CS536

Intro to Parsing

Last Time

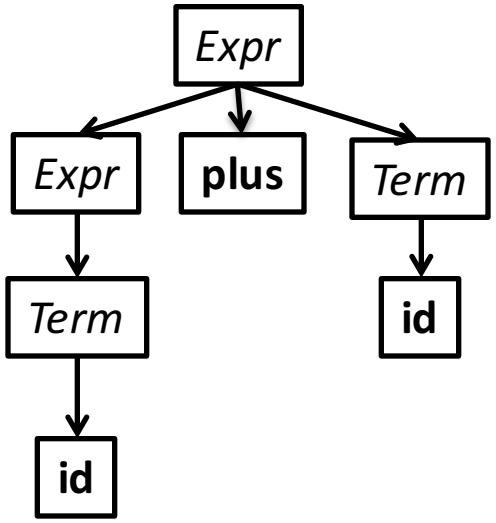
- Showed how to blindly use CUP for getting ASTs
- But we never saw HOW the parser works

This Time

- Dip our toe into parsing
  - Approaches to Parsing
  - CFG Transformations
    - Useless Nonterminals
    - CNF: A form of grammar that's easier to deal with
  - CYK:
    - powerful, heavyweight approach to parsing

Approaches to Parsing

- Top Down / “Goal driven”
  - Start at root of parse tree, grow downward to match the string
- Bottom Up / “Data Driven”
  - Start at terminal, generate subtrees until you get to the start



CYK: A general approach to Parsing

- Operates in $O(n^3)$
- Works Bottom-Up
- Only takes a grammar in CNF
  - This will not turn out to be a limitation

Chomsky Normal Form

- All rules must be one of two forms:

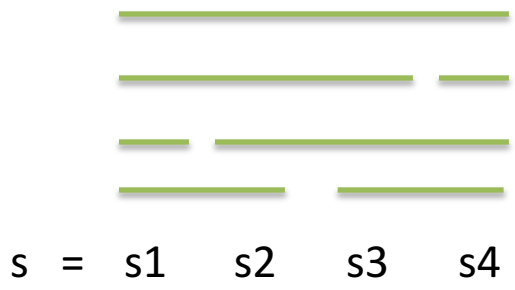
  $X \longrightarrow \mathbf{t}$

  $X \longrightarrow A\ B$

- The only rule allowed to derive epsilon is the start *S*, in which case it's forbidden on the RHS of any rule

What CNF buys CYK

Fact that nonterminals come in pairs allows you to think of subtree as a subspan of the input



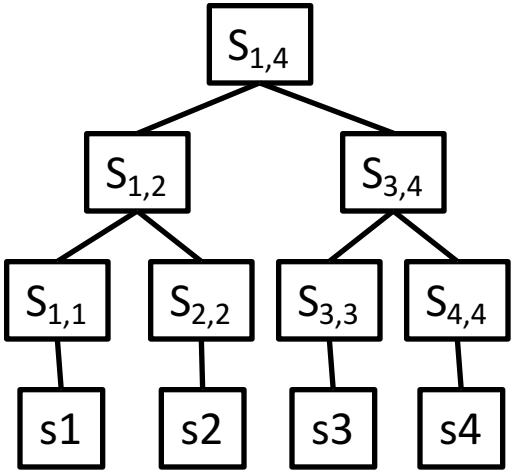
CYK: Dynamic Programming

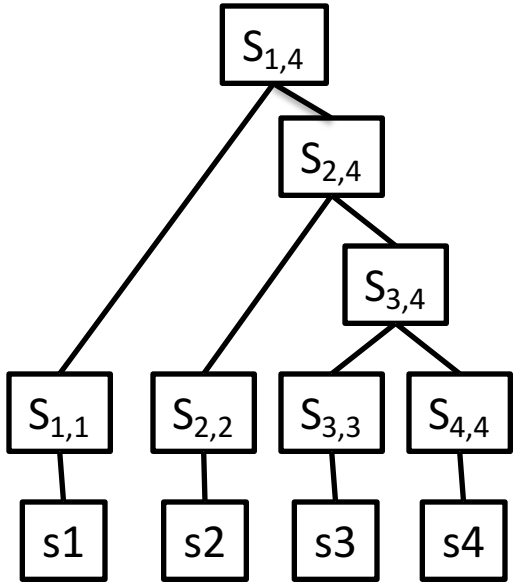
$X \longrightarrow \mathbf{t}$

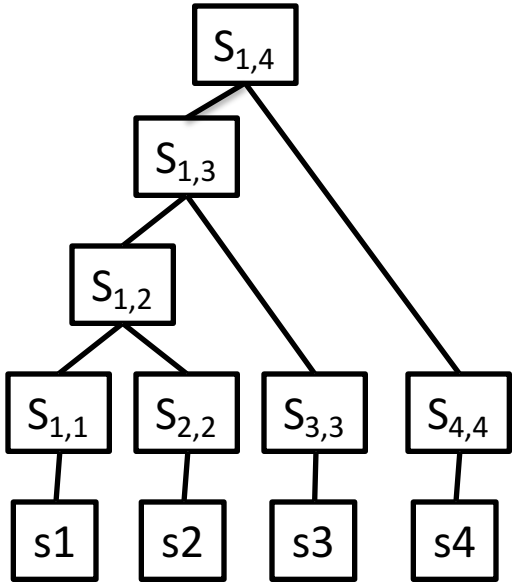
Prods. form the leaves of the parse tree

$X \longrightarrow A\ B$

Form binary nodes

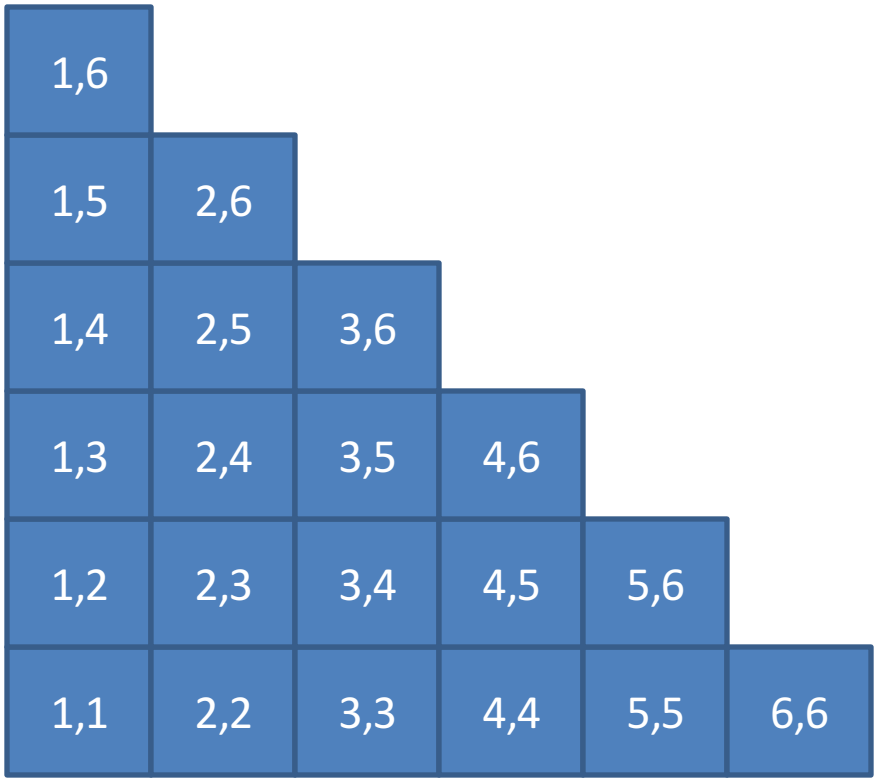




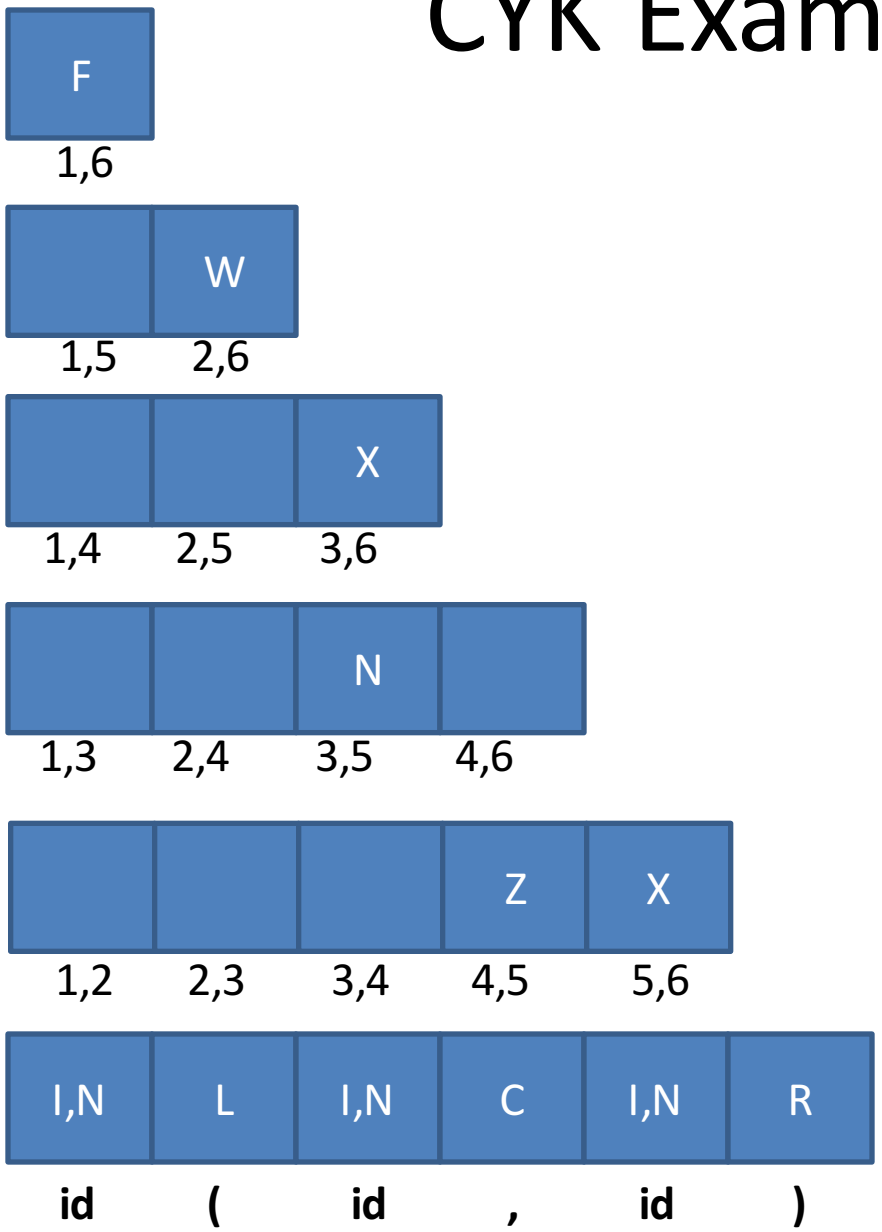


Running CYK…

– Track every viable subtree from leaf to root. Here are all the subspans for a string of 6 terminals

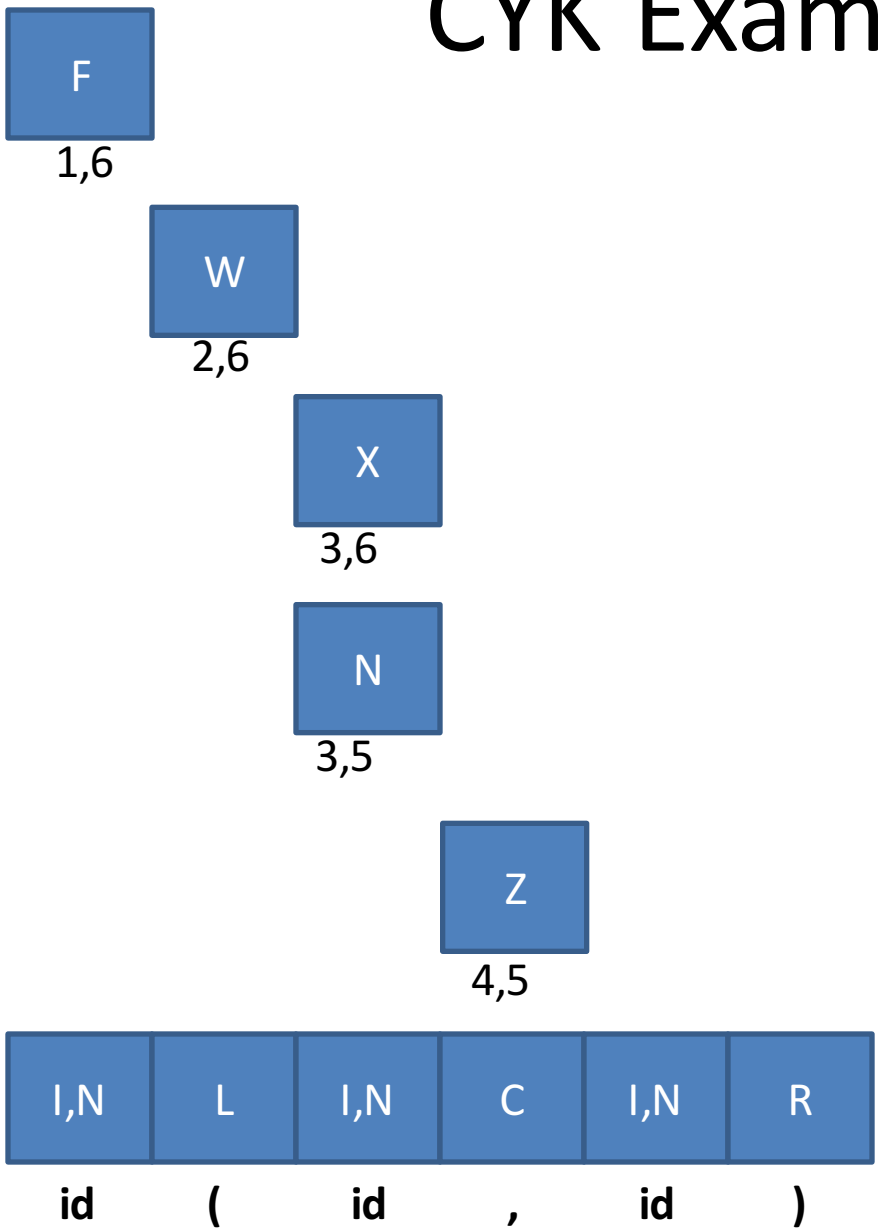


CYK Example



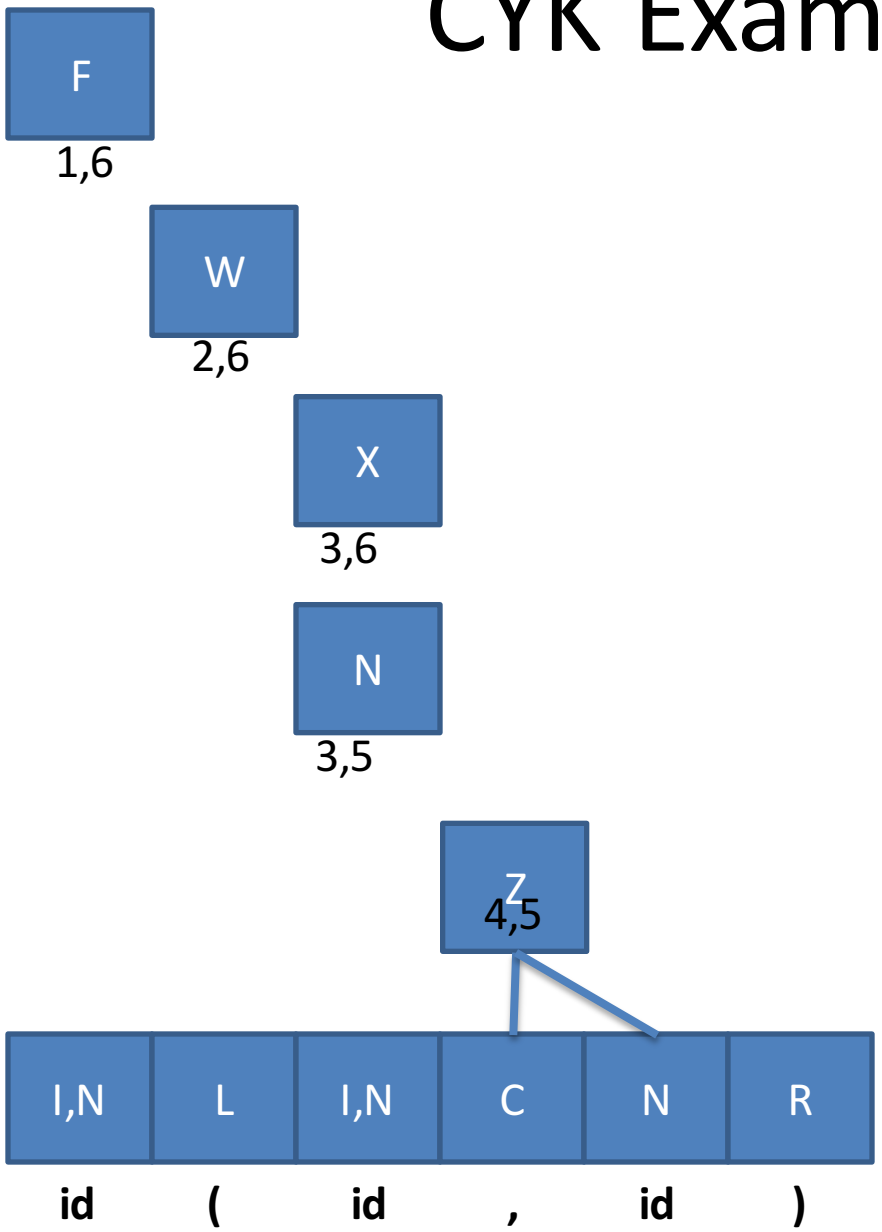
F	→	I W
F	→	I Y
W	→	L X
X	→	N R
Y	→	L R
N	→	**id**
N	→	I Z
Z	→	C N
I	→	**id**
L	→	**(**
R	→	**)**
C	→	**,**

CYK Example



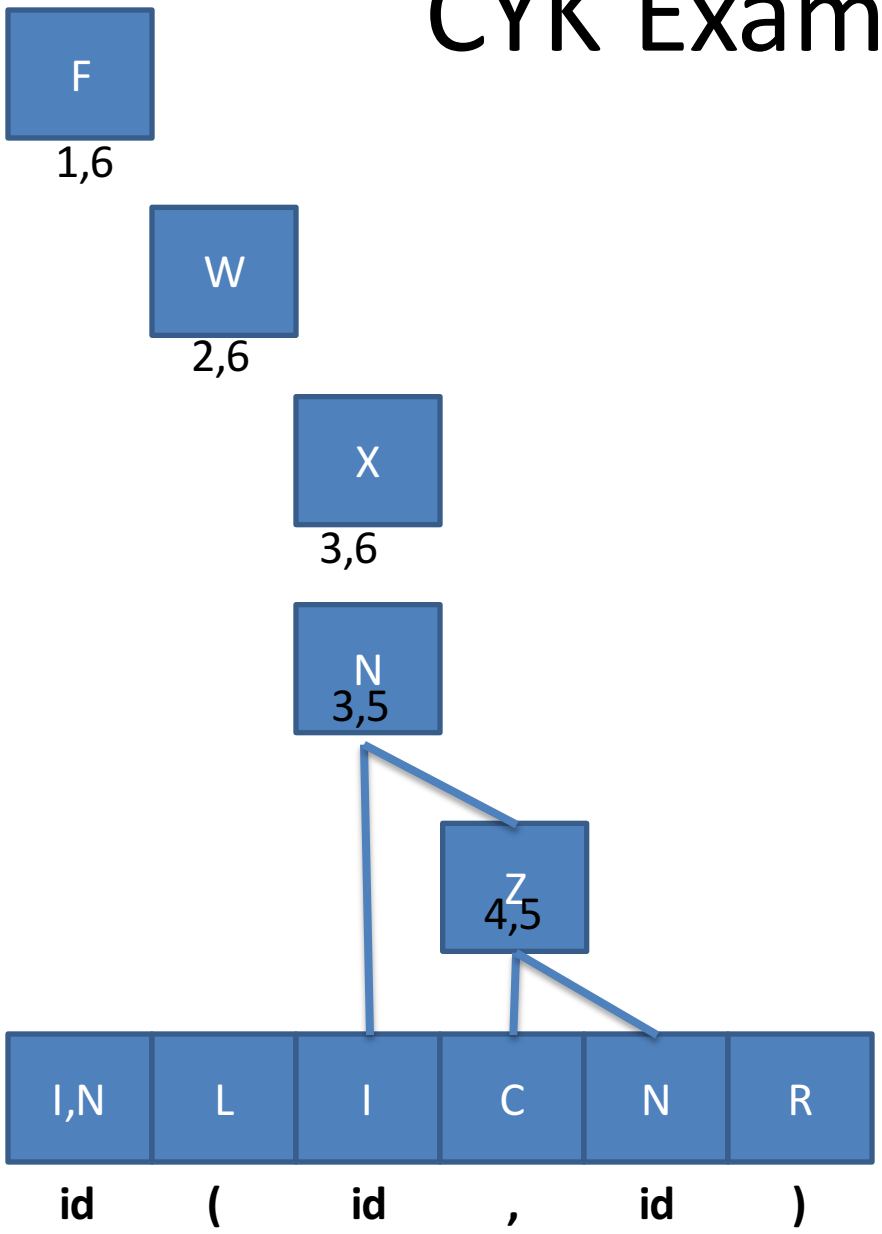
F	→	I W
F	→	I Y
W	→	L X
X	→	N R
Y	→	L R
N	→	**id**
N	→	I Z
Z	→	C N
I	→	**id**
L	→	**(**
R	→	**)**
C	→	**,**

CYK Example



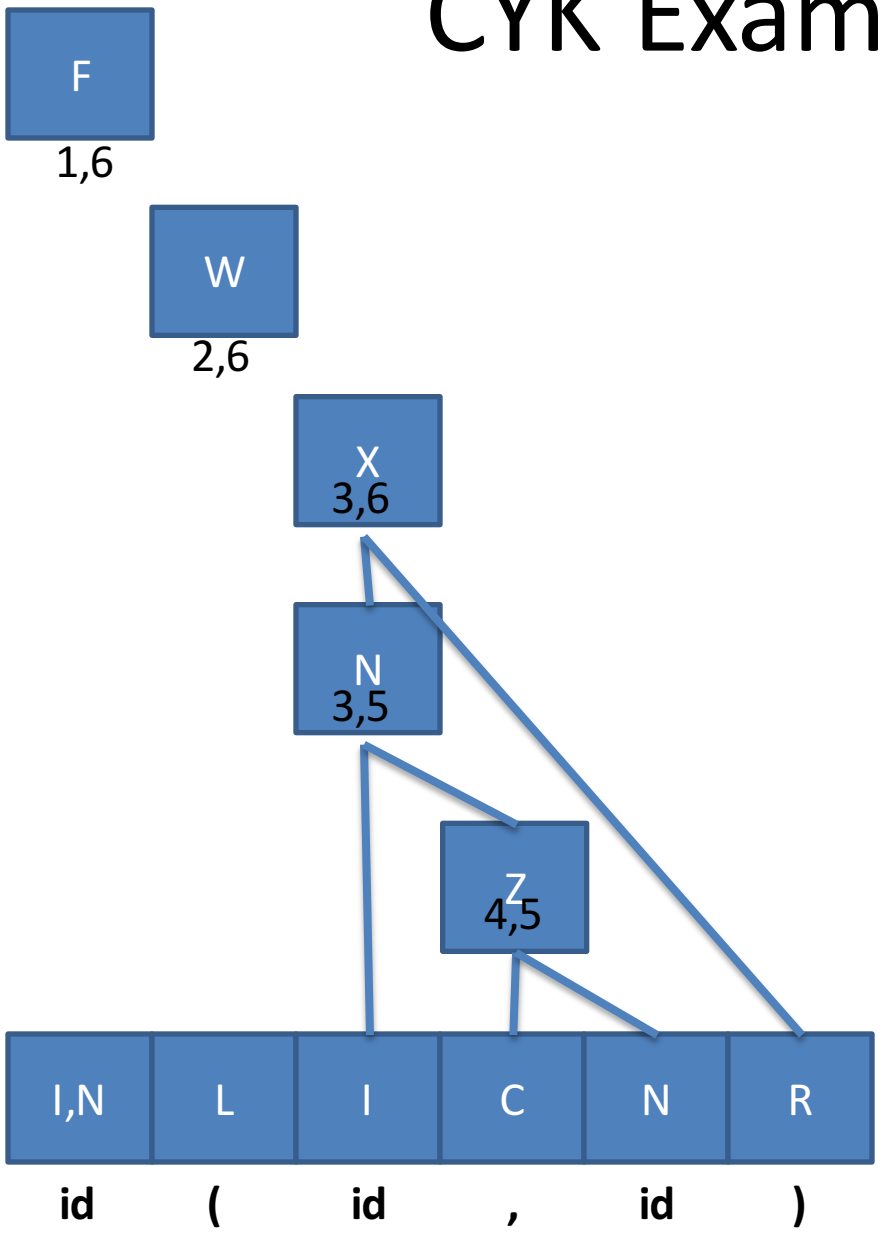
F	→	I W
F	→	I Y
W	→	L X
X	→	N R
Y	→	L R
N	→	**id**
N	→	I Z
Z	→	C N
I	→	**id**
L	→	**(**
R	→	**)**
C	→	**,**

CYK Example



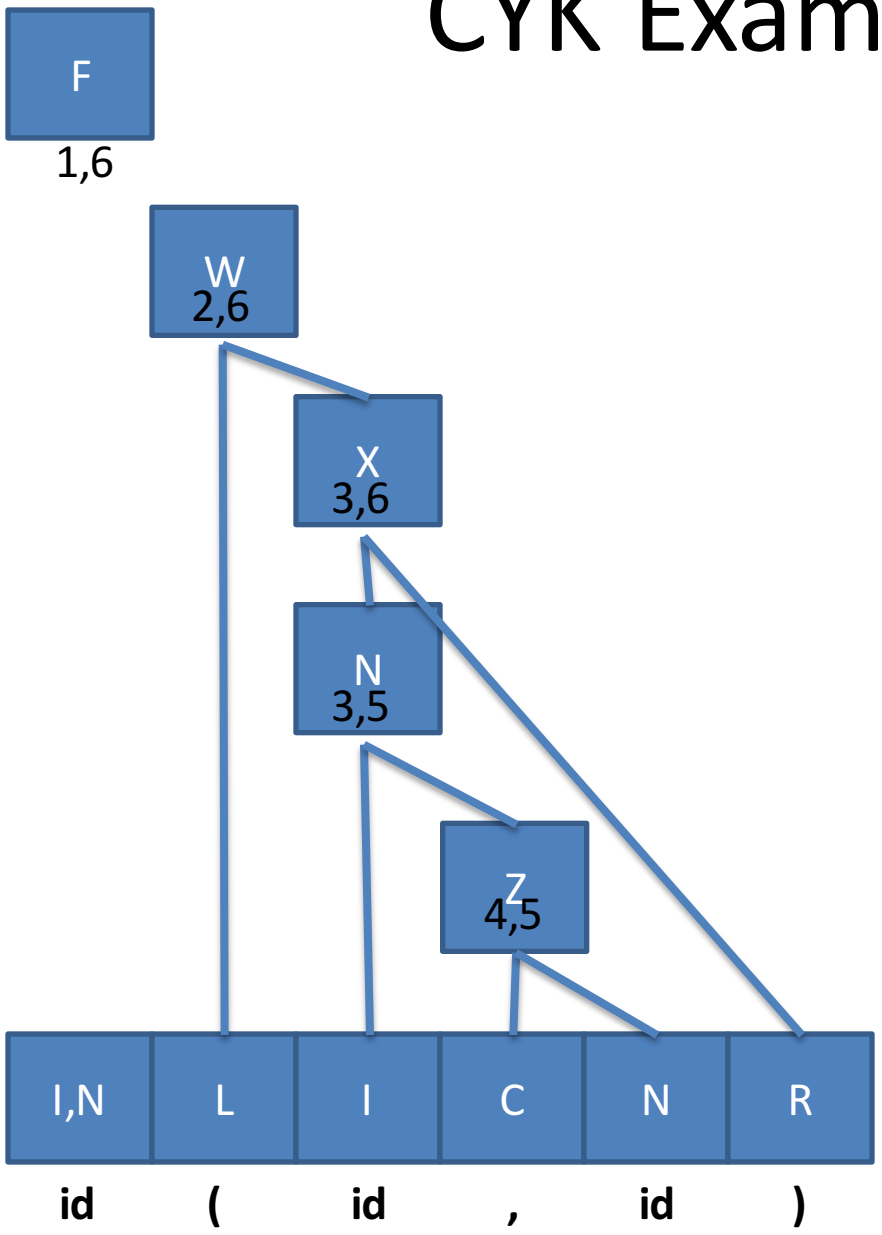
F	→	I W
F	→	I Y
W	→	L X
X	→	N R
Y	→	L R
N	→	**id**
N	→	I Z
Z	→	C N
I	→	**id**
L	→	**(**
R	→	**)**
C	→	**,**

CYK Example



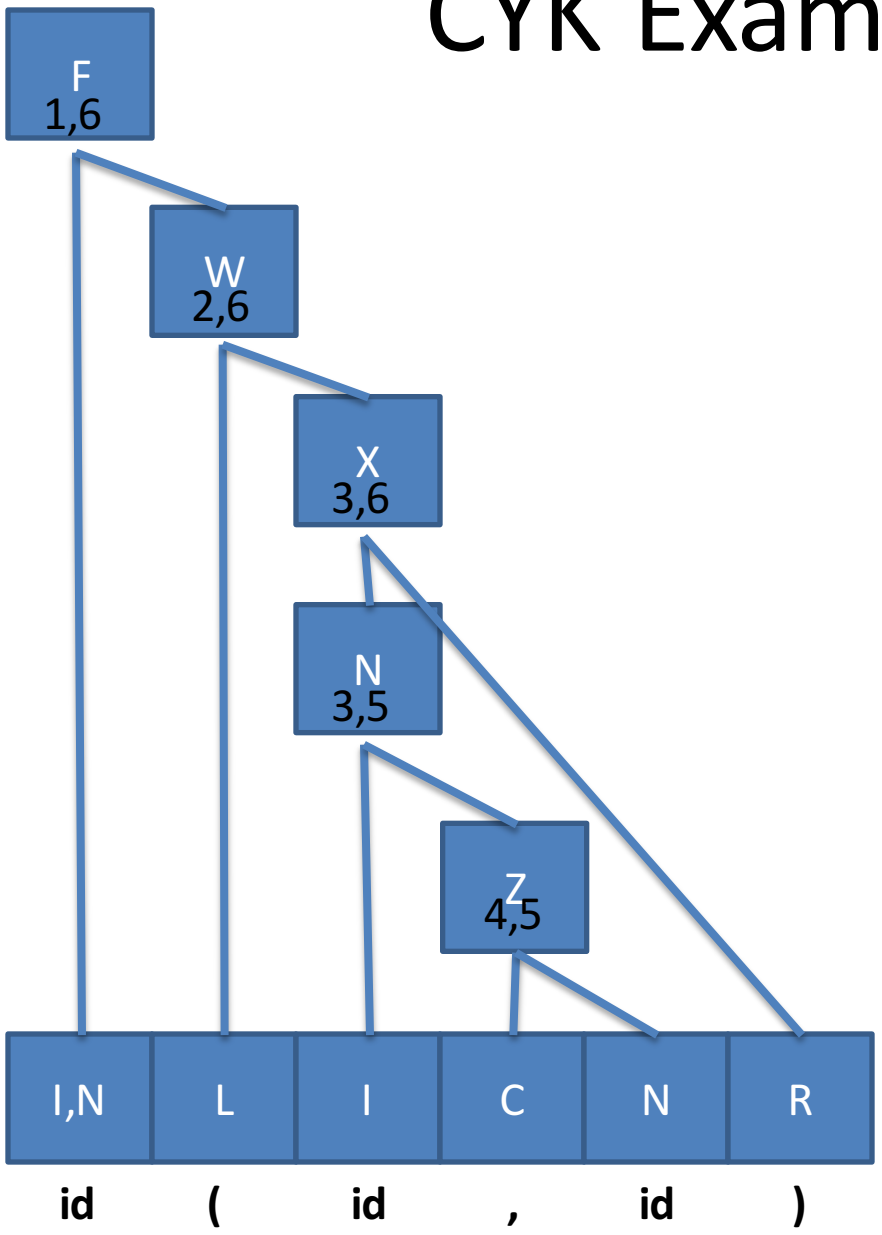
F	→	I W
F	→	I Y
W	→	L X
X	→	N R
Y	→	L R
N	→	**id**
N	→	I Z
Z	→	C N
I	→	**id**
L	→	**(**
R	→	**)**
C	→	**,**

CYK Example



F	→	I W
F	→	I Y
W	→	L X
X	→	N R
Y	→	L R
N	→	**id**
N	→	I Z
Z	→	C N
I	→	**id**
L	→	**(**
R	→	**)**
C	→	**,**

CYK Example



F	→	I W
F	→	I Y
W	→	L X
X	→	N R
Y	→	L R
N	→	**id**
N	→	I Z
Z	→	C N
I	→	**id**
L	→	**(**
R	→	**)**
C	→	**,**

Cleaning up our grammars

- We want to avoid unnecessary work
  - Remove *useless* rules

Eliminating Useless Nonterminals

1. If a nonterminal cannot derive a terminal symbol then it is useless
2. If a nonterminal cannot be derived from the start symbol, then it is useless

Eliminate Useless Nonterms

- If a nonterminal cannot derive a terminal symbol, then it is useless

```
Mark all terminal symbols
Repeat
    If all symbols on the
    righthand side of a
    production are marked
        mark the lefthand side
Until no more non-terminals
can be marked
```

Example:

$$
\begin{array}{lcl}
S & \rightarrow & X \mid Y \\
X & \rightarrow & ( ) \\
Y & \rightarrow & (Y Y)
\end{array}
$$

Eliminate Useless Nonterms

- If a nonterminal cannot be derived from the start symbol, then it is useless

```
Mark the start symbol
Repeat
    If the lefthand side of a
    production is marked
        mark all righthand
        non-terminal
Until no more non-terminals
can be marked
```

Example:

S	→	A B
A	→	+ | - | ε
B	→	**digit** | B **digit**
C	→	. B

Chomsky Normal Form

- 4 Steps
  - Eliminate epsilon rules
  - Eliminate unit rules
  - Fix productions with terminals on RHS
  - Fix productions with > 2 nonterminals on RHS

Eliminate (Most) Epsilon Productions

- If a nonterminal *A* immediately derives epsilon
  - Make copies of all rules with *A* on the RHS and delete all combinations of *A* in those copies

Example 1

F	→	**id (A)**
A	→	ε
A	→	N
N	→	**id**
N	→	**id , N**

F	→	**id (A)**
F	→	**id ( )**
A	→	N
N	→	**id**
N	→	**id , N**

Example 2

$$
\begin{array}{lcl}
X & \longrightarrow & A\ \mathbf{x}\ A\ \mathbf{y}\ A \\
A & \longrightarrow & \varepsilon \\
A & \longrightarrow & \mathbf{z}
\end{array}
$$

$$
\begin{array}{lcl}
X & \longrightarrow & A\ \mathbf{x}\ A\ \mathbf{y}\ A \\
 & | & A\ \mathbf{x}\ A\ \mathbf{y} \\
 & | & A\ \mathbf{x}\ \mathbf{y}\ A \\
 & | & A\ \mathbf{x}\ \mathbf{y} \\
 & | & \mathbf{x}\ A\ \mathbf{y}\ A \\
 & | & \mathbf{x}\ A\ \mathbf{y} \\
 & | & \mathbf{x}\ \mathbf{y}\ A \\
 & | & \mathbf{x}\ \mathbf{y} \\
A & \longrightarrow & \mathbf{z}
\end{array}
$$

Eliminate Unit Productions

- Productions of the form $A \longrightarrow B$ are called unit productions
- Place B anywhere A could have appeared and remove the unit production

Example 1

F	→	**id (A)**
F	→	**id ( )**
A	→	N
N	→	**id**
N	→	**id , N**

F	→	**id (N)**
F	→	**id ( )**
N	→	**id**
N	→	**id , N**

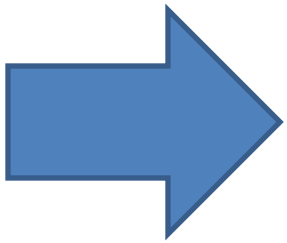
Fix RHS Terminals

- For productions with Terminals and something else on the RHS
  - For each terminal **t** add the rule

    $X \longrightarrow \mathbf{t}$
  - Replace t with X in the in the original rules

Example

F	→	**id (N)**
F	→	**id ( )**
N	→	**id**
N	→	**id , N**



F	→	I L N R
F	→	I L R
N	→	**id**
N	→	I C N
I	→	**id**
L	→	**(**
R	→	**)**
C	→	**,**

Fix RHS Nonterminals

- For productions with > 2 Nonterminals on the RHS
  - Replace all but the *first* nonterminal with a new nonterminal
  - Add a rule from the new nonterminal to the replaced nonterminal sequence
  - Repeat

Example

F → I L N R

F → I W

W → L N R

F → I W

W → L X

X → N R

Parsing is Tough

- CYK parses an arbitrary CFG, but
  - $O(n^3)$
  - Too slow!
- For special class of grammars
  - $O(n)$
  - Includes LL(1) and LALR(1)

Classes of Grammars

- LL(1)
  - Scans input from Left-to-right (first L)
  - Builds a Leftmost Derivation (second L)
  - Can peek (1) token ahead of the token being parsed
  - Top-down “predictive parsers”
- LALR(1)
  - Uses special lookahead procedure (LA)
  - Scans input from Left-to-right (second L)
  - Rightmost derivation (R)
  - Can also peek (1) token ahead
- LALR(1) strictly more powerful, much harder to understand

In summary

- We talked about how to parse with CYK and CNF