The Cocke-Kasami-Younger algorithm

Given $w \in \Sigma^*$ & G a CFG

Want to know $w \in L(G)$?

We assume G is in Chomsky Normal Form. The parse trees are binary so for a string of length n we will have a tree with $(2n-1)$ variables. There are exponentially many such trees so we can generate them all, check if they are valid trees that generate w. Exponential time!

We can get this down to $O(n^3)$ using dynamic programming.

Given $G = (V, \Sigma, P, S)$

Input $w = a_1 \cdots a_n \in \Sigma^*$ $a_i \in \Sigma$

We work bottom-up to construct a possible derivation of w. We first ask how we got each individual symbol. Since G is in CNF we have to have used rules of the form $A \longrightarrow a$

We define inductively a 2-indexed family of subsets of V:

$X_{ij} := \{ A \in V \mid A \overset{*}{\Rightarrow} a_i \cdots a_j \}$

BASE CASE $X_{ii} = \{ A \in V \mid A \Rightarrow a_i \}$ $X_{11}, X_{22}, \cdots X_{nn}$

Next row will have $X_{12}, X_{23}, \cdots X_{i(i+1)} \cdots X_{(n-1)n}$

Next row will have $X_{13}, X_{24}, \cdots X_{i(i+2)} \cdots X_{(n-2)n}$

so X_{ij} will be in row $(j-i)+1$.

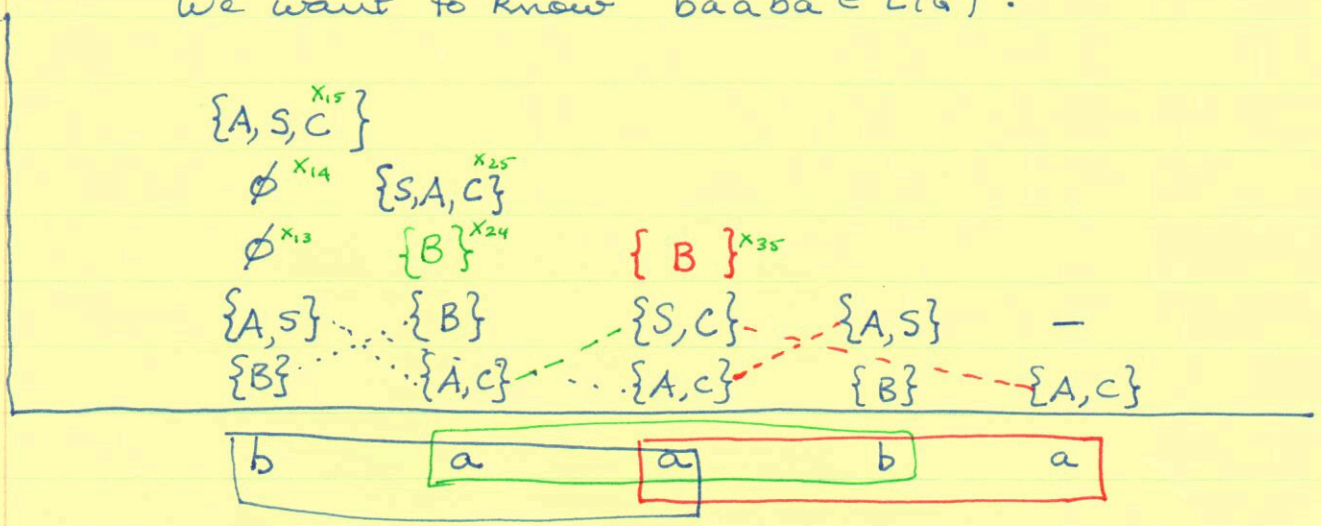
We compute and fill in the table ~~top~~ bottom to top.

When we compute X_{ij} we know X_{ik} & X_{kj} for all & $i \leq k \leq j$

Now if $B \in X_{ik}$, $C \in X_{(k+1)j}$ & $A \to BC$ is a rule we know $A \in X_{ij}$. Why? $B \overset{*}{\Rightarrow} a_i \cdots a_k$ $C \overset{*}{\Rightarrow} a_{k+1} \cdots a_j$ so since $A \to BC \overset{*}{\Rightarrow} a_i \cdots a_j$.

$S \to AB \mid BC \quad A \to BA \mid a \quad B \to CC \mid b \quad C \to AB \mid a$

We want to know $baaba \in L(G)$?



We see that $S \in X_{15}$ so

$S \overset{*}{\Rightarrow} a_1 \ldots a_5$

In general $w \in L(G)$ iff $S \in X_{1n}$

Size of table $O(n^2)$

The time taken to find X_{ij} is $O(j-i)$

The time taken to compare X_{ik} & $X_{(k+1)j}$ & find a variable that generates them is $O(1)$: it depends on the size of the grammar but not on n. So time to compute each X_{ij} is $O(n)$ & so overall $O(n^3)$.

There are better algorithms possible. Best is $O(n^{2.8})$.

— x —

Is a CFL empty? Given G we want to know $L(G) = \emptyset$?

~~Call a variable A productive if there is a rule $A \to \alpha$ where $\alpha \in \Sigma^*$~~

Call a variable A productive if $A \overset{*}{\Rightarrow} \alpha$ where $\alpha \in \Sigma^*$.

Check every variable A & see if $\exists\ A \to \alpha$ with $\alpha \in \Sigma^*$

If so label A productive. Then repeat.

go through all variables B in $V - Prod$ & check if there is a rule $B \rightarrow \beta$ where $\beta \in (\Sigma \cup Prod)^*$
If yes add B to $Prod$. Keep repeating until there are no more changes. If $S \in Prod$ then $L(G) \neq \emptyset$. & $S \notin Prod$ then $L(G) = \emptyset$.
Naively this $O(n^2)$ where n is the # of variables.
A more careful implementation does it in $O(n)$.

Parsing Hugely practical
Extract the parse tree & build it up.
We design DCFL in order to do this : LR(k), LL(k) LALR(k), Rec-descent etc.

$S \rightarrow (S) S \mid \varepsilon$ Unambiguous & deterministic
This is really an inductive definition so a parser for this is naturally recursive:

```
parse ( ):
    if x = ε return nil
    else if getsym = '(' then {
        t0 := parse ();
        if getsym = ')'
        t1 := mktree('(', t0, ')')
        else error
    else error;
t2 := parse ();
if t2 = nil return t1
else return mktree('(', t0, ')', t2)
```

$\{0^i 1^j / \; j = i^2\}$

$A \rightarrow p$

$I \rightarrow 0^p 1^{p^2}$

$A \rightarrow uvwxy = 0^p 1^{p^2}; |vwx| \leq p \quad |vx| > 0$

$I \rightarrow$ case analysis

$\underline{0 - - - 0\,1 - - - - - 1}$

(a) $vwx \rightarrow$ all zeros ; pick any $i \neq 1$

(b) $vwx \rightarrow$ all ones, pick any $i \neq 1$

(c) v overlaps 0's & 1's, x all ones ; pick any $i \neq 1$, 0's 1's out of order same story with v all zero x overlaps

(d) v all zeros , x all ones pick $i = 2$

$|v| = m > 0$, $|x| = q > 0$

$0^{p+m}\,1^{p^2+q}$,

Question is $(p+m)^2 = p^2 + q$?

$p + m^2 = p^2 + 2pm + m^2$

$2pm > p$ so $2pm + m^2 > p \geq q$

Thus we have left the language.

So $0^p 1^{p^2}$ is not context free.

Suppose $L \subseteq a^*$ is not regular. $\Rightarrow$ L not context free

$A \rightarrow p$	$A \rightarrow p$
I choose s.	I choose as before
$A \rightarrow s = xyz \quad \lvert y\rvert \leq p$	$s \rightarrow uvwxy \quad \lvert vx\rvert \leq p$
whatever	I did for the regular case can be mimicked!

Thus L not regular $\Rightarrow$ L not context-free

or L context-free $\Rightarrow$ L regular