Unit 4:
First and Follow Sets - Part Two

SCC 312 Compilation

John Vidler
j.vidler@lancaster.ac.uk

Original slide deck by John Mariani

Construction of FIRST and FOLLOW sets

- How do we construct FIRST and FOLLOW sets?

- A Reminder:
  - We have seen the problem caused by the presence of null-productions.
  - If X could generate the null string, FIRST(XY) is not just FIRST(X), it is FIRST(X) & FIRST(Y)
  - This is because FIRST(X) could be { } (the empty set), and therefore we have to consider what is the FIRST of Y.

Construction of first sets

Construction of FIRST sets

- To construct FIRST(α) where
  $\alpha \rightarrow$ **X1 X2 … Xn**
  (assuming simple BNF format):
- if X1 is a terminal, add it to FIRST(α)
- if X1 is a non-terminal with grammar rule **X1** $\rightarrow \beta$, add FIRST(X1) to FIRST(α)
- if X1 can generate the null string, then consider the terminals that can start X2, and so on

Construction of FIRST sets

- For simple BNF format and if there are no null-productions, a possible FIRST algorithm is:

```
for each non-terminal X
    set FIRST(X) to empty ;
do
    {
    for each production X → Y...
        if Y is a terminal
            add Y to FIRST(X) ;
        else // Y is a non-terminal
            add FIRST(Y) to FIRST(X) ;
    }
while there are changes to at least one FIRST set;
```

Construction of FOLLOW sets

- Consider the following.
- night_out → meal drink;
- drink → BEER | WINE | VODKA;
- FOLLOW(meal) = { BEER, WINE, VODKA}
- After we’ve had a meal, we go for a drink.
- What can follow a meal?
- Look for productions with meal on the RHS.
- Look at what follows it.

Construction of FOLLOW sets

- To construct FOLLOW(X) for some non-terminal X:
- put EOF in FOLLOW (distinguished symbol)
- if there is a production
  $\mathbf{Y} \rightarrow \alpha \ \mathbf{X} \ \beta$,
  add FIRST(β) to FOLLOW(X)
- if there is a production $\mathbf{Y} \rightarrow \alpha \ \mathbf{X}$, or
  $\mathbf{Y} \rightarrow \alpha \ \mathbf{X} \ \beta$ with $\beta \Rightarrow \varepsilon$, then add FOLLOW(Y) to FOLLOW(X)

The distinguished symbol
is the "root" of the syntax i.e.
for source text it would probably
be program.

Construction of FOLLOW sets

- For simple BNF format and if there are no null-productions, a possible FOLLOW algorithm is shown on the next slide.

Construction of FOLLOW sets

```
for each non-terminal X
    set FOLLOW(X) to empty ;
insert EOF in FOLLOW (distinguished symbol) ;
do
{
  for each production X → Y1 Y2 Y3 ... Yn
   {
   for each non-terminal Yi with i from 1 to n-1
            if Yi+1 is a terminal
                add Yi+1 to FOLLOW(Yi) ;
            else // Yi+1 is a non-terminal
                add FIRST(Yi+1) to FOLLOW(Yi) ;
   if Yn is a non-terminal
            add FOLLOW(X) to FOLLOW(Yn) ;
   }
} while there are changes to at least one FOLLOW set
;
```

A simpler statement of the FIRST algorithm

- (with no nulls in the grammar)
- FIRST (A) where A $\rightarrow$ X1 X2 … XN
  - if X1 is a terminal, add it to FIRST(A)
  - if X1 is a non-terminal with grammar rule add FIRST(X1) to FIRST(A)
- So we consider each not-terminal in turn.

A simpler statement of the FOLLOW algorithm

- FOLLOW sets
  - initialise FOLLOW sets to { }
  - put EOF in FOLLOW (distinguished symbol)
  - if Y → AXB, add FIRST(B) to FOLLOW(X)
  - if Y → ZXc, add ‘c’ to FOLLOW(X)
  - if Y → AXB, add FOLLOW(Y) to FOLLOW(B)
  - keep going until there are no further changes to the follow sets.

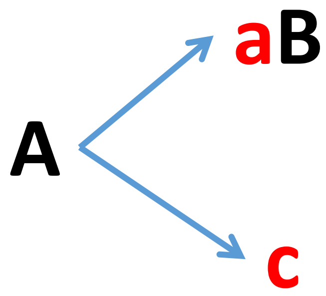
A Worked Example

Example

- (0) S $\rightarrow$ A b B
- (1) S $\rightarrow$ B
- (2) A $\rightarrow$ a B
- (3) A $\rightarrow$ c
- (4) B $\rightarrow$ A
- Terminals : a, b, c
- Non-terminals : S, A, B
- What are the FIRST sets of S, A, B?

FIRST of A

- By looking at rules (2) and (3), we can see that the first of A is {a, c}. Fairly obvious by inspection?



(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

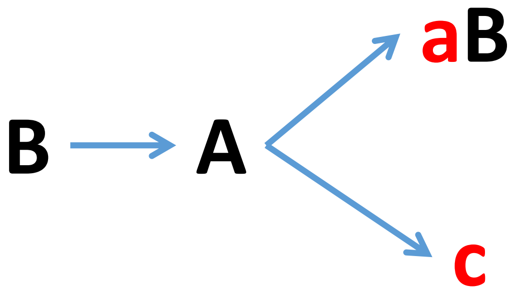
FIRST of A

- Initialise FI(A) = { }
- A appears on the LHS of rules (2) and (3).
- In both rules (2) and (3), the RHS starts with a terminal.
- We add both these terminals to FI(A).
- $FI(A) = FI(A) \cup \{a\} \cup \{c\} = \{\ \} \cup \{a\} \cup \{c\} = \{a, c\}$

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FIRST of B

- By looking at rule (4), we can see that the first of B is a non-terminal, A.
- We need to work out FIRST(A), which we have just done.
- FIRST(B) = FIRST(A) = {a, c}.



(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FIRST of B

- FI(B) = { }
- B appears on the LHS of rule (4).
- The RHS is a non-terminal, A.
- So we need to add FI(A) to FI(B)
- FI(B) = FI(B) $\cup$ FI(A) = { } $\cup$ {a, c} = {a, c}

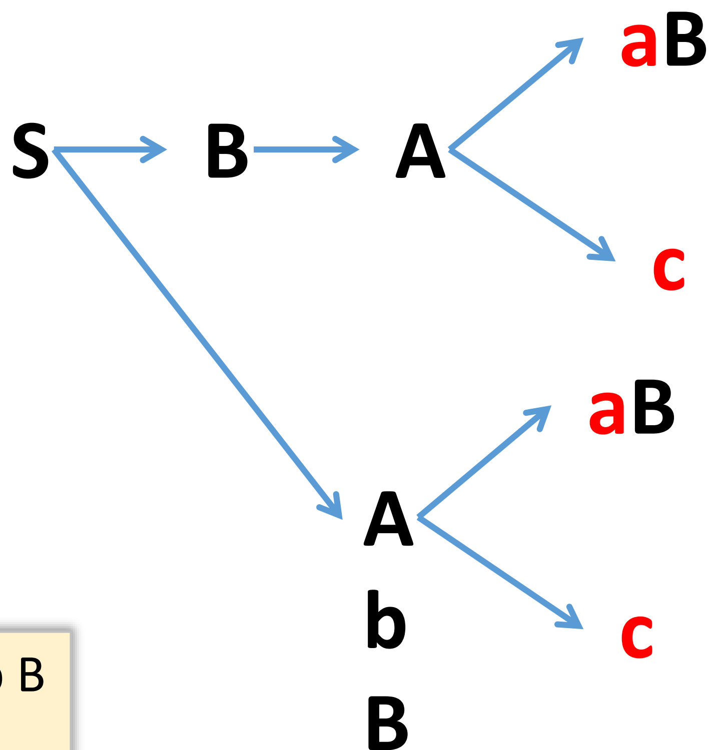
(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FIRST of S

- By looking at rule (0), we can see that the first of S is a non-terminal, A. Similarly, looking at rule (1), the first of S is a non-terminal, B
- We need to work out FIRST(A) and FIRST(B), which we have just done.
- FIRST(S) = FIRST(A) $\cup$ FIRST(B) =
  {a, c} $\cup$ {a, c} = {a, c}.
- Remember, sets do not contain duplicates.

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

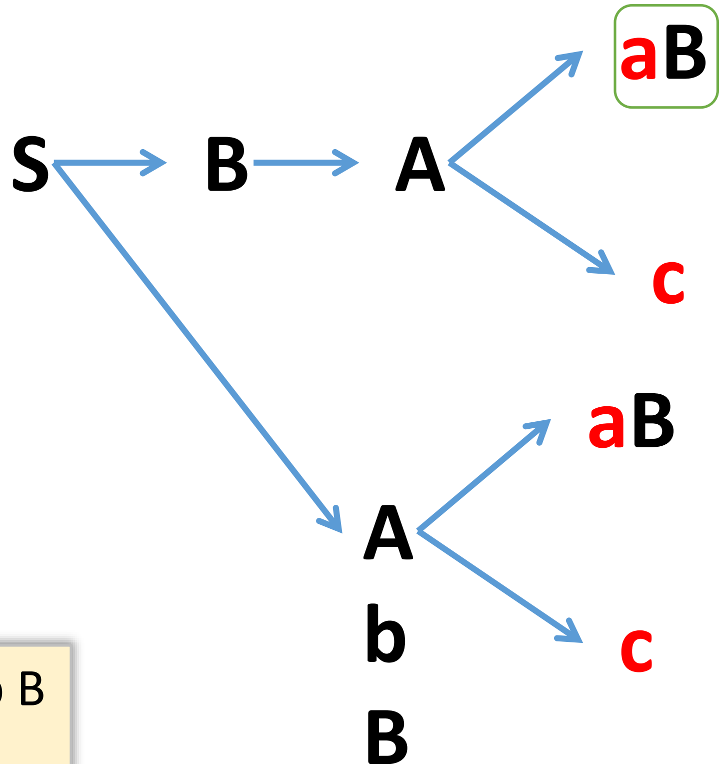
FIRST of S



Is there a problem with this diagram?

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

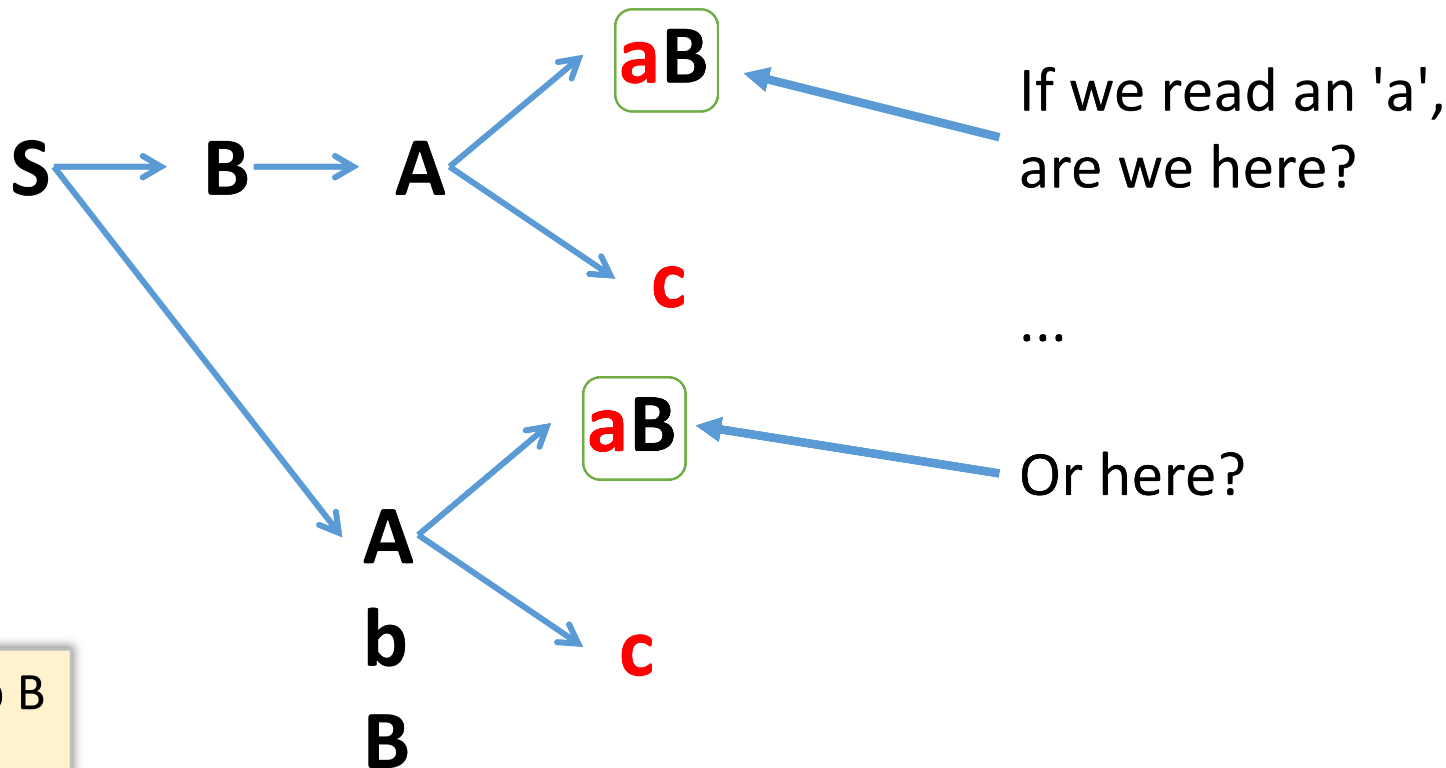
FIRST of S



If we read an 'a', are we here?

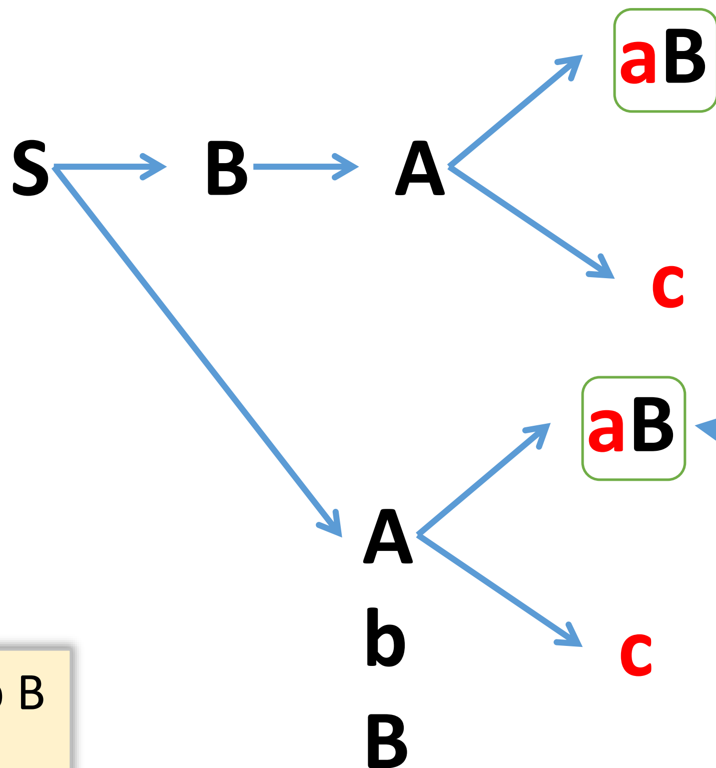
(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FIRST of S



(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FIRST of S



If we read an 'a', are we here?

...

Or here?

Syntactic Ambiguity!

Sometimes a single symbol won't tell us enough to know which rule we're parsing.

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FIRST sets

- FIRST (S) = {a, c}
- FIRST (A) = {a, c}
- FIRST (B) = {a, c}
- What are the FOLLOW sets of S, A, B?

The FOLLOW algorithm again

- To generate the follow sets, we have to look at the RHS of the production rules.
- We treat it as an ordered list of items, where an item is either a terminal or a non-terminal.
- For every non-terminal appearing, we look at what follows it.
- If it is a terminal, we add this to the follow set.
- If it is a non-terminal, we add the FIRST set of that non-terminal to the follow set.

- Finally, we treat the last of the list of items on the RHS as a special case.
- If it is a non-terminal, we add the FOLLOW set of the LHS to the FOLLOW set of that non-terminal.

Initialisation

- Initialise FOLLOW sets to empty.
- FOLLOW (S) = { }
- FOLLOW (A) = { }
- FOLLOW (B) = { }
- put EOF in FOLLOW (distinguished symbol).
  - we use the $ symbol to represent EOF.
  - In this grammar, S is the distinguished symbol.
- FOLLOW (S) = {$}
- A reminder that we are only interested in non-Ts on the RHS. This means we can ignore Rule (3).

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FOLLOW(S) = {$}
FOLLOW(B) = { }
FOLLOW(A) = { }

Rule 0 : S → $\boxed{A}$ b B

- A is a non-T. It is followed by a terminal, b.
- Add b to FOLLOW (A)
- FO(A) = FO(A) ∪ b = { } ∪ {b} = {b}

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FOLLOW(S) = {$}
FOLLOW(B) = { }
FOLLOW(A) = {b}

Rule 0 : S $\rightarrow$ A $\boxed{b}$ B

- b is a terminal.
- We are only concerned about what follows non-Ts.
- So we do nothing.

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FOLLOW(S) = {$}
FOLLOW(B) = { }
FOLLOW(A) = {b}

Rule 0 : S → A b $\boxed{B}$

- B is the last element in the list, and is a non-terminal.
- So we add FO(S) to FO(B).
- FO(B) = FO(B) ∪ FO(S) = { } ∪ {$} = {$}

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FOLLOW(S) = {$}
FOLLOW(B) = { }
FOLLOW(A) = {b}

Rule 1 : S → $\boxed{B}$

- Again, this is the last element in the list, and it is a non-T.
- So we add FO(S) to FO(B).
- (We've just done that! No need to do it again? But even if we did, it would OK).
- FO (B) = FO(B) ∪ FOLLOW(S) = {$} ∪ {$} = {$}

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FOLLOW(S) = {$}
FOLLOW(B) = { }
FOLLOW(A) = {b}

Rule 4 : B → $\boxed{A}$

- One more time, this is the last element in the list, and it is a non-T.
- So we add FO(B) to FO(A)
- FO(A) = FO(A) ∪ FO(B) = {b} ∪ {$} = {b, $}

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FOLLOW(S) = {$}
FOLLOW(B) = { }
FOLLOW(A) = {b}

Rule 2 : A → $\boxed{a}$ B

- This is a terminal, so we do nothing.

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FOLLOW(S) = {$}
FOLLOW(B) = { }
FOLLOW(A) = {b}

Rule 2 : $A \rightarrow a \boxed{B}$

- Yet again, this is the last element in the list, and it is a non-T.
- So we add FO(A) to FO(B)
- FO(B) = FO(B) ∪ FO(A) = {\$} ∪ {b, \$} = {b, \$}

(0) S -> A b B
(1) S -> B
(2) A -> a B
(3) A -> c
(4) B -> A

FOLLOW(S) = {\$}
FOLLOW(B) = { }
FOLLOW(A) = {b}

FOLLOW sets

- FOLLOW (S) = { \$}
- FOLLOW (A) = {b, \$}
- FOLLOW (B) = {b, \$}

Exercise : generate the FIRST and FOLLOW sets

(0) $S \rightarrow L$
(1) $E \rightarrow id$
(2) $E \rightarrow (L)$
(3) $L \rightarrow E$
(4) $L \rightarrow L + E$

FIRST sets

L = { id, (}
S = { id, (}
E = { id, (}

FOLLOW sets

L = { $,), +}
S = { $ }
E = { $,), + }

THE END