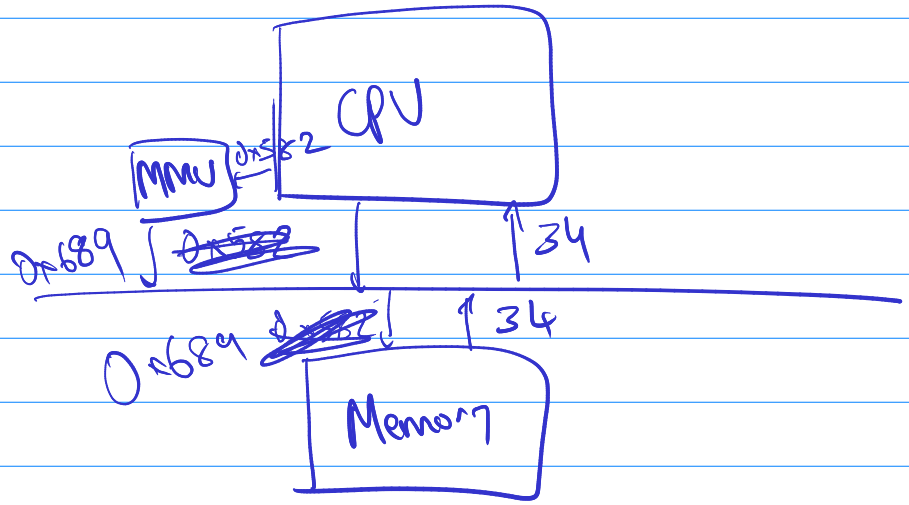
$[\,] + [\,]$ ✗

$e ::= int \mid e_1 + e_2 \mid e_1 - e_2 \mid len(l) \mid hd(l)$

$l ::= e :: l \mid [\,] \mid tl(l) \mid hd(ll)$

$1, L[1], L[L[1]], L[L[L[1]]]$




CPU
MMU
0x682
0x689
34
0x684
34
Memory