Earley States

- Let \( X \) be a non-terminal
- Let \( a \) and \( b \) be (possibly-empty) sequences of terminals and non-terminals
- Let \( X \rightarrow ab \) be a production in your grammar
- Let \( j \) be a position in the input
- Each Earley State is a tuple \( < X \rightarrow a \cdot b, j > \)
  - We are currently parsing an \( X \)
  - We have seen \( a \), we expect to see \( b \)
  - We started parsing this \( X \) after seeing the first \( j \) tokens from the input.

Formal shift operation

- Whenever
  - chart[i] contains \( < X \rightarrow ab \cdot cd, j > \)
  - \( c \) is a terminal (not a non-terminal)
  - the \( (i+1)^{th} \) input token is \( c \)
- The shift operation
  - Adds \( < X \rightarrow abc \cdot d, j > \) to chart[i+1]

Formal closure operation

- Whenever
  - chart[i] contains \( < X \rightarrow ab \cdot cd, j > \)
  - \( c \) is a non-terminal
  - The grammar contains \( < c \rightarrow p \ q \ r > \)
- The closure operation
  - Adds \( < c \rightarrow p \ q \ r, i > \) to chart[i]

- Note \( < c \rightarrow p \ q \ r, i > \) because “we started parsing this \( c \) after seeing the first \( i \) tokens from the input.”

Formal reduce operation

- Whenever
  - chart[i] contains \( < X \rightarrow ab \cdot , j > \)
  - chart[j] contains \( < Y \rightarrow q \ X \ r, k > \)
- The reduce operation
  - Adds \( < Y \rightarrow q \ X \ r, k > \) to chart[i]

- Note \( < Y \rightarrow q \ X \ r, k > \) because “we started parsing this \( Y \) after seeing the first \( k \) tokens from the input.”
Massive Earley Example

Grammar
S → F
F → id ( A )
A → N
A → ε
N → id
N → id , N

Input
id ( id , id )

--- | --- | --- | --- | --- | --- | ---
S → F , 0

#58