(p e,e) (q,s)
(q,e,S),(a,ABa)
(q,e,A),(q,aa)
((q,e,A),(q,e))
((q,e,B),(q,bb))
((q,e,a),(q,e))
((q,e,b),(q,e))

((p,e,e), (q,S))
((q,a,e), (qa,e))
((q,b,e),(qb,e))
((qa,e,S),(q,ABa))

((qa,e,S),(q,ABa))
((qb,e,S),(q,ABa))
((qb,e,A),(qb,e))
((qb,e,B),(qb,bb))
((q,$,e),(q$,e))

1. ((p ,e ,e), (q, S))
2. ((q ,a, e), (q_a, e))
3. ((q ,b, e), (q_b, e))
4. ((q_a ,e, A), (q, aa))
5. ((q_a ,e, S), (q, ABa))
6. ((q_b ,e, S), (q, ABa))
7. ((q_b ,e, B), (q, bb))
8. ((q_b ,e, A), (q, e))
9. ((q ,$, e), ($q_\$$, e))

FIRST

First(S) = FIRST(A)

∪

since A→e

First(B)

First(A) = {a, e}

First(B) = {b}

FOLLOWS

Follows(S) → {$}

Follows(A) → First(B) = {b}

Follows(B) → {a}

Grammar G: S→ ABa; A→ aa | e; B→bb

Deterministic Look-ahead transitions:

1. ((p ,e ,e), (q, S))
2. ((q ,a, e), (q_a, e))
3. ((q ,b, e), (q_b, e))
4. ((q_a ,e, A), (q, aa)) // since a is in FIRST of A
5. ((q_a ,e, S), (q, ABa)) // since a is in FRIST of S
6. ((q_b ,e, S), (q, ABa)) // since b is in FRIST of S
7. ((q_b ,e, B), (q, bb)) // since b is in FRIST of B
8. ((q_b ,e, A), (q, e)) // since b is in FOLLOWS of A
9. ((q_a ,$, e), ($q_s$, e))

FIRST (S)= FIRST(A) U FIRST(B) = {a,b}

FIRST (A)= {a} U {e}= {a,e}

FIRST (B)= {b}

FOLLOWS (S)= {$}

FOLLOWS (A)= FIRST (B)= {b}

FOLLOWS (B)= {a}

Click to add text

E→T
E→E+T
T→T*F
T→F
F→(E)
F→id
F→id(E)

E→ TE'
E'→ +TE'
E'→e
T→ FT'
T'→ *FT'
T'→e
F→(E)
F→idA
A→ e
A→ (E)

left factoring

(Never contains null)

NT	First	Follows
E	FIRST(T) = FIRST(F) = {'(', id}	{$,)}
E'	{+, e}	{$,)}
T	{'(', id}	{$, +,)}
T'	{*, e}	{), +, $}
F	{(, id}	First(T') = {*,), +, $}

since E' → e & First(e) = First(E')

E → TE'
E' → +TE'

now we need lookahead parser !

in tutorials !