CFG ⊆ EOL (first part of the proof CFG⊂EOL containing a transformation)

Proof:

① Transform a generic context-free grammar $G = (N, \Sigma, P_1, S)$ to EOL-system $H = (V, T, P_2, w)$

- approach: $V = N \cup \Sigma$
  $T = \Sigma$
  $P_2 = P_1 \cup \{a \to a \text{ for } \forall a \in \Sigma\}$
  $w = S$

② Proof that $L(G) = L(H)$

a) $L(H) \subseteq L(G)$, thus $\forall u \in L(H) \Rightarrow u \in L(G)$

proof by a mathematical induction (on derivation sequence) that for every $\{\omega \mid S \overset{H}{\Rightarrow}{}^{*} \omega\}$ there exists such a derivation in G so that $S \overset{G}{\Rightarrow}{}^{*} \omega$

idea:
I) we start with S in both G, H
II) we assume that ω can be derived from S in both G, H
III) we show that $\omega \overset{H}{\Rightarrow} \omega'$ can be simulated in G $\omega \overset{G}{\Rightarrow}{}^{*} \omega'$

b) $L(G) \subseteq L(H)$, thus $\forall u \in L(G) \Rightarrow u \in L(H)$

this is simple induction if we make $P_2 = P_1 \cup \{x \to x \text{ for } \forall x \in N \cup \Sigma\}$

not so easy in our case, idea:
- if we want to get $u \in \Sigma^*$ in G we need to replace all non-terminals in $\omega \in (N \cup \Sigma)^*$ using production rules P_1
- there is nothing preventing us to do this in parallel (this can be illustrated with parsing (derivation) trees where every non-terminal is a root of an independent sub-tree)

In other words, not all $\{\omega \in (N \cup \Sigma)^* \mid S \overset{G}{\Rightarrow}{}^{*} \omega\}$ ("sentence forms" derivable from S in G) can be derived in H BUT all $\{u \in \Sigma^* \mid S \overset{G}{\Rightarrow} u\}$ can be also derived in H

mathematical induction is not applicable, place for a direct proof?

Conclusion: using an alternate transformation where $P_2 = P_1 \cup \{x \to x \text{ for } \forall x \in N \cup \Sigma\}$ we can easily prove that CFG ⊆ EOL (and using example EOL language that CFG ⊂ EOL)

that does not mean such transformation is the simplest or the only possible

„The other way around" a.k.a. why we cannot get into the problem we want to

Known „problem" (property) of parallel $0L$-systems: production rule $a \to aa$ generates a^{2^n} and <u>not</u> a^*

So lets „recreate" this trick with a transformed EOL grammar.

- production rules $a \to aa$, $a \in \Sigma$ are not allowed in CFG
- even with alternative CFG definition, this is not a problem since $a \to a$ rules were added as a part of transformation

Well, use non-terminals: $A \to AA$

„catch" - if we want to end up with a terminal string, we also have an „escape rule" $A \to \alpha$ $\alpha \in (N \cup \Sigma)^*$ that we can use „in parallel" with the $A \to AA$ rule
- moreover, the derivation $A \Rightarrow^+ w$, $w \in \Sigma^*$ must be possible
- „inequal ending time" = inequal parsing (derivation) subtree depth is already covered by „$a \to a$" rules

The case that would break this „parallelism" eventually: $A \to AA$
$AA \to aa$

Althought language generated by such grammar $F = (\{A\}, \{a\}, P, A)$ is $(aa)^+$, thus context-free (even regular), such grammar is <u>not</u> context-free anymore.