# Algorithm to compute FIRST, FOLLOW and nullable

Initialize FIRST and FOLLOW to all empty sets and nullable to all false.

For each terminal symbol Z

FIRST(Z)={Z};

repeat

For each production $X->Y_1Y_2\ldots Y_k$

For each i from 1 to k, each j from i+1 to k

if all the $Y_i$ are nullable

then nullable[X]=true

if $Y_1\ldots Y_{i-1}$ are all nullable

then FIRST[X]=FIRST[X] U FIRST[$Y_i$]

if $Y_{i+1}..Y_k$ are all nullable

then FOLLOW[$Y_i$]=FOLLOW[$Y_i$]UFOLLOW[X]

if $Y_{i+1}..Y_{j-1}$ are all nullable

then FOLLOW[$Y_i$]=FOLLOW[$Y_i$] U FIRST[$Y_j$]

until FIRST and FOLLOW nullable did not change in this iteration.

# Example for compute nullable FIRST() and FOLLOW()

| | | |
|---|---|---|
| Z->d | Y-> | X->Y |
| Z->XYZ | Y->c | X->a |

Grammar

| | nullable | FIRST | FOLLOW |
|---|---|---|---|
| X | no | | |
| Y | no | | |
| Z | no | | |

*Initial*

| | nullable | FIRST | FOLLOW |
|---|---|---|---|
| X | no | a | cd |
| Y | yes | c | d |
| Z | no | d | |

*Iterative* 1

| | nullable | FIRST | FOLLOW |
|---|---|---|---|
| X | yes | ac | acd |
| Y | yes | c | acd |
| Z | no | acd | |

Iterative 2

*In iterative 3, because no new information is generated, the algorithm stops.*

# Example for generating predictive parsing table

| | | |
|---|---|---|
| S->E$ | | |
| E->E+T | T->T*F | F->id |
| E->E-T | T->T/F | F->num |
| E->T | T->F | F->(E) |

**Grammar**

| | | |
|---|---|---|
| S->E$ | | |
| E->TE’ | T->FT’ | F->id |
| E’->+TE’ | T’->*FT’ | |
| E’->-TE’ | T’->/FT’ | F->num |
| E’-> | T’-> | F->(E) |

**Derived Grammar -eliminating left recursion**

# • Compute nullable FIRST() and FOLLOW()

| | NULLABLE | FIRST | FOLLOW |
|---|---|---|---|
| S | no | ( id num | |
| E | no | ( id num | )$ |
| E’ | yes | + - | )$ |
| T | no | ( id num | )+-$ |
| T’ | yes | * / | )+-$ |
| F | no | ( id num | )*/+-$ |

- **Build a predictive parsing table.**
  - **Enter production X→r in row X, column T of the table for each T ∈FIRST(r)**
  - **If r is nullable, enter the production in row X, column T for each T ∈FOLLOW(X)**

| | +, - | *, / | Id, num | ( | ) | $ |
|---|---|---|---|---|---|---|
| S | | | S->E$ | S->E$ | | |
| E | | | E->TE'$ | E->TE'$ | | |
| E' | E'->+TE' | | | | E'-> | E'-> |
| T | | | T->FT'$ | T->FT$ | | |
| T' | T'-> | T'->*FT' | | | T'-> | T'-> |
| F | | | F->id | F->( | | |