EECS483 D4:
Top-down & Bottom-up Parsing

Chun-Hung Hsiao

Feb 1, 2013

LL(1) Parsing: A Complete Example

- Grammar
  $E \rightarrow T + E\ x \mid F$
  $T \rightarrow T * F\ y \mid w$
  $F \rightarrow E \mid z \mid \varepsilon$
- Is it an LL(1) grammar?

Constructing LL(1) Parsing Table

- Eliminate ε-productions
- Eliminate cycles
- Remove left recursions
- Left factoring
- Compute FIRST and FOLLOW
- Construct the parsing table

Eliminating ε-productions

- Nullable nonterminals
  - E, F

$$E \rightarrow T + E\ x \mid F$$
$$T \rightarrow T * F\ y \mid w$$
$$F \rightarrow E \mid z \mid \varepsilon$$

Eliminating ε-productions

- Nullable nonterminals
  - E, F
- Remove all direct ε-productions

$E \rightarrow T + E\ x \mid F$
$T \rightarrow T * F\ y \mid w$
$F \rightarrow E \mid z$

Eliminating ε-productions

- Nullable nonterminals
  - E, F
- Remove all direct ε-productions
- For each nullable nonterminal appearing in RHS, generate a corresponding rule without it
  - Discard any generated ε-production

$E \to T + E\ x \mid F \mid$
$\quad T + x$
$T \to T * F\ y \mid w$
$F \to E \mid z$

Eliminating ε-productions

- Nullable nonterminals
  - E, F
- Remove all direct ε-productions
- For each nullable nonterminal appearing in RHS, generate a corresponding rule without it
  - Discard any generated ε-production

$E \rightarrow T + E\ x \mid F \mid$
$\quad T + x$
$T \rightarrow T * F\ y \mid w \mid$
$\quad T * y$
$F \rightarrow E \mid z$

Eliminating ε-productions

- Nullable nonterminals
  - E, F
- Remove all direct ε-productions
- For each nullable nonterminal appearing in RHS, generate a corresponding rule without it
  - Discard any generated ε-production
- Add a new starting nonterminal if the original one is nullable

$S \rightarrow E \mid \varepsilon$

$E \rightarrow T + E\ x \mid F \mid$
$\quad\quad T + x$

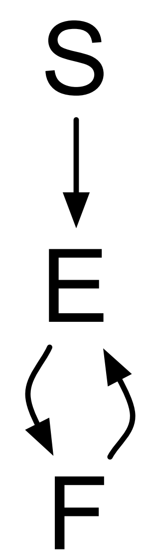
$T \rightarrow T * F\ y \mid w \mid$
$\quad\quad T * y$

$F \rightarrow E \mid z$

Eliminating Cycles

- Cyclic nonterminals
  - E, F

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E\ x \mid F \mid$
$\quad T + x$
$T \rightarrow T * F\ y \mid w \mid$
$\quad T * y$
$F \rightarrow E \mid z$



Eliminating Cycles

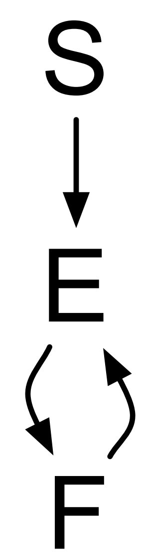
- Cyclic nonterminals
  - E, F
- Find out all productions that are acyclic for each cyclic symbol

$S \rightarrow E \mid \varepsilon$

$E \rightarrow T + E\ x \mid F \mid T + x$

$T \rightarrow T * F\ y \mid w \mid T * y$

$F \rightarrow E \mid z$



Eliminating Cycles

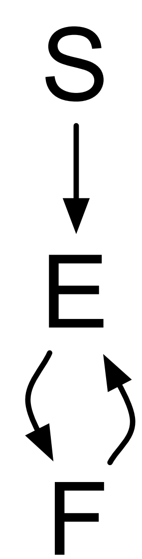
- Cyclic nonterminals
  - E, F
- Find the acyclic strings produced by these cyclic nonterminals

$S \rightarrow E \mid \varepsilon$

$E \rightarrow T + E\ x \mid F \mid T + x$

$T \rightarrow T * F\ y \mid w \mid T * y$

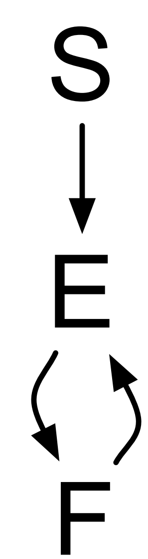
$F \rightarrow E \mid z$



Eliminating Cycles

- Cyclic nonterminals
  - E, F
- Find the acyclic strings produced by these cyclic nonterminals
- Replace the cyclic RHS of each single production with the acyclic strings

S → E | ε
E → T + E x | z |
    T + x
T → T * F y | w |
    T * y
F → T + E x |
    T + x | z



Removing Left Recursions

- Remove left recursions for each nonterminal in an arbitrary order
  - S, E, T, F

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E\ x \mid z \mid$
$\quad T + x$
$T \rightarrow T * F\ y \mid w \mid$
$\quad T * y$
$F \rightarrow T + E\ x \mid$
$\quad T + x \mid z$

Removing Left Recursions

- Rewrite the production rules in order such that they cannot being with previous nonterminals
  - If we have:
    $A_i \rightarrow \delta_1 \mid \delta_2 \mid \delta_3$
    $A_j \rightarrow A_i\gamma$ (j > i)
  - Rewrite the later to:
    $A_j \rightarrow \delta_1\gamma \mid \delta_2\gamma \mid \delta_3\gamma$
- Remove direct left recursions
  - If we have:
    $A \rightarrow A\alpha_1 \mid A\alpha_2 \mid \beta_1 \mid \beta_2$
  - Rewrite it to:
    $A \rightarrow \beta_1 A' \mid \beta_2 A'$
    $A' \rightarrow \alpha_1 A' \mid \alpha_2 A' \mid \varepsilon$

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E\ x \mid z \mid T + x$
$T \rightarrow T * F\ y \mid w \mid T * y$
$F \rightarrow T + E\ x \mid T + x \mid z$

Removing Left Recursions

- Rewrite the production rules in order such that they cannot being with previous nonterminals
  - If we have:
    $A_i \rightarrow \delta_1 \mid \delta_2 \mid \delta_3$
    $A_j \rightarrow A_i\gamma$ (j > i)
  - Rewrite the later to:
    $A_j \rightarrow \delta_1\gamma \mid \delta_2\gamma \mid \delta_3\gamma$
- Remove direct left recursions
  - If we have:
    $A \rightarrow A\alpha_1 \mid A\alpha_2 \mid \beta_1 \mid \beta_2$
  - Rewrite it to:
    $A \rightarrow \beta_1 A' \mid \beta_2 A'$
    $A' \rightarrow \alpha_1 A' \mid \alpha_2 A' \mid \varepsilon$

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E\ x \mid z \mid$
$\quad T + x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ F\ y\ T' \mid$
$\quad *\ y\ T' \mid \varepsilon$
$F \rightarrow T + E\ x \mid$
$\quad T + x \mid z$

Removing Left Recursions

- Rewrite the production rules in order such that they cannot being with previous nonterminals
  - If we have:

    $A_i \rightarrow \delta_1 \mid \delta_2 \mid \delta_3$

    $A_j \rightarrow A_i\gamma \qquad (j > i)$
  - Rewrite the later to:

    $A_j \rightarrow \delta_1\gamma \mid \delta_2\gamma \mid \delta_3\gamma$
- Remove direct left recursions
  - If we have:

    $A \rightarrow A\alpha_1 \mid A\alpha_2 \mid \beta_1 \mid \beta_2$
  - Rewrite it to:

    $A \rightarrow \beta_1A' \mid \beta_2A'$

    $A' \rightarrow \alpha_1A' \mid \alpha_2A' \mid \varepsilon$

$S \rightarrow E \mid \varepsilon$

$E \rightarrow T + E\ x \mid z \mid T + x$

$T \rightarrow w\ T'$

$T' \rightarrow *\ F\ y\ T' \mid *\ y\ T' \mid \varepsilon$

$F \rightarrow T + E\ x \mid T + x \mid z$

Removing Left Recursions

- Rewrite the production rules in order such that they cannot being with previous nonterminals
  - If we have:
    $A_i \rightarrow \delta_1 \mid \delta_2 \mid \delta_3$
    $A_j \rightarrow A_i\gamma$ (j > i)
  - Rewrite the later to:
    $A_j \rightarrow \delta_1\gamma \mid \delta_2\gamma \mid \delta_3\gamma$
- Remove direct left recursions
  - If we have:
    $A \rightarrow A\alpha_1 \mid A\alpha_2 \mid \beta_1 \mid \beta_2$
  - Rewrite it to:
    $A \rightarrow \beta_1 A' \mid \beta_2 A'$
    $A' \rightarrow \alpha_1 A' \mid \alpha_2 A' \mid \varepsilon$

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E\ x \mid z \mid T + x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ F\ y\ T' \mid *\ y\ T' \mid \varepsilon$
$F \rightarrow w\ T' + E\ x \mid w\ T' + x \mid z$

Removing Left Recursions

- Remark: Removing left recursions changes the associativity!
  - No known automatic left recursion removal algorithm to preserve left associativity

S → E | ε
E → T + E x | z | T + x
T → w T'
T'→ * F y T' | * y T' | ε
F → w T' + E x | w T' + x | z

Left Factoring

- Rewrite the production rules of each nonterminal to remove common prefixes
  - If we have:
    $A \to \alpha A_1 \mid \alpha A_2 \mid \beta$
  - Rewrite it to:
    $A \to \alpha A' \mid \beta$
    $A' \to A_1 \mid A_2$

$S \to E \mid \varepsilon$
$E \to T + E\ x \mid z \mid$
$\quad T + x$
$T \to w\ T'$
$T' \to *\ F\ y\ T' \mid$
$\quad *\ y\ T' \mid \varepsilon$
$F \to w\ T' + E\ x \mid$
$\quad w\ T' + x \mid z$

Left Factoring

- Rewrite the production rules of each nonterminal to remove common prefixes
  - If we have:
    $A \rightarrow \alpha A_1 \mid \alpha A_2 \mid \beta$
  - Rewrite it to:
    $A \rightarrow \alpha A' \mid \beta$
    $A' \rightarrow A_1 \mid A_2$

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ F\ y\ T' \mid *\ y\ T' \mid \varepsilon$
$F \rightarrow w\ T' + E\ x \mid w\ T' + x \mid z$

Left Factoring

- Rewrite the production rules of each nonterminal to remove common prefixes
  - If we have:
    $A \rightarrow \alpha A_1 \mid \alpha A_2 \mid \beta$
  - Rewrite it to:
    $A \rightarrow \alpha A' \mid \beta$
    $A' \rightarrow A_1 \mid A_2$

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ F\ y\ T' \mid$
$\quad *\ y\ T' \mid \varepsilon$
$F \rightarrow w\ T' + E\ x \mid$
$\quad w\ T' + x \mid z$

Left Factoring

- Rewrite the production rules of each nonterminal to remove common prefixes
  - If we have:
    $A \rightarrow \alpha A_1 \mid \alpha A_2 \mid \beta$
  - Rewrite it to:
    $A \rightarrow \alpha A' \mid \beta$
    $A' \rightarrow A_1 \mid A_2$

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + E\ x \mid w\ T' + x \mid z$

Left Factoring

- Rewrite the production rules of each nonterminal to remove common prefixes
  - If we have:
    $A \rightarrow \alpha A_1 \mid \alpha A_2 \mid \beta$
  - Rewrite it to:
    $A \rightarrow \alpha A' \mid \beta$
    $A' \rightarrow A_1 \mid A_2$

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + E\ x \mid$
$w\ T' + x \mid z$

Left Factoring

- Rewrite the production rules of each nonterminal to remove common prefixes
  - If we have:
    $A \rightarrow \alpha A_1 \mid \alpha A_2 \mid \beta$
  - Rewrite it to:
    $A \rightarrow \alpha A' \mid \beta$
    $A' \rightarrow A_1 \mid A_2$

$S \rightarrow E \mid \varepsilon$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

Computing FIRST and FOLLOW

- $A \rightarrow X_1X_2...X_n$
  - If X_1 is a terminal, the add X_1 into FIRST(A)
  - If $A \rightarrow \varepsilon$, then add ε into FIRST(A)

	FIRST	FOLLOW
S		
E		
E_1		
T		
T'		
T'_1		
F		
F_1		

$S \rightarrow E\$ \mid \$$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

Computing FIRST and FOLLOW

- $A \rightarrow X_1X_2...X_n$
  - If X_1 is a terminal, the add X_1 into FIRST(A)
  - If $A \rightarrow \varepsilon$, then add ε into FIRST(A)

	FIRST	FOLLOW
S	{$}	
E	{z}	
E_1	{x}	
T	{w}	
T'	{*, ε}	
T'_1	{y}	
F	{w, z}	
F_1	{x}	

$S \rightarrow E\$ \mid \$$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

Computing FIRST and FOLLOW

- $A \rightarrow X_1X_2 \ldots X_n$
  - Add $FIRST(X_1)$ - {ε} into $FIRST(A)$
  - If $\varepsilon \in FIRST(X_1) \cap \ldots \cap FIRST(X_{i-1})$, then add $FIRST(X_i)$ into $FIRST(A)$
  - If $\varepsilon \in FIRST(X_1) \cap \ldots \cap FIRST(X_n)$, then add ε into $FIRST(A)$

	FIRST	FOLLOW
S	{$}	
E	{z}	
E_1	{x}	
T	{w}	
T'	{*, ε}	
T'_1	{y}	
F	{w, z}	
F_1	{x}	

$S \rightarrow E\$ \mid \$$

$E \rightarrow T + E_1 \mid z$

$E_1 \rightarrow E\ x \mid x$

$T \rightarrow w\ T'$

$T' \rightarrow *\ T'_1 \mid \varepsilon$

$T'_1 \rightarrow F\ y\ T' \mid y\ T'$

$F \rightarrow w\ T' + F_1 \mid z$

$F_1 \rightarrow E\ x \mid x$

Computing FIRST and FOLLOW

- $A \rightarrow X_1X_2\ldots X_n$
  - Add FIRST(X_1) - {ε} into FIRST(A)
  - If $\varepsilon \in$ FIRST(X_1)∩...∩FIRST(X_{i-1}), then add FIRST(X_i) into FIRST(A)
  - If $\varepsilon \in$ FIRST(X_1)∩...∩FIRST(X_n), then add ε into FIRST(A)

	FIRST	FOLLOW
S	{$, z, w}	
E	{z, w}	
E_1	{x, z, w}	
T	{w}	
T'	{*, ε}	
T'_1	{y, w, z}	
F	{w, z}	
F_1	{x, z, w}	

$S \rightarrow E\$ \mid \$$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

Computing FIRST and FOLLOW

- If A is a starting nonterminal, then add $ into FOLLOW(A)
- Otherwise, FOLLOW(A) = ∅

$S \rightarrow E\$ \mid \$$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

	FIRST	FOLLOW
S	{$, z, w}	{$}
E	{z, w}	{}
E_1	{x, z, w}	{}
T	{w}	{}
T'	{*, ε}	{}
T'_1	{y, w, z}	{}
F	{w, z}	{}
F_1	{x, z, w}	{}

Computing FIRST and FOLLOW

- $A \rightarrow \alpha B \beta$
  - Add FIRST(β) - {ε} into FOLLOW(B)
  - If $\varepsilon \in$ FIRST(β), then add FOLLOW(A) into FOLLOW(B)

	FIRST	FOLLOW
S	{$, z, w}	{$}
E	{z, w}	{}
E_1	{x, z, w}	{}
T	{w}	{}
T'	{*, ε}	{}
T'_1	{y, w, z}	{}
F	{w, z}	{}
F_1	{x, z, w}	{}

$S \rightarrow E\$ \mid \$$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

Computing FIRST and FOLLOW

- $A \rightarrow \alpha B \beta$
  - Add FIRST(β) - {ε} into FOLLOW(B)
  - If $\varepsilon \in$ FIRST(β), then add FOLLOW(A) into FOLLOW(B)

$S \rightarrow E\$ \mid \$$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

	FIRST	FOLLOW
S	{$, z, w}	{$}
E	{z, w}	{$, x}
E_1	{x, z, w}	{$, x}
T	{w}	{+}
T'	{*, ε}	{+}
T'_1	{y, w, z}	{+}
F	{w, z}	{y}
F_1	{x, z, w}	{y}

Constructing the Parsing Table

$S \rightarrow E\$ \mid \$$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

	FIRST	FOLLOW
S	{$, z, w}	{$}
E	{z, w}	{$, x}
E_1	{x, z, w}	{$, x}
T	{w}	{+}
T'	{*, ε}	{+}
T'_1	{y, w, z}	{+}
F	{w, z}	{y}
F_1	{x, z, w}	{y}

	w	x	y	z	+	*	$
S							
E							
E_1							
T							
T'							
T'_1							
F							
F_1							

Constructing the Parsing Table

$S \rightarrow E\$ \mid \$$
$E \rightarrow T + E_1 \mid z$
$E_1 \rightarrow E\ x \mid x$
$T \rightarrow w\ T'$
$T' \rightarrow *\ T'_1 \mid \varepsilon$
$T'_1 \rightarrow F\ y\ T' \mid y\ T'$
$F \rightarrow w\ T' + F_1 \mid z$
$F_1 \rightarrow E\ x \mid x$

	FIRST	FOLLOW
S	{$, z, w}	{$}
E	{z, w}	{$, x}
E_1	{x, z, w}	{$, x}
T	{w}	{+}
T'	{*, ε}	{+}
T'_1	{y, w, z}	{+}
F	{w, z}	{y}
F_1	{x, z, w}	{y}

	w	x	y	z	+	*	$
S	$S \rightarrow E\$$			$S \rightarrow E\$$			$S \rightarrow \$$
E	$E \rightarrow T + E_1$			$E \rightarrow z$			
E_1	$E_1 \rightarrow E\ x$	$E_1 \rightarrow x$		$E_1 \rightarrow E\ x$			
T	$T \rightarrow w\ T'$						
T'					$T' \rightarrow \varepsilon$	$T' \rightarrow *\ T'_1$	
T'_1	$T'_1 \rightarrow F\ y\ T'$		$T'_1 \rightarrow y\ T'$	$T'_1 \rightarrow F\ y\ T'$			
F	$F \rightarrow wT'+F_1$			$F \rightarrow z$			
F_1	$F_1 \rightarrow E\ x$	$F_1 \rightarrow x$		$F_1 \rightarrow E\ x$			

Parsing Example

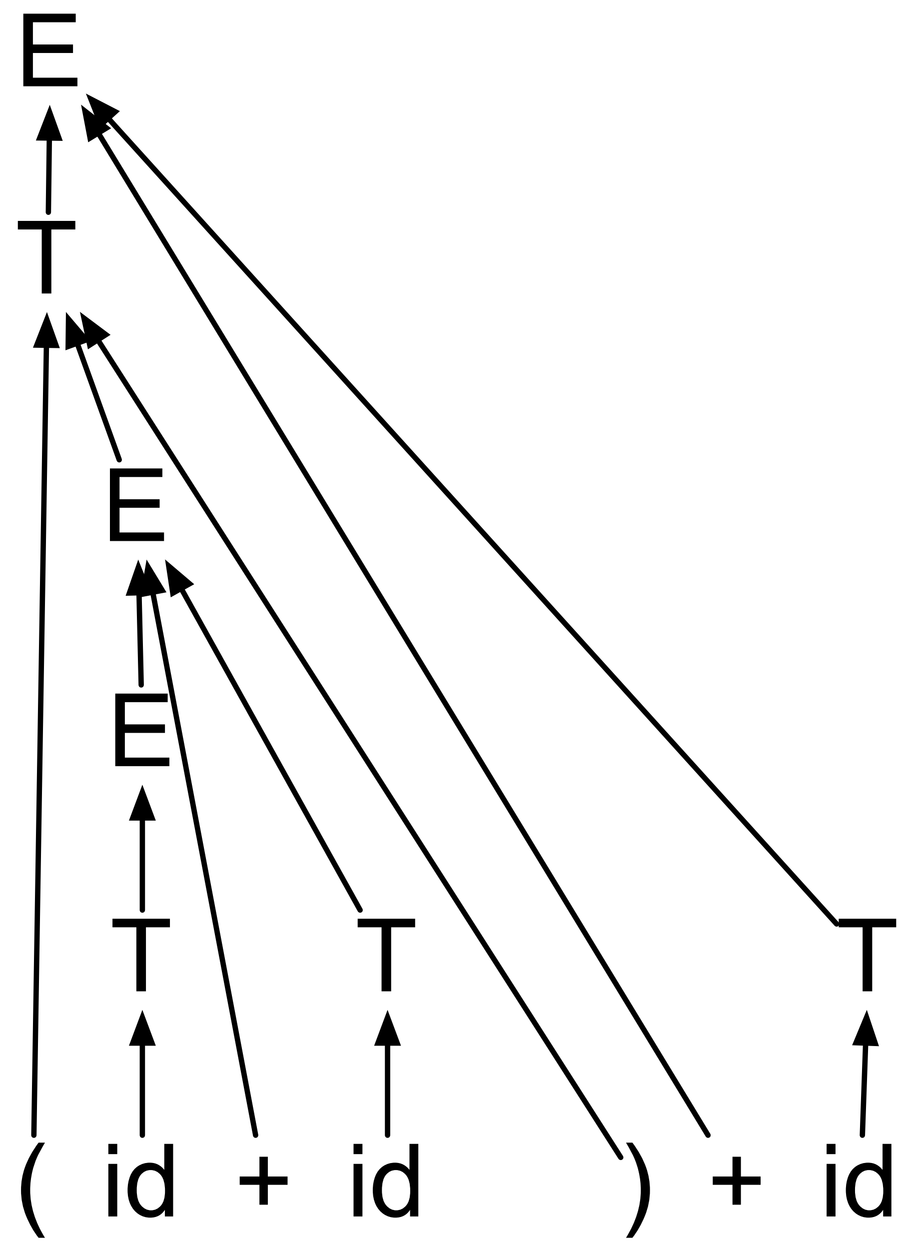
- w*zy+w*y+xx

	w	x	y	z	+	*	$
S	$S \rightarrow E\$$			$S \rightarrow E\$$			$S \rightarrow \$$
E	$E \rightarrow T + E_1$			$E \rightarrow z$			
E_1	$E_1 \rightarrow E\ x$	$E_1 \rightarrow x$		$E_1 \rightarrow E\ x$			
T	$T \rightarrow w\ T'$						
T'					$T' \rightarrow \varepsilon$	$T' \rightarrow *\ T'_1$	
T'_1	$T'_1 \rightarrow F\ y\ T'$		$T'_1 \rightarrow y\ T'$	$T'_1 \rightarrow F\ y\ T'$			
F	$F \rightarrow wT'+F_1$			$F \rightarrow z$			
F_1	$F_1 \rightarrow E\ x$	$F_1 \rightarrow x$		$F_1 \rightarrow E\ x$			

Bottom-up Parsing

- Grammar:
  E → E + T | T
  T → (E) | id
- Parsing "(id + id) + id"
  - ↑ (id + id) + id : expecting E
    - Could be E + T or T
    - If T is going to appear, then we also expect (E) or id
  - (↑ id + id) + id : found (
    - Should be (E) so expecting the coming E
    - Again this E may be E + T or T
  - (id + id) ↑ + id
    - Complete a (E), meaning that it is a T

Flow of Bottom-up Parsing



Hint to Project 2

- Use ε-productions carefully
  - A highly probably source of conflicts!
- Implement IfStmt later
  - Dangling Else ambiguity
  - Try to resolve it with operator precedence
- Read y.outyput to help you debug

y.output

- State 174 conflicts: 1 shift/reduce
- state 174
  51 ConditionalStmt: T_If '(' Expr ')' Stmt .
  52 | T_If '(' Expr ')' Stmt . T_Else Stmt

  T_Else shift, and go to state 181

  T_Else [reduce using rule 51 (ConditionalStmt)]
  $default reduce using rule 51 (ConditionalStmt)
- state 181
  52 ConditionalStmt: T_If '(' Expr ')' Stmt T_Else . Stmt

Thanks and all the best!