

CSE 401/M501 – Compilers

Section 4: CUP and LR parsing
Nate Yazdani & Aaron Johnston
Spring 2018

Administrivia

- Homework 2 is due tonight!
 - You have late days if you need them
- Parser is due one week from today
- Scanner feedback before the weekend
 - Be sure to check when debugging parser 😊

Agenda

- CUP tips, tricks, and demo
- LL parsing
 - See Sec. 3.3 of Cooper & Torczon for more
- A fun worksheet!

The CUP parser generator

- Uses LALR(1)
 - Weaker but faster variant of LR(1)
- LALR is more sensitive to ambiguity than LR
- CUP can resolve some ambiguities itself
 - Precedence for reduce/reduce conflicts
 - Associativity for shift/reduce conflicts
- If you use those features, read the docs carefully

The CUP parser generator

Demo: testing and debugging a CUP parser

LL(k) parsing

- LL(k) scans left-to-right, builds leftmost derivation, and looks ahead k symbols
- Typically $k = 1$, just like LR
- LL(1) requires for every nonterminal A ...
 - $\bigcap_{\alpha \in \text{RHS}(A)} \text{FIRST}(\alpha) = \emptyset$
 - $\text{nullable}(A) \Rightarrow \text{FIRST}(A) \cap \text{FOLLOW}(A) = \emptyset$
- Those restrictions enable the parser to choose productions correctly with 1 symbol of look-ahead
- We can transform a grammar to satisfy them

Factoring out common prefixes

When multiple productions of a nonterminal share a common prefix, turn the different suffixes (“trails”) into a new nonterminal.

Greeting ::= “hello, world” | “hello, friend” | “hello, ” *Name*
Name ::= “Sarah” | “John” | ...

Greeting ::= “hello, ” *Address*
Address ::= “world” | “friend” | *Name*
Name ::= “Sarah” | “John” | ...

Removing direct left recursion

When a nonterminal has left-recursive productions, turn the different suffixes ("trails") into a new nonterminal, appended to the remaining productions.

$$Sum ::= Sum \text{ "+" } Sum \mid Sum \text{ "-" } Sum \mid Constant$$
$$Constant ::= \text{"1"} \mid \text{"2"} \mid \text{"3"} \mid \dots$$
$$Sum ::= Constant SumTrail$$
$$SumTrail ::= \text{"+" } Sum \mid \text{"-"} Sum \mid \varepsilon$$
$$Constant ::= \text{"1"} \mid \text{"2"} \mid \text{"3"} \mid \dots$$

Removing indirect left recursion

- Pseudocode from Cooper & Torczon:

```
impose an order on the nonterminals,  $A_1, A_2, \dots, A_n$   
for  $i \leftarrow 1$  to  $n$  do;  
  for  $j \leftarrow 1$  to  $i - 1$  do;  
    if  $\exists$  a production  $A_i \rightarrow A_j \gamma$   
      then replace  $A_i \rightarrow A_j \gamma$  with one or more  
        productions that expand  $A_j$   
  end;  
  rewrite the productions to eliminate  
    any direct left recursion on  $A_i$   
end;
```

■ FIGURE 3.6 Removal of Indirect Left Recursion.

- Rather conservative: no need to push A_j into A_i if you know that $A_j \not\Rightarrow \alpha A_i \beta$ for any α, β

Worksheet time

- Feel free to discuss and work in small groups
- Reminders:
 - $\text{FIRST}(\alpha)$ is the set of terminal symbols that can begin a string derived from α
 - $\text{FOLLOW}(A)$ is the set of terminal symbols that may immediately follow A in a derived string
 - $\text{nullable}(A)$ is whether A can derive ε