CONTEXT-FREE GRAMMAR and PARSING TABLE

Problem: Given a grammar, how do we construct a parsing table?

Context-free grammar:

$G = (N, T, P, S)$ where:

- N is finite set of non-terminals.
- T is finite set of terminals.
- P is finite set of productions in the form of $A \rightarrow \alpha$
  - where:
    - $A \in N$
    - $\alpha \in (N \cup T)^*$ is a sentential form
- $S \in N$ is the start symbol

Sentential form:

String $\in N \cup T$ that can be derived from S

Context-free grammar vs Regular grammar

Context-free

- Exactly 1 non-terminal on LHS
- RHS can be a mix of terminals and non-terminals

Regular

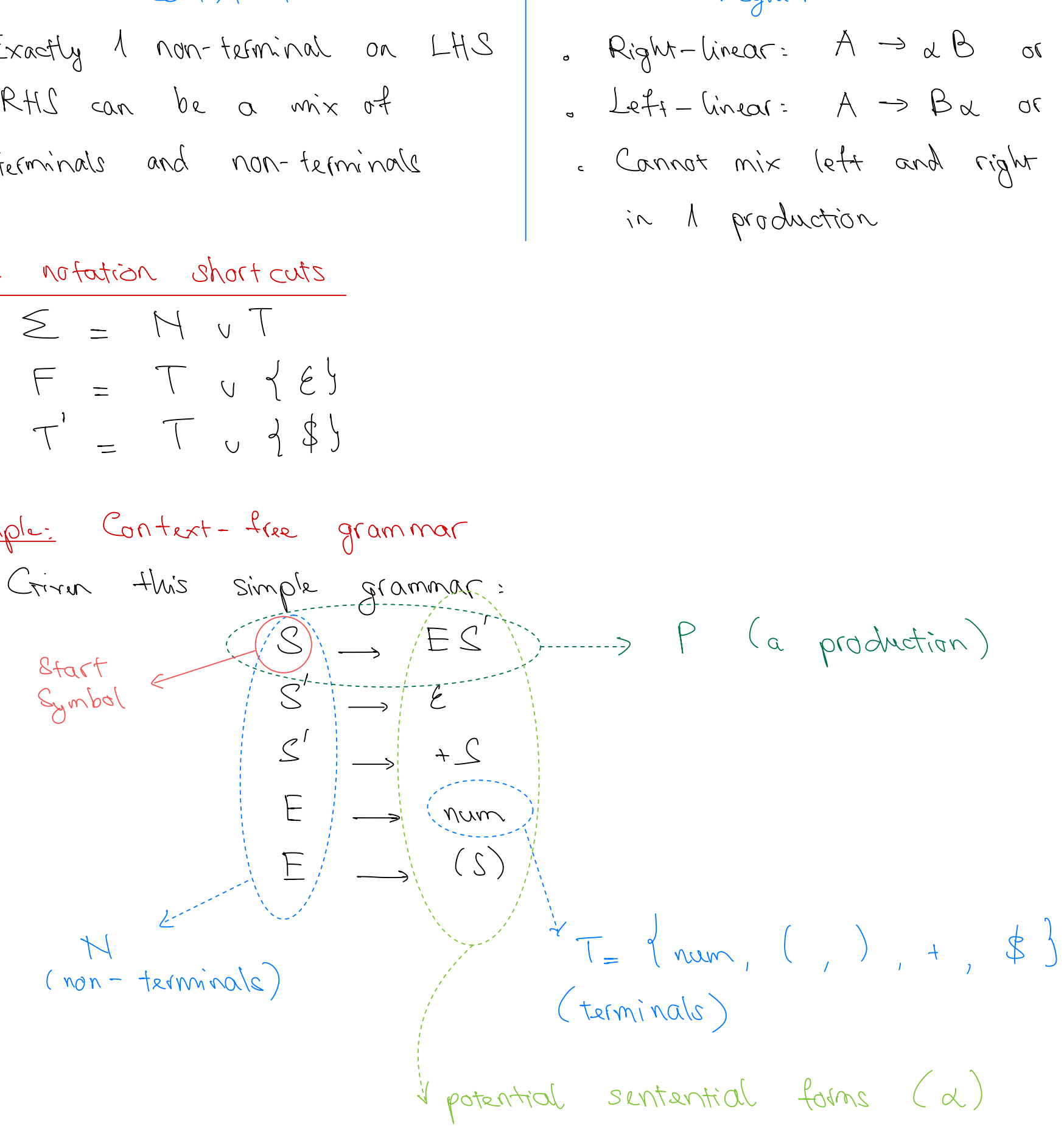
- Right-linear: $A \rightarrow \alpha B$ or $A \rightarrow \alpha$
- Left-linear: $A \rightarrow B\alpha$ or $A \rightarrow \alpha$
- Cannot mix left and right linear in 1 production

Some notation shortcuts

- $\Sigma = N \cup T$
- $F = T \cup \{\epsilon\}$
- $T' = T \cup \{\$\}$

Example: Context-free grammar

Given this simple grammar:



Construct Parsing Table from Grammar

$S \rightarrow ES'$
$S' \rightarrow \varepsilon$
$S' \rightarrow +S$
$E \rightarrow num$
$E \rightarrow (S)$

?

	num	+	(	)	$
S	→ ES'		→ ES'		
S'		→ +S		→ ε	→ ε
E	→ num		→ (S)		

A

t

Problem: Given a terminal t, what production $A \rightarrow \alpha$, should be filled in the parsing table cell $PT[A, t]$

→ **Intuition:** a production $A \rightarrow \alpha$ is a candidate if ang of these true

1. $t \in FIRST(\alpha)$?

Does this terminal t the first thing we see using this production $A \rightarrow \alpha$, where α is the RHS ?

Example: Given $t =$ "(", what production goes in $PT[S, (]$?

Consider production $S \rightarrow ES'$

$FIRST(S)$
$= FIRST(E)$
$=$ "(" → Valid production

2. $NULLABLE(\alpha)$ and $t \in FOLLOW(A)$?

If current RHS α can derive null (ε), then does the terminal t following the prodution A ?

Example: Given $t =$ ")", what production goes in $PT[S',)]$?

Consider production $S' \rightarrow \varepsilon$ (A: S', α: ε)

$FOLLOW(S')$ $NULLABLE(\varepsilon)$
$= FOLLOW(S)$ ✓
$=$ ")"
✓ ⇒ Valid production

Construct Parsing Table Algorithm