LL parsing example (Conclusion)

Recap

- HW on flex due tomorrow via git
- Office hours Thursday:
  11-noon
  (not noon-1pm)

  I'm also around after class today.

- Next HW: over parsing / CFGs.

Last time:

One simple yet useful CFG class: LL(1)

left-to-right

leftmost derivation

1 token of look-ahead

Meaning: Keep current (partial) parsing as well as input string.

Algorithm: Look at leftmost nonterminal and next token in input string.

Use these to determine which rule to apply.

(If more than 1 choice, try one, + if stuck, back up + try next.)

Key: Build a table to help make decision

FIRST & FOLLOW Sets (for LL(1)):

FIRST(α) ← any string of non-terminals & terminals

:= set of possible first terminals in any derivaton of α by the grammar

So:

1) if x is a terminal,

$$\text{FIRST}(x) = \{x\}$$

2) if $X \to \varepsilon$ is a production, add ε to FIRST(x)

3) If X is a nonterminal:

If $X \to Y_1 Y_2 \dots Y_k$ is a production:

- Everything in FIRST(Y_1) is in FIRST(X)
- add a if a is in FIRST(Y_i) and ε is in FIRST(Y_1), …, FIRST(Y_{i-1})
- add ε if ε is in FIRST(Y_1), … FIRST(Y_k)

Ex:

$$\left.\begin{aligned}
S &\to E \\
E &\to TE' \\
E' &\to +TE' \mid \varepsilon \\
T &\to FT' \\
T' &\to \underline{*}FT' \mid \varepsilon \\
F &\to (E) \mid id
\end{aligned}\right]$$

addition & mult grammar

$\text{FIRST}(S) = \{(, id\}$

$\text{FIRST}(E) = \{(, id\}$

$\text{FIRST}(E') = \{+, \varepsilon\}$

$\text{FIRST}(T) = \{\overbrace{(, id}^{F(F)}\}$

$\text{FIRST}(T') = \{*, \varepsilon\}$

$\text{FIRST}(F) = \{(, id\}$

Follow Sets:

(We'll assume any input ends in $, just to have an end of file character)

Rules:

1) Put $ in FOLLOW(S) where S is start symbol. ✓

2) Given a production: ✓

$$A \to \underline{\alpha B \beta}$$

everything in FIRST(β) goes in FOLLOW(B) (except ε, if it is there).

3) Given a production:

$$A \to \alpha B$$

or $\underline{A \to \alpha B \beta}$ with $\varepsilon \in$ FIRST(β)

then everything in FOLLOW(A) also goes in FOLLOW(B)

$S \to E$ ←

Ex: $\underline{E} \to \underbrace{TE'}$

$E' \to \underbrace{+TE'} \mid \varepsilon$ 2 ←

$T \to FT'$ ✓

$T' \to *FT' \mid \varepsilon$ ✓

$F \to \underbrace{(\underline{E})} \mid id$ *

We had:

FIRST(S) = FIRST(E) = FIRST(T) = FIRST(F)

= { (, id }

FIRST(E') = { +, ε }

FIRST(T') = { *, ε)

So:

FOLLOW(S) = { $ } +?

FOLLOW(E) = { ,), *, $ }

FOLLOW(E') = { *, $

FOLLOW(T) = { *, $

FOLLOW(T') = { *, $

FOLLOW(F) = { *, $

BUG!

Try again: $S \to E$

$E \to \underline{TE'}$

$E' \to +TE' \mid \varepsilon$

$T \to FT'$

$T' \to * FT' \mid \varepsilon$

$F \to (E) \mid id$

$FIRST(S) = FIRST(E) = FIRST(T)$

$= FIRST(F) = \{(, id\}$

$FIRST(T') = \{*, \varepsilon\}$

$FIRST(E') = \{+, \varepsilon)$

$FOLLOW(S) = \{\$\}$

$FOLLOW(E) = \{), \$\}$

$FOLLOW(E') = \{), \$\}$

$FOLLOW(T) = \{+,), \$\}$

$FOLLOW(T') = \{+,), \$\}$

$FOLLOW(F) = \{*, +,), \$\}$

Then, the Table: M:

For any production $X \to \alpha$, do

1) for each terminal a in FIRST(α), add

   $X \to \alpha$ to $M[A, a]$

2) If ε is in FIRST(α), add $X \to \alpha$ to $M[A, b]$

   for each terminal b in FOLLOW(A).

   If ε is in FIRST(α) and $\$$ is in FOLLOW(A), add $A \to \alpha$ to $M[A, \$]$.

Any other entries are errors.

(construct on board)

End result:

	Inputs					
Nonterminal	id	+	*	(	)	$
S	$S \to E$			$S \to E$		
E	$E \to TE'$			$E \to TE'$		
E'		$E' \to +TE'$			$E' \to \varepsilon$	$E' \to \varepsilon$
T	$T \to FT'$			$T \to FT'$		
T'		$T' \to \varepsilon$	$T' \to *FT'$		$T' \to \varepsilon$	$T' \to \varepsilon$
F	$F \to id$			$F \to (E)$		

Then: Parsing!

Stack	Input	Action	Matched
S $	id+id*id $		

on board
(check schedule page)

Runtime:

One more example:

A grammer for lists/tuples;

$$S' \longrightarrow S\,\$$$

$$S \longrightarrow (L) \mid id$$

$$L \longrightarrow L, S \mid id$$

Ex: $(a, (b, c))\$$

Derivation:

Problem:

LL version: Same trick as before

$$S' \to S\ \$$$

$$S \to (L) \mid id$$

$$L \to S L'$$

$$L' \to ,\ S L' \mid \varepsilon$$

FIRST & FOLLOW Sets:

Table	FIRST	FOLLOW
S'	(, id	\$
S	(, id	{, \$ &)}
L	(, id	)
L'	, , ε	)

↑ comma

$L \to S$ FOLLOW(S) ⊆ FOLLOW(L)

$L' \to S$ ∪ FOLLOW(L

(Note: ε can't be in follow sets!)

Recall: To generate table:

1. For each terminal in FIRST(A), add $A \to \alpha$ to $M[A, a]$

2. If $\varepsilon \in$ FIRST(A), then for each b in FOLLOW(A), add $A \to \alpha$ in $M[A, b]$

In ours, ε in FIRST(L')
Only thing in FOLLOW(L') is)

3. Any blanks become errors.

Table is key! Tells it how to parse.

Our table:

Nonterminals	(	)	id	,	$
S'	$S' \to S$		$S' \to S$		
S	$S' \to (L)$		$S \to id$		
L	$L \to SL'$		$L \to SL'$		
L'		$L' \to \varepsilon$		$L' \to ,SL'$	

State (including non-terms)

which rule we apply, based on table

Matched	Stack	Input	Action
	$S'\$$	$(a,(b,c))\$$	use $S' \to S$
	$S\$$		

Remember:

This whole approach is just to "automate" parsing.

LL is a simple yet powerful + fast class.