CSE 396 Lecture Tue 4/4 Spr 2017 ①

Example CFG:

$$S \to \epsilon \mid aB \mid bA$$

$$A \to SS \mid aS \mid bAa$$

$$B \to bS \mid ABB$$

I 1. S is immediately nullable, so $NULL := \{S\}$.

2. Rule $A \to SS$, $SS \in NULL^*$, so $NULL$ now $= \{S, A\}$.

After that we need only consider the rules ~~for~~ B:

$B \to bS$: RHS has a terminal, so never nullable.

$B \to ABB$: Rule is "self-recursive" but $ABB \notin \{S, A\}^*$, so stop. $NULLABLE = \{S, A\}$.

II: Add rules deleting some occurrences of nullable vars.

$$S_1 \to \cancel{\epsilon} \mid aB \mid bA \mid b$$

$$A \to SS \mid S \mid aS \mid a \mid bAa \mid ba$$

$$B \to bS \mid b \mid ABB \mid BB$$

I could have added $A \to \epsilon$ but it will be removed anyway.

This step created the new "unit rule" $A \to S$. We could replace it by all right-hand sides of S...

III - Delete ϵ-rules. New grammar is G_1.

$$L(G_1) = L(G) \setminus \{\epsilon\}.$$

Recall the general defn: $L_A = \{x \in \Sigma^* : A \Rightarrow^* x\}$. for any variable A.

- A {complies with / is sound for} a target language T_A if $L_A \subseteq T_A$. exactly if $L_A = T_A$.

②

Testing Some Targets — are the variables sound? exact?

$T_S = \{x \in \{a,b\}^* : \#a(x) = \#b(x)\}$. "equal a's, b's"

$T_A = \{x : \#a(x) = \#b(x) + 1\}$ "one more a"

$T_B = \{x : \#b(x) = \#a(x) + 1\}$. "one more b"

Is S sound for T_S? We need to suppose A & B are sound for <u>their</u> targets...

Rule

$S \to \varepsilon$: We have $\varepsilon \in T_S$ since $\#a(\varepsilon) = \#b(\varepsilon) = 0$. So this rule is sound (unconditionally).

$S \to aB$: Assuming B is sound, it generates 1 more b than a. So the total RHS has equal a's and b's. Thus this rule is <u>conditionally</u> sound for S.

$S \to bA$: Similar, by symmetry, <u>if</u> A is sound for T_A.

$A \to SS$: alas this <u>busts</u> T_A, and T_S, T_B are voided too

Let's just <u>delete</u> that rule. New grammar G'

$$G': \begin{array}{l} S \to \varepsilon \mid aB \mid bA \\ A \to aS \mid bAa \\ B \to bS \mid ABB \end{array}$$

Because the verification of the other rules for A and B worked, the variables are now <u>sound</u> for their targets. So $L_S \subseteq T_S$ etc.

Note that in G', A is no longer Nullable, so G_1' is

$$G_1' = \begin{array}{l} S \to aB \mid bA \\ A \to aS \mid \underline{a} \mid bAa \\ B \to bS \mid \underline{b} \mid ABB \end{array}$$

Since the Chomsky NF procedure preserves soundness, these rules are sound for the targets too.

New Q: Are they comprehensive?

(When asking about "exact targets" let's use the letter E: E_S, E_A, E_B etc.

$E_S = \{x : \#a(x) = \#b(x)\}$

$E_A = \{x : \#a(x) = \#b(x) + 1\}$

$E_B = \{x : \#a(x) = \#b(x) - 1\}$.

In generall, proving that rules are exact or comprehensive is ~~hard~~. But we can prove when they're *not*.

To prove they are not, show that some variable (say A) actually complies with a stronger target T'_A such that $E_A \setminus T'_A \neq \emptyset$.

$T'_A = \{x : \#a(x) = \#b(x) + 1$ and: $\boxed{\text{if } x \text{ begins with } b \text{ then it ends with } a}\}$

The string $x = baaab$ belongs to $E_A \setminus T'_A$.

Thus $L_A \subseteq T'_A \subsetneq$ ie $\neq E_A$, so $L_A \neq E_A$ (Indeed, $L_A \subsetneq E_A$).

Does this knock on to S? I.e. is $L_S = L(G_1) \subsetneq E'_S = E_S \setminus \{\varepsilon\}$?

Try $w = b \cdot x = bbaaab$ as a possible counterexample: $w \in E'_S \setminus L_S$

Try G_2: $S \to aB \mid bA$

$A \to aS \mid a \mid \underline{BAA}$

$B \to bS \mid b \mid ABB$.

$\underbrace{b}_{B}\ \underbrace{a}_{A}\ \underbrace{a\ \underline{a}\ b}_{\underbrace{A}_{S}}$

This gives a LM derivation

$S \Rightarrow b\underline{A} \Rightarrow b\underline{B}AA \Rightarrow bbAA \Rightarrow bba\underline{A} \Rightarrow bbaa\underline{S}$

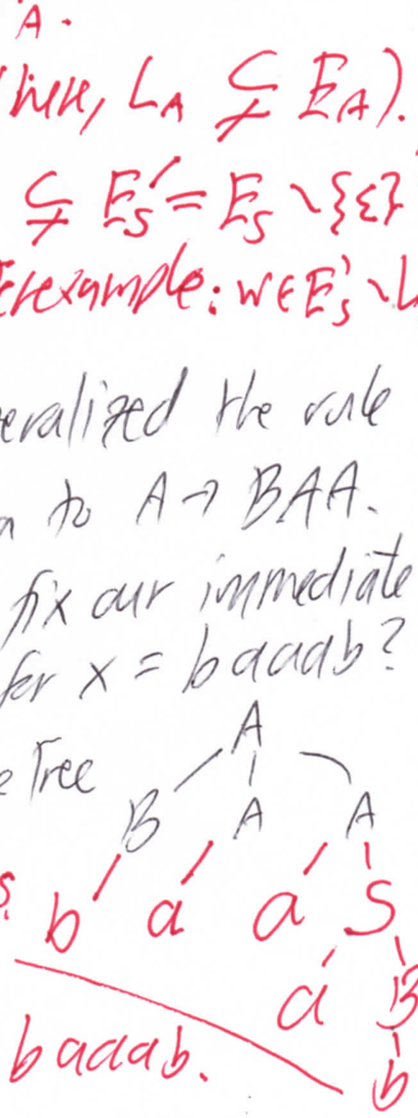
$\Rightarrow bbaaa\underline{B} \Rightarrow bbaaab = w.$

We get $\overset{S}{b\ \ A}$ on top of this to get a parse tree for w from S.

We liberalized the rule $A \to bAa$ to $A \to BAA$.

Does this fix our immediate problem for $x = baaab$?

Yes: Parse Tree



Thus we fixed the problem cases $x = baaab$, $w = bx$.
Did we fix the entire grammar to make $L_S = E_S$?
(And $L_A = E_A$, and $L_B = E_B$)? Proved via parsing plus induction/recursion

Consider any $x \in E_A$. We need to show that $A \overset{*}{\underset{G_2}{\Rightarrow}} x$

Cases: ① x begins with a. Then either $x = a$, where $A \Rightarrow a$, or $x = ay$ where $y \neq \varepsilon$ and $\#a(y) = \#b(y)$, ie $y \in E_S$.
$x = ay$, $A \Rightarrow$, S
By induction on $|y|$, $S \overset{*}{\Rightarrow} y$, so $A \Rightarrow aS \overset{*}{\Rightarrow} ay = x$.

② x begins with b. Then $x = bz$ where z has $\underline{\underline{2}}$ more a's than b's. Hence z can be broken as $z = uv$ where u and v each have $\underline{1}$ more a than b.
By self-recursion, we get $A \overset{*}{\Rightarrow} u$ and $A \overset{*}{\Rightarrow} v$.
Finally we get $A \Rightarrow BAA \Rightarrow bAA \overset{*}{\Rightarrow} buA \overset{*}{\Rightarrow} buv = x$

<u>That's the idea:</u> The full proof needs handling B too and S and base cases, for over <u>triple the work.</u>

(Skipped this year)

But the T_A, T_B, T_S kind of proof will be used. of sandwich