Lecture Tue Mar 31

\[ \text{Diff}(X, i) = \# \{ \text{chars} (X_i, X) | 0 \leq i \leq n \} \]

**Bal** = \{ \( x \in \{0, 1\}^* \) : \( \text{Diff}(x, i) = 0 \) \( \forall i : 0 \leq i < n \) \}(x, i) \geq 0 \}

G1: \( S \rightarrow \varepsilon \mid SS \mid (S) \)

**Prove** \( L(G_1) \subseteq \text{Bal} \Rightarrow L(G_1) = \text{Bal} \) (both grammars have \( S \rightarrow \varepsilon \))

Note: \( G_1 \) simulates \( G_2 \) rule-wise via \( S \Rightarrow SS \Rightarrow (S)S \).

Hence: soundness of \( G_1 \) \( \Rightarrow \) soundness of \( G_2 \). i.e. \( L(G_1) \subseteq \text{Bal} \Rightarrow L(G_2) \subseteq \text{Bal} \).

But, \( G_2 \) does not simulate \( G_1 \) rule-wise: in \( G_2 \) you cannot get the so-called "sentential form" \( SS \). (So the "expanded" languages of the two grammars are not the same, but we don't care)

Hence also: If \( G_2 \) is comprehensive, then \( G_1 \) is comprehensive. i.e.

\[ L(G_2) \supseteq \text{Bal} \Rightarrow L(G_1) \supseteq \text{Bal} \]

Earlier we proved \( G_1 \) is sound, so we only need \( L(G_2) \supseteq \text{Bal} \).

Proof by induction on the lengths \( n \) of strings:

Prove \( \forall n \ P(n) \), where \( P(n) \equiv \text{for each } x \in \{0, 1\}^n \): \( S \Rightarrow x \).

\( G = S \rightarrow \varepsilon \mid (S)S \)

**Base (n=0)** \( \exists x \in \text{Bal} : S \Rightarrow \varepsilon \) so that checks.

Another **Base (n=1)**: The only strings \( x \in \{0, 1\}^2 \) are \( x = ("0" \text{ and } x = ") \).

Neither string belongs to \( \text{Bal} \). So this holds by default — nothing to do.

**Induction (n \geq 2)**: Assume \( P(n) \) the statement \( \forall m < n \) \( P(m) \).

Given \( n \), goal: show \( P(n) \). Let \( x \in \{0, 1\}^n \) be given.

1. If \( x \in \text{Bal} \) (as always happens when \( n \) is odd), there is **nothing to do**.
2. Let \( x \notin \text{Bal} \). Our goal is to show \( S \Rightarrow x \) in \( G_2 \). Consider how to replace \( X \).

We know \( \text{Diff}(X, i) = 0 \) when \( i = 0 \) and \( i = n \) and \( \text{Diff}(X, i) \geq 0 \) for all \( i < n \).

We want to isolate a "critical" value \( i \). \( x = (\{00\}, \{11\}) \) so that \( S \Rightarrow (S)S \Rightarrow x \).

\( S \Rightarrow (S)S \Rightarrow x \) won't work for \( x \): the parse \( S \Rightarrow (S)S \Rightarrow x \) leaves \( L \cap \text{Bal} \).
Take \( i \) to be the least \( i > 0 \) such that \( \text{Diff}(X, i) = 0 \).

By \( \text{Diff}(X, n) = 0 \) we know that the loop while \( (\text{Diff}(X, i), i > 0) \) \( i \) \( \neq 0 \) must exit. [Side remark: When we don't have such a bound, this kind of]

"Impredicative" math definition is highly controversial.

Clearer rewrite: Take \( i \) to be the least such that \( \text{Diff}(X, i) = 0 \) \( (i > 0) \).

Let \( X = \left( \begin{array}{c} - \cdots - \end{array} \right) \)

Then we can parse \( X = (Y_{i-1})^2 \) where \( |Y_{i-1}| = i-2 \) and:

\( \text{Diff}(Y, |Y|) = \text{Diff}(X, i-1) - 1 = 1 - i = 0 \)

For each \( j, 0 \leq j \leq i-2 \), \( \text{Diff}(Y, j) = \text{Diff}(X, j+1) - 1 \).

Because \( i \) is least such that \( \text{Diff}(X, i) = 0 \) \( (i > 0) \), we have that for all \( j, j+1 \) runs from \( 0 \) to \( i-1 \) as indices of \( X \), so that

\( \text{Diff}(X, j+1) \geq 1 \rightarrow A_j, 0 \leq j \leq |Y| - \text{Diff}(Y, j) \geq 0. \)

\( \text{Diff}(Y, |Y|) = 0 \) and \( A_j, 0 \leq j \leq |Y|, \text{Diff}(Y, j) \geq 0 \) \( \rightarrow Y \in \text{Bal} \).

Moreover, for all \( K, i \leq K \leq n \), \( \text{Diff}(X, K) = \text{Diff}(Z, K-i) \).

\( \rightarrow \text{Diff}(Z, i) = \text{Diff}(X, n) = 0 \), and for all \( l, 0 \leq l \leq 2, \text{Diff}(Z, l) = 0 \).

\( \rightarrow Y \in \text{Bal} \) and \( Z \in \text{Bal} \). And \( |Y| = i-2 \leq n \) (even when \( i = n \), \( Z = n-i < n \).

Hence we can apply \( \text{Th} P[i-2] \) and \( P(n-i) \) to get since \( i > 0 \)

\( S \models Y \) and \( S \models Z \) in \( \mathcal{E}_2 \). Thus \( S \models (S \models Y) \models Y \models Y \models (Y) \models X \). done
Q: Do we need \( S \rightarrow \varepsilon \)?

G2: \( S \rightarrow \varepsilon \mid (S)S \). \( L(G_2) = \{a^i b^j \} \).

Yes since \( \varepsilon \in \{a^i b^j \} \), but can we get all of \( \{a^i b^j \} \) without \( \varepsilon \)-rules?

How about \( G' = S \rightarrow (S)S \). \( G' \) is sound, but is it comprehensive for \( \{a^i b^j \} \)?

Try \( X = (()) \). \( S \rightarrow (S)S \Rightarrow ((()))S \) as leaves on \( S \)

Theorem: [First part of conversion to Chomsky Normal Form]

For any CFG \( G = (V, \Sigma, R, S) \), we can build a CFG \( G' \) without \( \varepsilon \)-rules such that \( L(G') = L(G) \setminus \{\varepsilon \}. \)

[Optionally, as the text does, it \( \varepsilon \in L(G) \) we can finally add a new start symbol \( S' \) with rules \( S' \rightarrow \varepsilon \mid S \).

Proof:

1. Identify the subset \( W \subseteq V \) of nullable variables, meaning \( A \in V \) such that \( A \Rightarrow^* \varepsilon \). In \( G_2 \): \( W = V = \{S\} \).

2. For every rule \( B \Rightarrow X \) in \( G \), add to \( R \) all rules obtained by deleting 1 or more nullable variables. [As literally described, this can be an exponential explosion]

\[ S \rightarrow \varepsilon \mid (S)S \]  \text{and} \[ S \rightarrow (S) \]  \text{and} \[ S \rightarrow () \] deleting both occurrences.

Finally, define \( G' \) by deleting all \( \varepsilon \)-rules.

Hence, show that \( L(G') \subseteq L(G) \setminus \{\varepsilon \} \).

We have shown that all non-empty strings \( x \in L(G) \) can be derived in \( G' \).

This is sound because all added rules could be simulated by deriving the deleted occurrences to \( \varepsilon \).

Consider any parse tree for \( x \) in the original grammar \( G \). Just delete all subtrees that yielded \( \varepsilon \). \( G' \) covers all such deletions with \( \varepsilon \)-rules, so the new parse is legal.
Defn: A CFG $G$ is in Chomsky Normal Form (CNF) if all rules in $R$ have the form $A \rightarrow \varepsilon$ or $A \rightarrow BC$ with $A, B, C \in V$, $\varepsilon \in \Sigma$. Possibly $B, C = A$

Theorem: For every CFG $G$ we can build a CFG $G''$ in CNF such that $L(G'') = L(G) - \{\varepsilon\}$. (originally add $S'' \rightarrow \varepsilon$ to allow $L(G'') = L(G)$.)

Note: CNF involves the step of replacing every terminal $c \in \Sigma$ to a variable $X_c$ and adding the rules $X_c \rightarrow c$ as the only production with terminals on the RHS. $S \rightarrow X_L S X_R S | X_L X_R S | X_L S X_R | X_L X_R X_R \rightarrow c | X_L \rightarrow c$.

Extra: The rest of the algorithm:
1. Eliminate $\varepsilon$-rules as above (don't do the optional $S'' \rightarrow \varepsilon$ yet: save it for the end).
2. Now find all pairs of variables $(A, B)$ such that $A \Rightarrow \varepsilon B$. This might happen by transitivity: $A \rightarrow C$, $C \rightarrow D$, $D \rightarrow B$. Add all right-hand sides of $B$ as rule options for $A$. Finally delete all "unit productions" $A \rightarrow C$ etc. Call the result $G'$.
3. Finally, the "silly steps": Alias every $c \in \Sigma$ to $X_c$ as above, except in rules $A \rightarrow B$.
4. For every right-hand side of length $\geq 3$, make variables $Y_1, \ldots, Y_{k-1}$. If the rule is $A \rightarrow V_1 V_2 \cdots V_k$, the new rules are $A \rightarrow V_1 Y_1$, $Y_1 \rightarrow V_2 Y_2$, $Y_2 \rightarrow V_3 Y_3$, $\ldots$, $V_{k-1} \rightarrow V_k-1 V_k$. Use different "$Y$" variables for each rule. Call that $G''$. Optionally, add $S'' \rightarrow \varepsilon$ to all right-hand sides of $S$ in $G''$. Then $L(G'') = L(G)$ in CNF.

Example of step 4 for our parentheses grammar:

$s \rightarrow x_L y_{11} | x_L y_{21} | x_L y_{31} | x_L y_{41} \mid x_R x_2 \mid x_R \rightarrow c$, $x_L \rightarrow c'$

Then $G''$ is in Chomsky NF. ($\text{Can combine } Y_{11} \text{ and } Y_{21}$, but this is hideously unreadable anyway)

$y_{11} \rightarrow S y_{12}, y_{12} \rightarrow x_R S, y_{21} \rightarrow x_R S, y_{31} \rightarrow x_R S$. Option: $S'' \rightarrow \varepsilon | Y_{11} y_{11} | Y_{21} y_{21} | Y_{31} y_{31} | Y_{41} y_{41}$