# CS 536 Announcements for Wednesday, February 28, 2024

**Midterm 1, Thursday, February 29, 7:30 – 9 pm, S429 Chemistry**

**Last Time**

- review for Midterm 1

**Today**

- approaches to parsing
- bottom-up parsing
- CFG transformations
  - removing useless non-terminals
  - Chomsky normal form (CNF)
- CYK algorithm

In addition to printed copies of the overheads, there are full-sized versions of the diagrams for running the CYK algorithm available

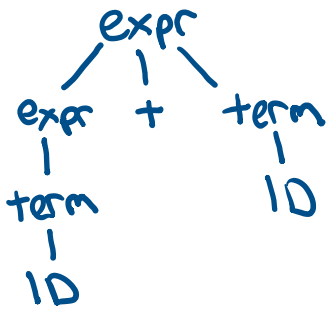
**Next Time**

- wrap up CYK
- classes of grammars
- top-down parsing

## Parsing: two approaches

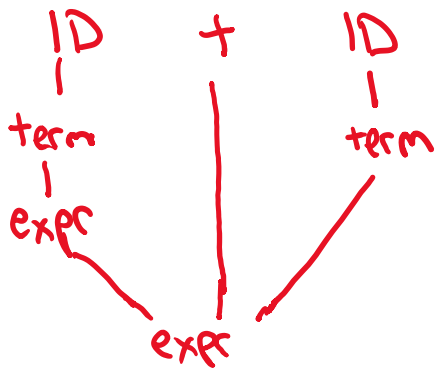
**Top-down / "goal driven"**

- start at start nonterminal
- grow parse tree downward until entire sequence is matched




**Bottom-up / "data driven"**

- start with terminals (sequence)
- generate ever larger subtrees until get to single tree whose root is the start nonterminal (note: parse tree is upside down)




**Example:**

**CFG**: expr → expr + term | term

term → term * ID | ID

**Derive**: ID + ID

# Cocke – Younger – Kasami (CYK) algorithm

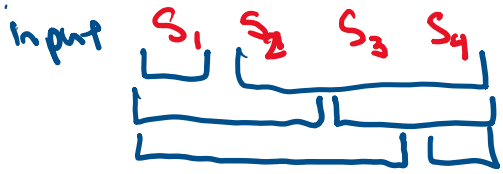
- Works bottom-up
- Time complexity : $O(n^3)$ n = length of input (# of tokens in sequence)
- Requires grammar to be in Chomsky Normal Form

**Chomsky Normal Form (CNF)**

- all rules must be in one of two forms
  - $x \rightarrow \mathbf{T}$ (T is a terminal)
  - $x \rightarrow a\ b$
- only rule allowed to derive epsilon is the start symbol *s*

**Why CNF is helpful?**

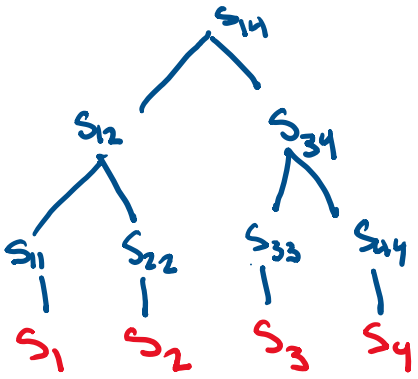
- nonterminals in pairs → can think of a subtree as a subspan
- nonterminals (except start) can't derive epsilon → each subspan has at least 1 token

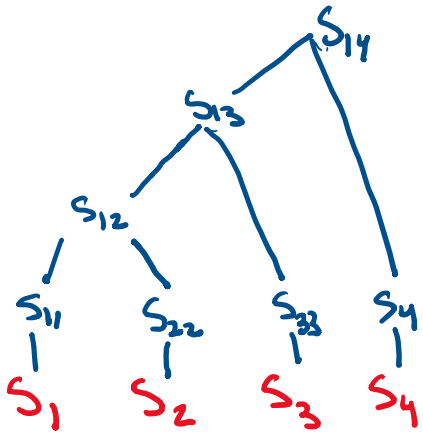



**CYK : Dynamic Programming**

$x \rightarrow \mathbf{T}$ → forms leaf of parse tree

$x \rightarrow a\ b$ → binary interior node of parse tree

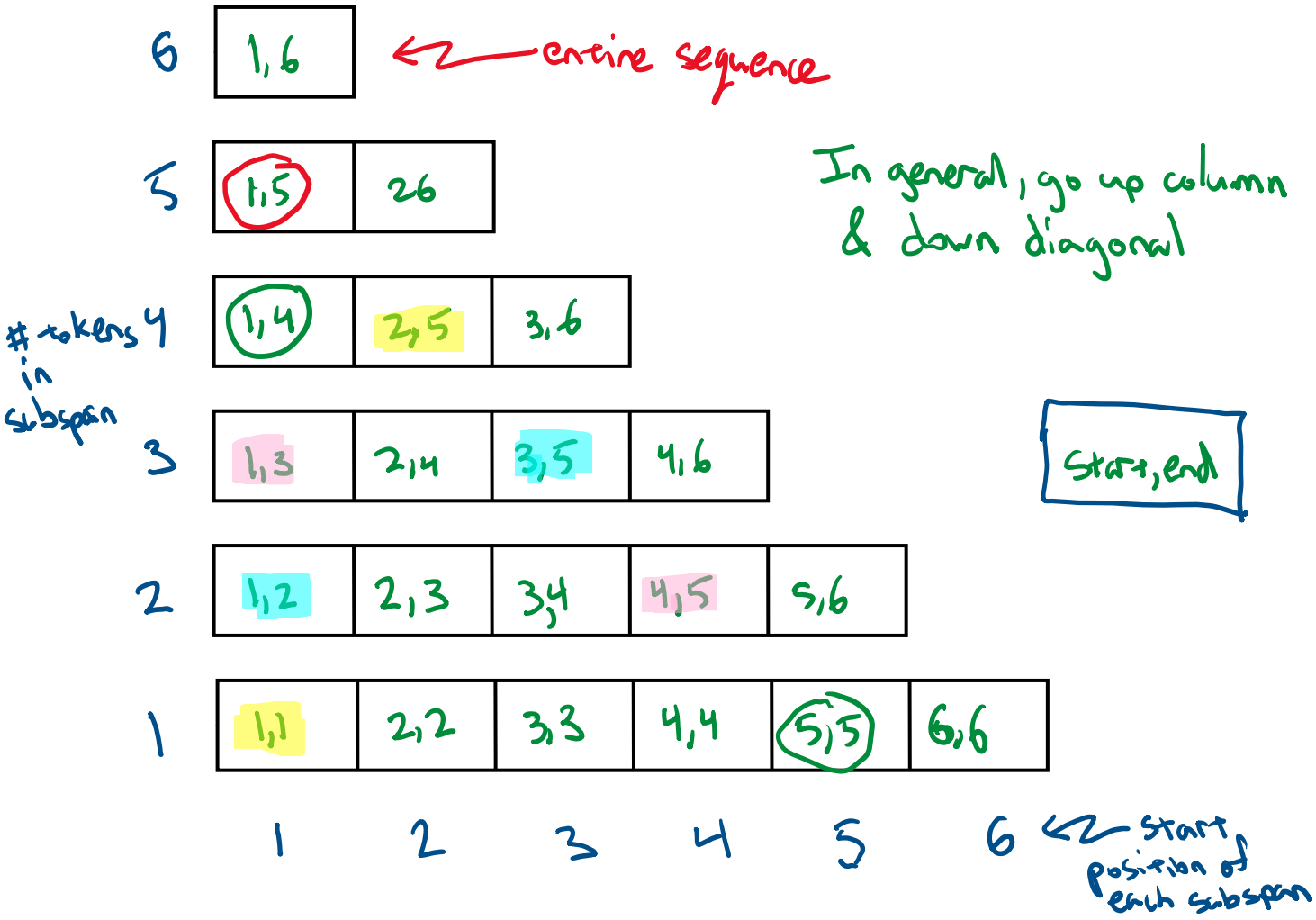


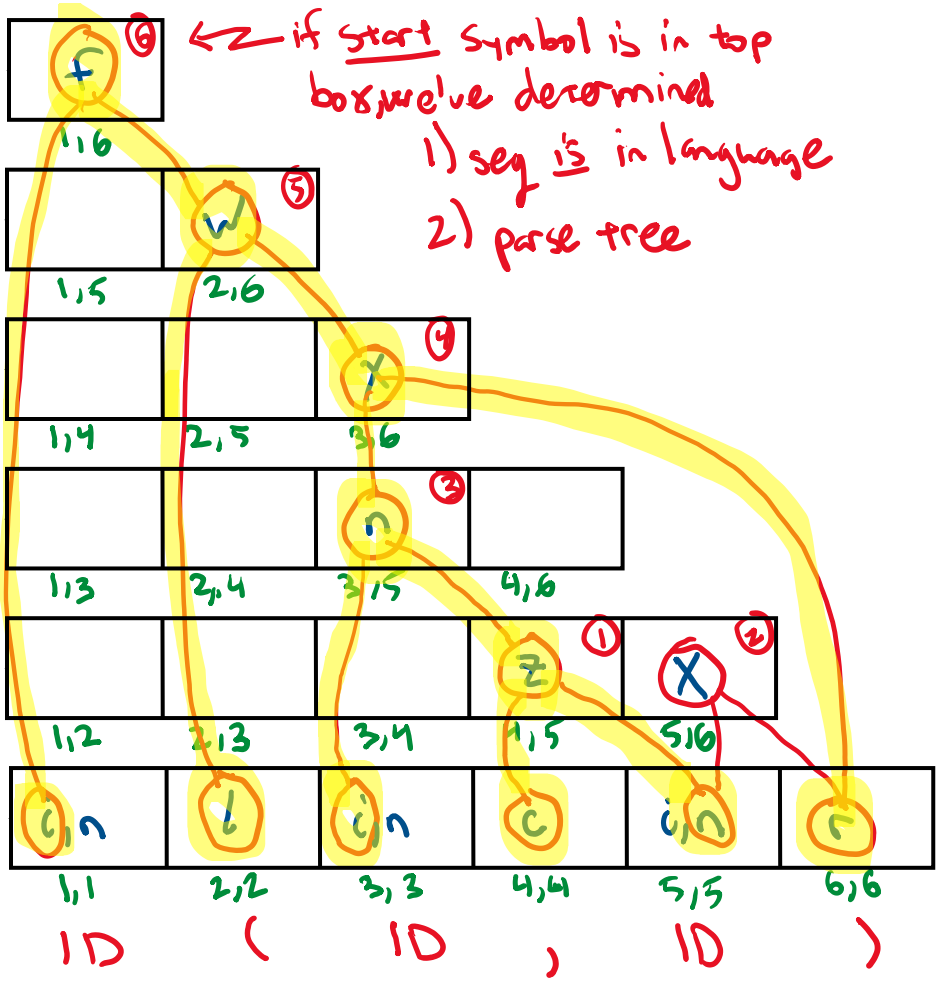
# Running CYK

Track every viable subtree from leaf to root.

All subspans for a sequence (string) with 6 terminals



# CYK Example




$f \rightarrow i\ w$ ⑥

$f \rightarrow i\ y$

$w \rightarrow l\ x$ ⑤

$x \rightarrow n\ r$ ② ④

$y \rightarrow l\ r$

$n \rightarrow$ **ID**

$n \rightarrow i\ z$ ③

$z \rightarrow c\ n$ ①

$i \rightarrow$ **ID**

$l \rightarrow$ **(**

$r \rightarrow$ **)**

$c \rightarrow$ **,**

# Eliminating useless nonterminals

**Avoid unnecessary work** – remove ***useless*** rules

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

**Nonterminals that cannot derive a sequence of terminal symbols**

```
mark all terminal symbols
repeat
    if all symbols on the RHS of a production are marked
        mark the LHS nonterminal (everywhere it shows up)
until no more nonterminals can be marked
```

Example

*s* → *x* | *y*

*x* → ( )

*y* → ( *y* )

y is useless

S ⇒ x

S ⇒ y

S ⇒ x | y

**Nonterminals that cannot be derived from the start symbol**

```
mark the start symbol
repeat
    if the LHS of a production is marked
        mark all RHS nonterminals (wherever they show up)
until no more nonterminals can be marked
```

Example

*s* → *a b*

*a* → **+** | **−** | ε

*b* → **digit** | *b* **digit**

*c* → **.** *b*

c is useless

# Chomsky Normal Form

**Four steps**

- eliminate epsilon productions
- eliminate unit productions
- fix productions with terminal on RHS (along w/ other stuff)
- fix productions with > 2 nonterminals on RHS

new start symbol
$S_0 \to S$
$S \to \varepsilon$
$a \to \ldots S \ldots$

ok to have start $S \to \varepsilon$
but if so can't have start on RHS

**Eliminate (most) epsilon productions**

If nonterminal *a* immediately derives epsilon
- make copies of all rules with a on RHS & delete all combinations of a in copies

Example 1

*f* → **ID** ( *a* )
*a* → ε
*a* → *n*
*n* → **ID**
*n* → **ID** , *n*

$f \to ID(a)$
$f \to ID(\not{a}) \Rightarrow f \to ID()$
$a \to n$
$n \to ID$
$n \to ID, n$

Example 2

*x* → *a* **X** *a* **Y** *a*
*a* → ε
*a* → **Z**

$X \to aXaYa \mid aXaY \mid aXYa \mid XaYa \mid aXY \mid XaY \mid XYa \mid XY$
$a \to Z$

# Chomsky Normal Form (cont.)

**Eliminate unit productions**

Productions of the form $a \rightarrow b$ are called ***unit productions***

If this is the only rule with a on LHS, place b anywhere a could have appeared & remove unit production

Example

*f* → **ID** ( *a* )
*f* → **ID** ( )
*a* → *n*
*n* → **ID**
*n* → **ID** , *n*

f → ID(n)
| ID( )
n → ID
| ID, n

If there are multiple rules with a on LHS, for each rule of the form $b \rightarrow \delta$, add $a \rightarrow \delta$ & remove $a \rightarrow b$

a → bX
| cb
| b
b → ZY
| Yc
c → Za

→

a → bX
| cb
| ZY
| Yc
b → ZY
| Yc
c → Za

# Chomsky Normal Form (cont.)

**Fix RHS nonterminals**

For productions with terminals and something else on the RHS

- for terminal T, add rule X → T
  where X is a new non-terminal
- replace T with X in those productions

Example

*f* → **ID ( *n* )**
*f* → **ID ( )**
*n* → **ID**
*n* → **ID , *n***

i → ID
l → (
r → )
c → ,
f → i l n r
f → i l r
n → ID
n → i c n

For productions with > 2 nonterminals on the RHS

- replace all but 1st nonterm with new nonterm
- add rule from new nonterm to replaced nonterm sequence
- repeat

Example

f → i l n r ⇒ f → i w ⇒ f → i w
w → l n r ⇒ w → l x
x → n r

Left for you: f → i l r
n → i c n