Section: LL Parsing

LL(k) Parser:

- top-down parser - starts with start symbol on stack, and repeatedly replace nonterminals until string is generated.
- predictive parser - predict next rewrite rule
- first L of LL means - read input string left to right
- second L of LL means - produces leftmost derivation
- k - number of lookahead symbols

LL parsing process:

- **convert CFG to PDA (different method than before)**
- **Use the PDA and lookahead symbols**
- **Lookahead symbol is next symbol in input string**

Convert CFG to NPDA

The constructed NPDA:

- **Three states: s, q, f**

  start in state s

  push S on stack, move into q

  all rewrite rules in state q: If lhs of rewrite rule on top of stack, replace it with rhs of rewrite rule and stay in state q

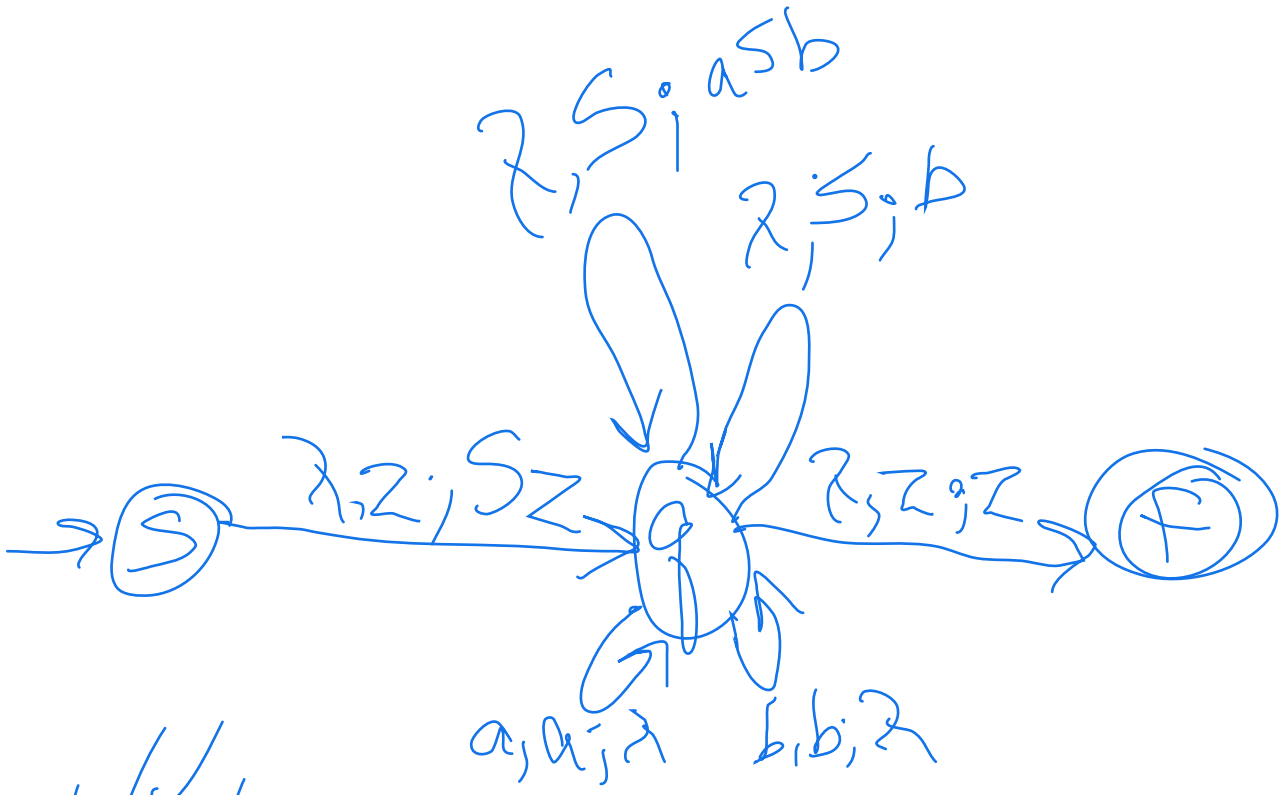
  additional rules in q to recognize terminals: read input symbol, pop input symbol, stay in state q

  pop z from stack, move into f, accept

Example:

$L = \{a^n b b^n : n \geq 0\}$

$S \to aSb \mid b$



trace

aabbb$

a

S Sb

S b b b b

S b b b b b b

z z z z z z z z z

aabb ∈ L

```
state = s
push(S)
state = q
read(symbol)
while top-of-stack ≠ z do
    case top-of-stack of
        S: if symbol == a then
               {pop(); push(aSb)}
           else if symbol == b then
               {pop(); push(b)}
           else error
        a: if symbol ≠ a, then error
               else {pop(); read(symbol)}
        b: if symbol ≠ b, then error
               else {pop(); read(symbol)}
        end case
end while
pop()
if symbol ≠ $ then error
state = f
```

LL Parse Table - 2-dim array

- **rows - variables**
- **cols - terminals, $ (end of string marker)**
- **LL[i,j]**

contains rhs of a rule

Example: Parse table for

$L = \{a^n b b^n : n \geq 0\}$

S → aSb | b

	a	b	$
S	aSb	b	

blank means error

A generic parsing routine

```
push(S)
read(symbol)
while stack not empty do
 case top-of-stack of
  terminal:
   if top-of-stack == symbol
    then {pop(); read(symbol)}
   else
    error
  variable:
   if LL[top-of-stack, symbol] ≠ error
    then {pop()
     push(LL[top-of-stack,symbol])}
   else
    error
 end case
end while
if symbol ≠ $, then error
```

Example:

$$\mathbf{S} \rightarrow \mathbf{aSb}$$
$$\mathbf{S} \rightarrow \mathbf{c}$$

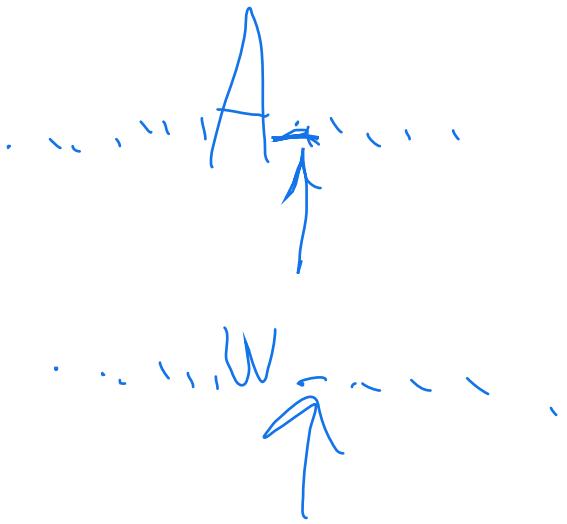
	a	**b**	**c**	**$**
S	**aSb**	**error**	**c**	**error**

Example:

$$\mathbf{S} \rightarrow \mathbf{Ac} \mid \mathbf{Bc}$$
$$\mathbf{A} \rightarrow \mathbf{aAb} \mid \lambda$$
$$\mathbf{B} \rightarrow \mathbf{b}$$

To construct an LL parse table LL[rows,cols]:

1. **For each rule A $\rightarrow$ w**
   (a) **For each a in FIRST(w)**
   add w to LL[A,a]
   (b) **If λ is in FIRST(w)**
   add w to LL[A,b] for each b in FOLLOW(A)
2. **Each undefined entry is error.**



Example:

$$\mathbf{S} \rightarrow \mathbf{aSc} \mid \mathbf{B}$$
$$\mathbf{B} \rightarrow \mathbf{b} \mid \lambda$$

	FIRST	**FOLLOW**
S	**a,b,λ**	**\$,c**
B	**b,λ**	**\$,c**

To Compute the LL Parse Table for this example:

- **For S $\rightarrow$ aSc,**
  FIRST(aSc) = {a}
- **For S $\rightarrow$ B,**
  FIRST(B) = {b, λ}
  FOLLOW(S) = {\$, c}
- **For B $\rightarrow$ b,**
  FIRST(b) = {b}
- **For B $\rightarrow \lambda$**
  FIRST(λ) = {λ}

LL(1) Parse Table:

	a	b	c	$
S	aSc	B	B	B
B		b	λ	λ

Parse string: aacc

Trace aabcc

			a					
	a		S	S	B	b		
	S	S	c	c	c	c	c	
Stack:	S	c	c	c	c	c	c	c	c
symbol:	a	a	a'	a'	b	b	b	c	c'

lookahead

second a

aabcc ∈ L

Example: Construct Parse Table for:

$$\mathbf{S} \rightarrow \mathbf{AcB}$$
$$\mathbf{A} \rightarrow \mathbf{aAb}$$
$$\mathbf{A} \rightarrow \lambda$$
$$\mathbf{B} \rightarrow \mathbf{aBb}$$
$$\mathbf{B} \rightarrow \mathbf{c}$$

FIRST(A) = {a, λ}

FIRST(S) = {a, c}

FIRST(B) = {a, c}

FOLLOW(A) = {b, c}

FOLLOW(S) = {$}

FOLLOW(B) = {b, $}

S$ → AcB$

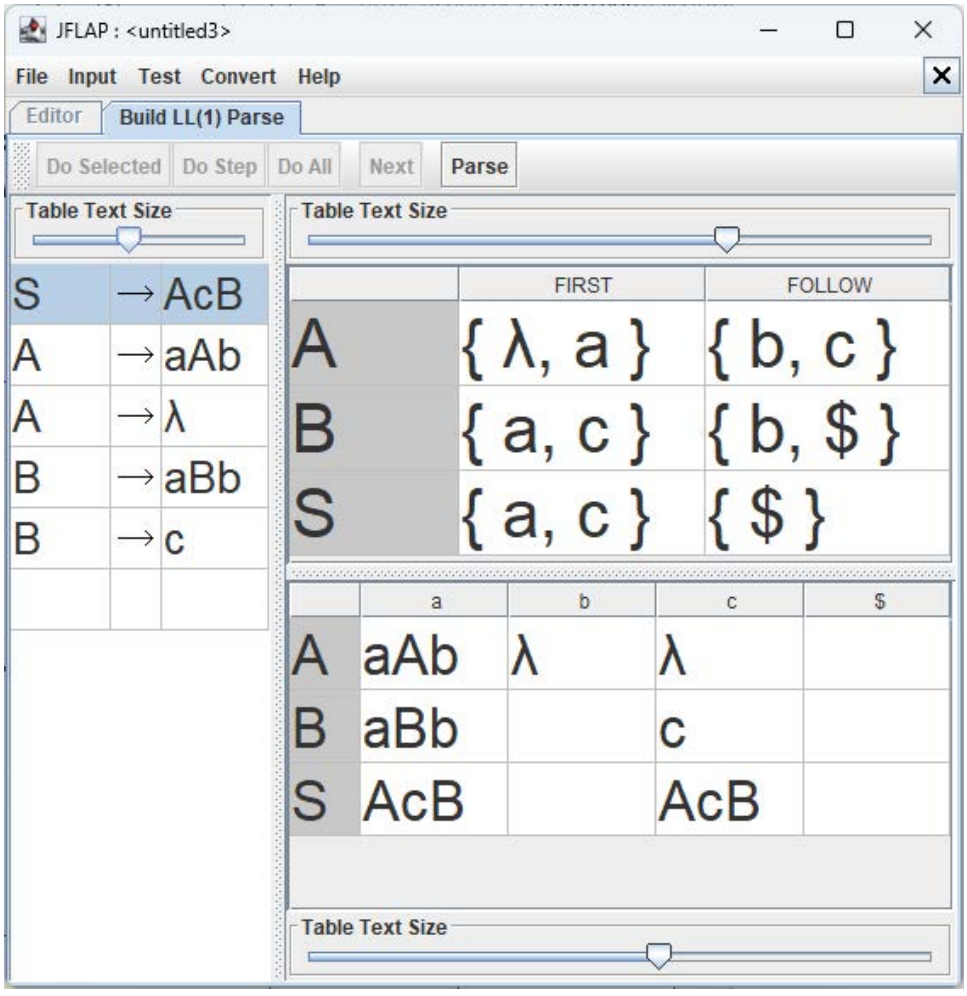
To compute the parse table:

- **For S $\rightarrow$ AcB,**
  FIRST(AcB) = $\{a, c\}$
- **For A $\rightarrow$ aAb,**
  FIRST(aAb) = $\{a\}$
- **For A $\rightarrow \lambda$,**
  FIRST(λ) = $\{\lambda\}$
- **For B $\rightarrow$ aBb,**
  FIRST(aBb) = $\{a\}$
- **For B $\rightarrow$ c,**
  FIRST(c) = $\{c\}$
- **All other entries are errors.**

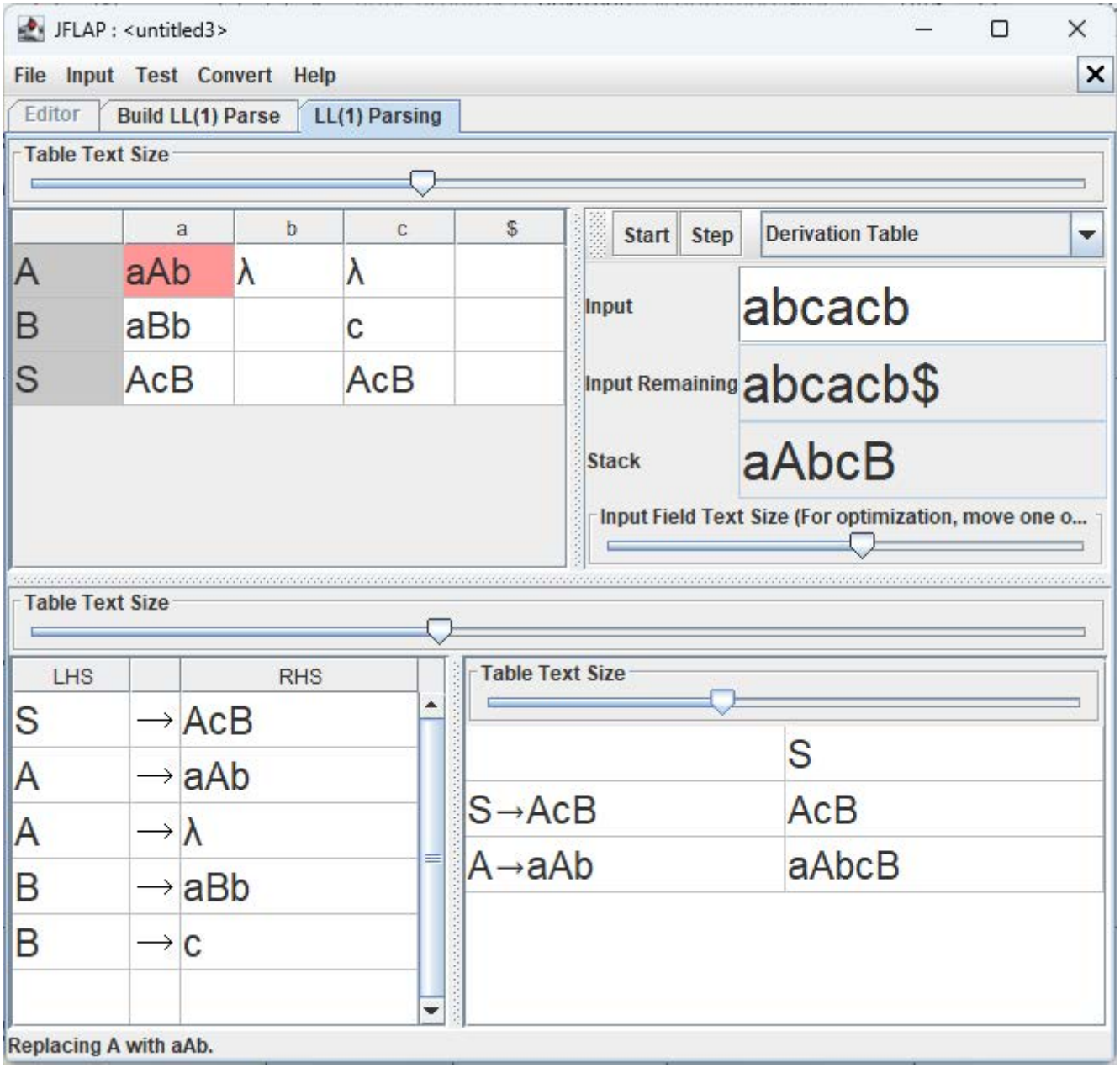
LL(1) Parse Table:

	a	b	c	$
S	AcB		AcB	
A	aAb	λ	λ	
B	aBb		c	

In JFLAP, start with grammar, Then "Input", then build LL(1) Parse Table



Use table to parse string abcacb



Example:

$$\mathbf{S \rightarrow AcB}$$
$$\mathbf{A \rightarrow aAb \mid ab}$$
$$\mathbf{B \rightarrow aBb \mid acb}$$

	FIRST	**FOLLOW**
S	a	
A	a	
B	a	

	a	**b**	**c**	**$**
S	AcB			
A	ab aAb			
B	acb aBb			

not good

one lookahead doesn't work

not LL(1)

LL(2) Parse Table:

use 2 lookaheads

	aa	ab	ac	a$	b	c	$
S	AcB	AcB	error	error	error	error	error
A	aAb	ab	error	error	error	error	error
B	aBb	error	acb	error	error	error	error

this is LL(2) grammar

parse string: aabbcacb

			a		a					
			A	A	b	b				
		A	b	b	b	b	b		a	
		c	c	c	c	c	c	c	c	c
Stack:	S	B	B	B	B	B	B	B	B	b	b
symbol:	aa	aa	aa	ab	ab	bb	bc	ca	ac	ac	cb

↑ 2 lookaheads

Example:

$L = \{a^n : n \geq 0\} \cup \{a^n b^n : n \geq 0\}$

$$\mathbf{S} \rightarrow \mathbf{A}$$
$$\mathbf{S} \rightarrow \mathbf{B}$$
$$\mathbf{A} \rightarrow \mathbf{aA}$$
$$\mathbf{A} \rightarrow \lambda$$
$$\mathbf{B} \rightarrow \mathbf{aBb}$$
$$\mathbf{B} \rightarrow \lambda$$

This is LL(k) what is k?

not LL(k) for any k

Example:

$L = \{a^n : 0 \leq n \leq 10\} \cup \{a^n b^n : 0 \leq n \leq 10\}$

Example

$$\mathbf{S} \rightarrow \mathbf{bbCd} \mid \mathbf{Bcc}$$
$$\mathbf{B} \rightarrow \mathbf{bB} \mid \mathbf{b}$$
$$\mathbf{C} \rightarrow \mathbf{cC} \mid \mathbf{c}$$