Context Free Languages

Last Updated: February 8th, 2023

1 Introduction

Context free languages are foundational for defining several types of computing languages that occur in practice; including programming languages, markup languages, and languages for communication protocols. CFL's were first studied in relation to natural-language processing in the 1950's.

Their importance stems from the following.

1. CFL's are significantly more expressive than regular languages in that they are capable of defining recursive languages that may have unlimited recursive depth.
2. a CFL can be recognized by a pushdown automaton (PDA). Unlike DFA's a PDA has unlimited memory, albeit in the form of a stack whose access is limited to i) reading the top of the stack, ii) pushing on to the stack, and iii) popping the top of the stack. PDA's are also interesting because their nondeterministic counterparts (NPDA's) are more powerful than PDA's, and the set of languages accepted by an NPDA is equal to the set of CFL languages.

Although every regular language is also a CFL (see the exercises), the converse is not true. For example, it can be proved that the language

$$L = \{a^n b^n | n \geq 0\}$$

is a CFL but is *not* regular. Intuitively, L is not regular because a DFA M would have to keep track of the number a's read and them make sure that the same number of b's are subsequently read. However, the number of a's read can grow without bound and exceed M's finite and bounded memory capacity. Example 7 below shows that L is a CFL.

2 Context-Free Grammars

A **Context-Free Grammar (CFG)** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set of variables
2. Σ is a finite set that is disjoint from V, called the **terminal set**
3. R is a finite set of rules where each **rule** has the form

   $$A \to s,$$

   where $A \in V$ and $s \in (V \cup \Sigma)^*$. Variable A is referred to as the **head** of the rule, while s is referred to its **body**.
4. $S \in V$ is the **start variable**

Example 1. Consider the set of rules

Legal: (( ))( )

$$R = \{S \to SS, S \to aSb, S \to \varepsilon\}.$$

Not Legal:

)( )(

Then we may use this set of rules to define a CFG $G = (V, \Sigma, R, S)$, where

$$V = \{S\},$$

$$\Sigma = \{a, b\},$$

and variable S is the start variable.

For brevity we may list together rules having the same head as follows.

$$S \to SS \mid aSb \mid \varepsilon.$$

Here, each of the rule bodies is separated by a vertical bar. $\square$

Example 2. One common use of CFG's is to provide grammatical formalism for natural languages. For example, consider the set of rules R:

$$\langle \text{SENTENCE} \rangle \to \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle$$

$$\langle \text{NOUN-PHRASE} \rangle \to \langle \text{COMPLEX-NOUN} \rangle \mid \langle \text{COMPLEX-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle$$
$$\langle \text{VERB-PHRASE} \rangle \to \langle \text{COMPLEX-VERB} \rangle \mid \langle \text{COMPLEX-VERB} \rangle \langle \text{PREP-PHRASE} \rangle$$
$$\langle \text{PREP-PHRASE} \rangle \to \langle \text{PREP} \rangle \langle \text{COMPLEX-NOUN} \rangle$$
$$\langle \text{COMPLEX-NOUN} \rangle \to \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle$$
$$\langle \text{COMPLEX-VERB} \rangle \to \langle \text{VERB} \rangle | \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle$$
$$\langle \text{ARTICLE} \rangle \to \text{a} \mid \text{the}$$
$$\langle \text{NOUN} \rangle \to \text{trainer} \mid \text{dog} \mid \text{whistle}$$
$$\langle \text{VERB} \rangle \to \text{calls} \mid \text{pets} \mid \text{sees}$$
$$\langle \text{PREP} \rangle \to \text{with} \mid \text{in}$$

Here, the variables are the ten parts of speech delimited by $\langle\ \rangle$, Σ is the lowercase English alphabet, including the space character, and $\langle \text{SENTENCE} \rangle$ is the start variable.

Example 3. A CFG may also be used to define the syntax of a programming language. One fundamental language component to any programming language is that of an *expression.* The following rules imply a CFG for defining expressions formed by a single terminal `a`, parentheses, and the two arithmetic operations $+$ and $\times$. Here E stands for `expression`, T for `term`, and F for `factor`.

$$E \to E + T \mid T$$

$$T \to T \times F \mid F$$

$$F \to (E) \mid a$$

We have $V = \{E, T, F\}$, $\Sigma = \{+, \times, a, (,)\}$, and E is the start variable.

2.1 Grammar derivations

Let $G = (V, \Sigma, R, S)$ be a CFG, then the language $D(G) \in (V \cup \Sigma)^*$ of **derived words** is structurally defined as follows.

Atom $S \in D(G)$.

Compound Rule Suppose $s \in D(G)$, s is of the form uAv for some $u, v \in (V \cup \Sigma)^*$, $A \in V$, and $A \to \gamma$ is a rule of G, then

$$u\gamma v \in D(G).$$

In this case we write $s \Rightarrow u\gamma v$, and say that s **yields** $u\gamma v$. In words, to get a new derived word, take an existing derived word and replace one of its variables A with the body of a rule whose head is A.

The subset $L(G)$ of derived words $w \in D(G)$ for which $w \in \Sigma^*$ is called the **context-free language (CFL)** associated with G. Thus, the words of $L(G)$ consist only of terminal symbols.

$L(G) \subseteq D(G)$

words whose characters are all terminals

2.2 The Derivation relation

Let u and v be words in $(V \cup \Sigma)^*$. We say that u **derives** v, written $u \overset{*}{\Rightarrow} v$ if and only if either $u = v$ or there is a sequence of words $w_1, w_2, \ldots, w_n$ such that

$$u = w_1 \Rightarrow w_2 \Rightarrow w_3 \Rightarrow \cdots \Rightarrow w_n = v.$$

Such a sequence is called a **derivation sequence** from u to v.

$$L(G) = \{w \in \Sigma^* | S \overset{*}{\Rightarrow} w\}.$$

$w_1 \Rightarrow w_2$

"w_1 yields w_2"

Example 4. Use the CFG from Example 1 to derives the word `aabbaababb`.

$$S \to SS \mid aSb \mid \varepsilon.$$

Solution.

$S \Rightarrow SS \Rightarrow aSbS \Rightarrow aaSbbS$

$\Rightarrow aabbS \Rightarrow aabbaSb \Rightarrow$

$aabbaSSb \Rightarrow$

$aabbaaSbSb \Rightarrow$

$aabbaabSb \Rightarrow aabbaabaSbb$

$\Rightarrow \boxed{aabbaababb}$

2.3 Derivation parse trees

Determining if an arbitrary word belongs to $L(G)$ is of fundamental importance. But in addition, it is sometimes important to know the structure of the grammar's derivation of the word. For example, if a CFG generates arithmetic expressions, then knowing the structure of the derivation allows one to readily evaluate the expression (assuming the expression terminals have assigned values and the expression operations are properly defined). A **parse tree** for a word $w \in L(G)$ is a tree whose structure and node labels reflect the derivation w, where, from left to right, the leaves of the tree are labeled with the letters of w. Indeed, consider the derivation sequence

$$S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n = w.$$

Then the parse tree for w can be defined in a step-by-step manner. To begin the parse tree T_1 for $S = w_1$ consists of a single node labeled with S.

Now suppose a parse tree T_k has been associated with w_k, the k th word of the derivation. Moreover, assume that, from left to right, the leaves of T_k are labeled in one-to-one correspondence with the symbols of w_k. Moreover, assume that w_k has the form $w_k = uAv$, where A is substituted for a word γ, so that $w_{k+1} = u\gamma v$. Then T_{k+1} is obtained from T_k by assigning the leaf node labeled with A a number of children equal to the length of γ and for which, from left to right, the i th child is labeled with the ith symbol of γ.

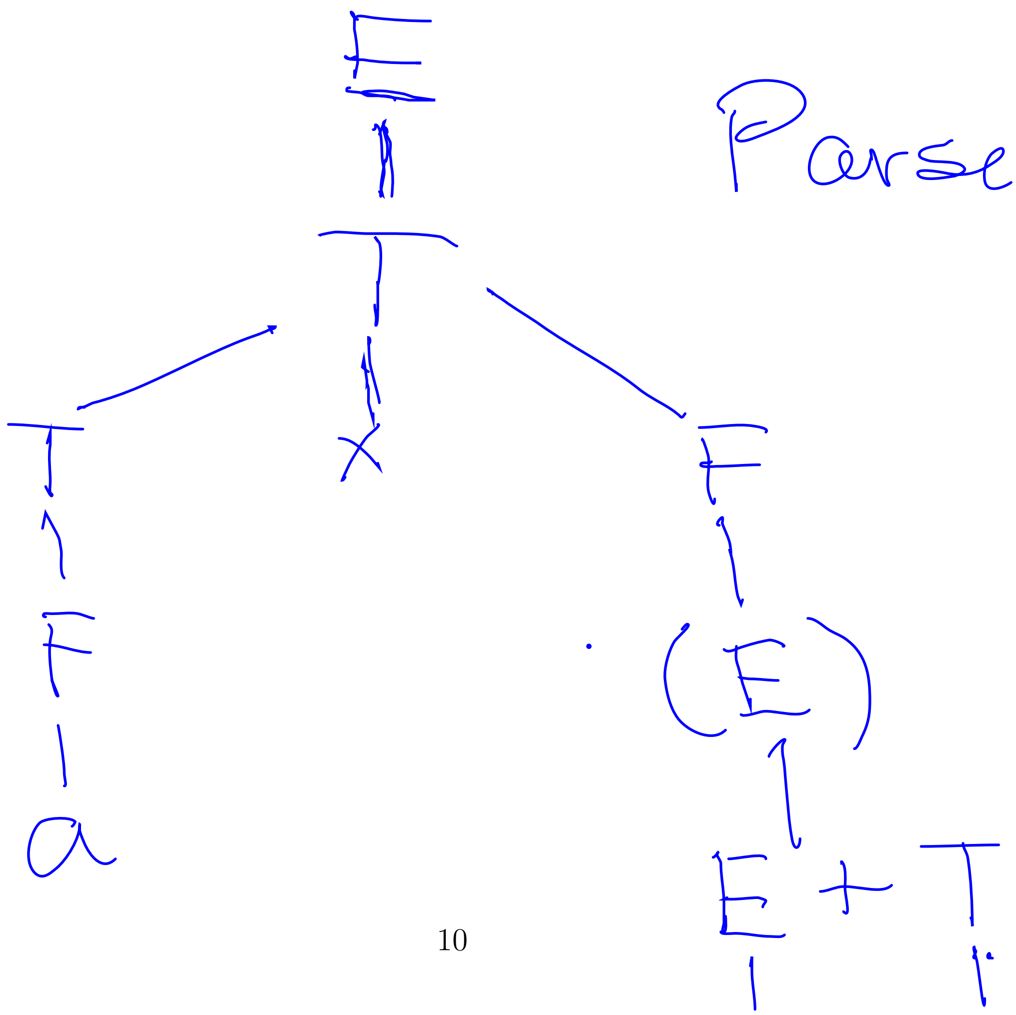
Example 5. Use the CFG from Example 3 to derive the expression $a \times (a + a)$, and provide the parse tree associated with the derivation.

$$E \to E + T \mid T$$

$$T \to T \times F \mid F$$

$$F \to (E) \mid a$$

Solution. $E \Rightarrow T \Rightarrow T \times F \Rightarrow F \times F \Rightarrow a \times F \Rightarrow a \times (E) \Rightarrow a \times (E+T) \Rightarrow a \times (T+T) \Rightarrow a \times (F+T) \Rightarrow a \times (a+T) \Rightarrow a \times (a+F) \Rightarrow \boxed{a \times (a+a)}$



Parse Tree

2.4 Ambiguity

Given a CFG G, and a word $w \in L(G)$, there may be several different derivations of w from start symbol S. Many of these derivations however will yield identical parse trees. But in the event that two different derivation sequences of w from S yield two different parse trees, then we call G **ambiguous**. It turns out that an easy way to check for ambiguity is to check that no word w has more than one *leftmost derivation*.

Given grammar $G = (V, \Sigma, R, S)$ and word $w \in L(G)$, a derivation sequence $S = w_0 \Rightarrow w_1 \Rightarrow \cdots \Rightarrow w_{n-1} \Rightarrow w_n = w$ is called a **leftmost derivation** of w provided that, for every $0 \le i \le n-1$, the yielding of w_i from w_{i-1} was obtained by replacing the leftmost variable A of w_{i-1} with the body of one of a rule whose head is A. Therefore, if w has more than one leftmost derivation, it must be the case that a different sequence of rules were used to derive w. When this happens we call G ambiguous, since some words in the grammar have more than one parsing structure.

Example 6. Show that the grammar defined by the following rules is ambiguous.

$$\langle \text{SENTENCE}\rangle \to \langle \text{NOUN-PHRASE}\rangle\langle \text{VERB-PHRASE}\rangle$$
$$\langle \text{NOUN-PHRASE}\rangle \to \langle \text{COMPLEX-NOUN}\rangle \mid \langle \text{COMPLEX-NOUN}\rangle\langle \text{PREP-PHRASE}\rangle$$
$$\langle \text{VERB-PHRASE}\rangle \to \langle \text{COMPLEX-VERB}\rangle \mid \langle \text{COMPLEX-VERB}\rangle\langle \text{PREP-PHRASE}\rangle$$
$$\langle \text{PREP-PHRASE}\rangle \to \langle \text{PREP}\rangle\langle \text{COMPLEX-NOUN}\rangle$$
$$\langle \text{COMPLEX-NOUN}\rangle \to \langle \text{ARTICLE}\rangle\langle \text{NOUN}\rangle$$
$$\langle \text{COMPLEX-VERB}\rangle \to \langle \text{VERB}\rangle \mid \langle \text{VERB}\rangle\langle \text{NOUN-PHRASE}\rangle$$
$$\langle \text{ARTICLE}\rangle \to \text{a} \mid \text{the}$$
$$\langle \text{NOUN}\rangle \to \text{trainer} \mid \text{dog} \mid \text{whistle}$$
$$\langle \text{VERB}\rangle \to \text{calls} \mid \text{pets} \mid \text{sees}$$
$$\langle \text{PREP}\rangle \to \text{with} \mid \text{in}$$

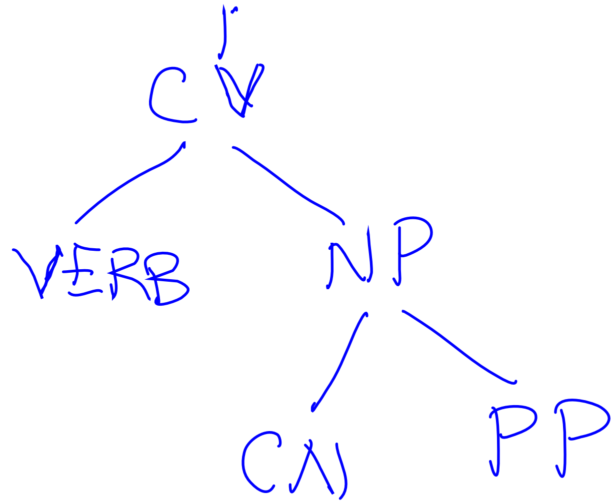
The trainer calls the dog with the whistle.

Solution. $\langle \text{Sentence}\rangle \Rightarrow \langle NP\rangle\langle VP\rangle \Rightarrow \langle CN\rangle\langle VP\rangle \Rightarrow \langle ART\rangle\langle NOUN\rangle\langle VP\rangle \overset{*}{\Rightarrow} \text{the trainer } \langle VP\rangle \Rightarrow \text{the trainer } \langle CV\rangle\langle PP\rangle \Rightarrow \text{the trainer } \langle V\rangle\langle NP\rangle\langle PP\rangle \Rightarrow \text{the trainer calls } \langle NP\rangle\langle PP\rangle \Rightarrow \text{the trainer calls } \langle CN\rangle\langle PP\rangle \Rightarrow \text{the trainer calls } \langle ART\rangle\langle NOUN\rangle\langle PP\rangle \overset{*}{\Rightarrow} \text{the trainer calls the dog } \langle PP\rangle \Rightarrow \text{the trainer calls the dog } \langle PREP\rangle\langle CN\rangle \Rightarrow \text{the trainer calls the dog with } \langle ART\rangle\langle NOUN\rangle \overset{*}{\Rightarrow} \text{the trainer calls the dog with the whistle.}$

Second derivation

$\underbrace{\text{The trainer}}_{NP}\ \underbrace{\text{calls } \boxed{\text{the dog with the whistle}}}_{VP}$

Solution Continued.



Exercise: derive the second interpretation

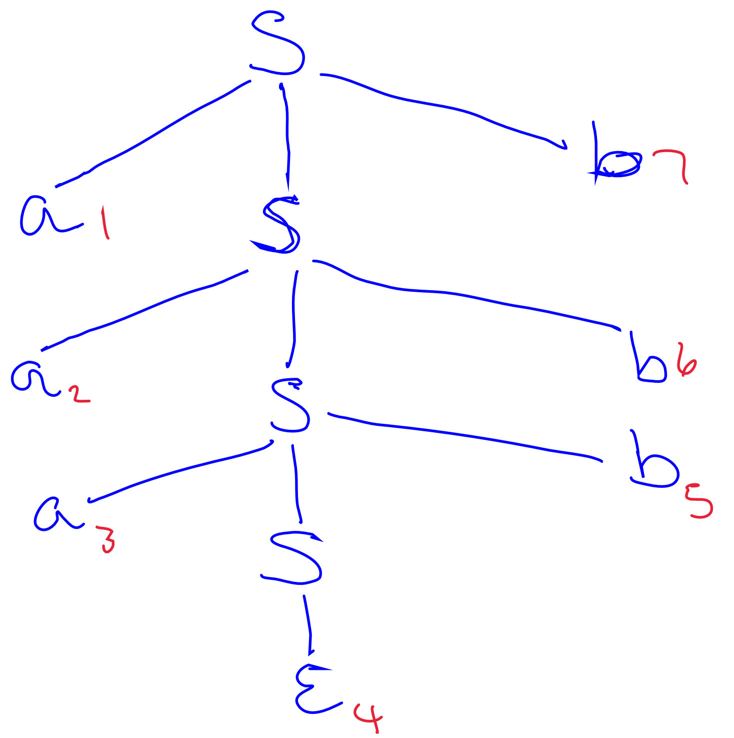
Example 7. Provide a CFG G for which

$$L(G) = \{a^n b^n | n \geq 0\}.$$

Provide a derivation of $a^3 b^3$ and draw its parse tree.

$$S \rightarrow aSb \mid \varepsilon$$

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb$$
$$\Rightarrow aaabbb = a^3 b^3$$



Example 8. Provide a CFG G for which

$$L(G) = \{x^i y^j z^k | i, j, k \geq 0 \text{ and } j = 2i \text{ or } k = 2j\}.$$

$$S \rightarrow R_{xy} \mid R_{yz}$$

$$R_{xy} \rightarrow R'_{xy} Z$$

$$R'_{xy} \rightarrow x R'_{xy} yy \mid \varepsilon$$

$$Z \rightarrow zZ \mid \varepsilon$$

$$R_{yz} \rightarrow X R'_{yz}$$

$$R'_{yz} \rightarrow y R'_{yz} zz$$

$$R'_{yz} \rightarrow \varepsilon$$

$$X \rightarrow xX \mid \varepsilon$$

xy^3z^6

Example 9. Use the CFG from the previous example to provide a derivation of [redacted] and draw its parse tree.

$$S \to R_{yz} \Longrightarrow X R'_{yz} \Longrightarrow x X R'_{yz} \Longrightarrow$$

$$x R'_{yz} \Longrightarrow x y R'_{yz} zz \Longrightarrow x yy R'_{yz} zzzz$$

$$\Longrightarrow x yyy R'_{yz} zzzzzz \Longrightarrow x yyy zzzzzz$$

$$= \boxed{x y^3 z^6}$$

Exercises

1. For the CFG defined in Example 1, provide a derivation for the following words.

   a. ababab

   b. aaababbbab

   c. aababaabbbaabb

2. For the CFG defined in Example 3, provide a derivation and parse tree for the following expressions.

   a. a

   b. $a + a$

   c. $a \times (a \times a)$

   d. $((a))$

Exercise Solutions

1. We have the following derivations.

   a. ababab

$$\text{S} \Rightarrow \text{SS} \Rightarrow \text{aSbS} \Rightarrow \text{abS} \Rightarrow \text{abSS} \Rightarrow \text{abaSbS} \Rightarrow \text{ababS} \Rightarrow \text{ababaSb} \Rightarrow \text{ababab}.$$

   b. aaababbbab

$$\text{S} \Rightarrow \text{SS} \Rightarrow \text{aSbS} \Rightarrow \text{aaSbbS} \Rightarrow \text{aaSSbbS} \Rightarrow \text{aaaSbSbbS} \Rightarrow \text{aaabSbbS} \Rightarrow \text{aaabaSbbbS}.$$

$$\Rightarrow \text{aaababbbS} \Rightarrow \text{aaababbbaSb} \Rightarrow \text{aaababbbab}.$$

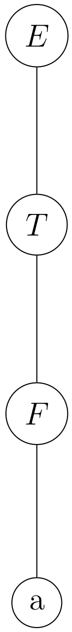
   c. aababaabbbaabb

$$\text{S} \Rightarrow \text{SS} \Rightarrow \text{aSbS} \Rightarrow \text{aSSbS} \Rightarrow \text{aaSbSbS} \Rightarrow \text{aabSbS} \Rightarrow \text{aabSSbS}$$
$$\Rightarrow \text{aabaSbSbS} \Rightarrow \text{aababSbS} \Rightarrow \text{aababaSbbS} \Rightarrow \text{aababaaSbbbS} \Rightarrow \text{aababaabbbS}$$
$$\Rightarrow \text{aababaabbbaSb} \Rightarrow \text{aababaabbbaaSbb} \Rightarrow \text{aababaabbbaabb}.$$

2. For the CFG defined in Example 3, provide a derivation and parse tree for the following expressions.

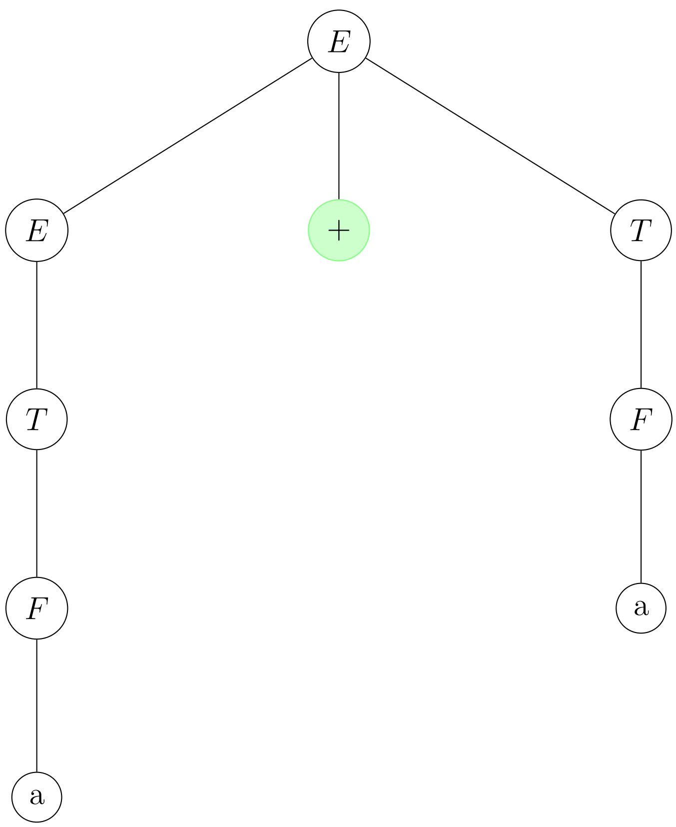
   a. a

$$E \Rightarrow T \Rightarrow F \Rightarrow \text{a}.$$



b. $a + a$

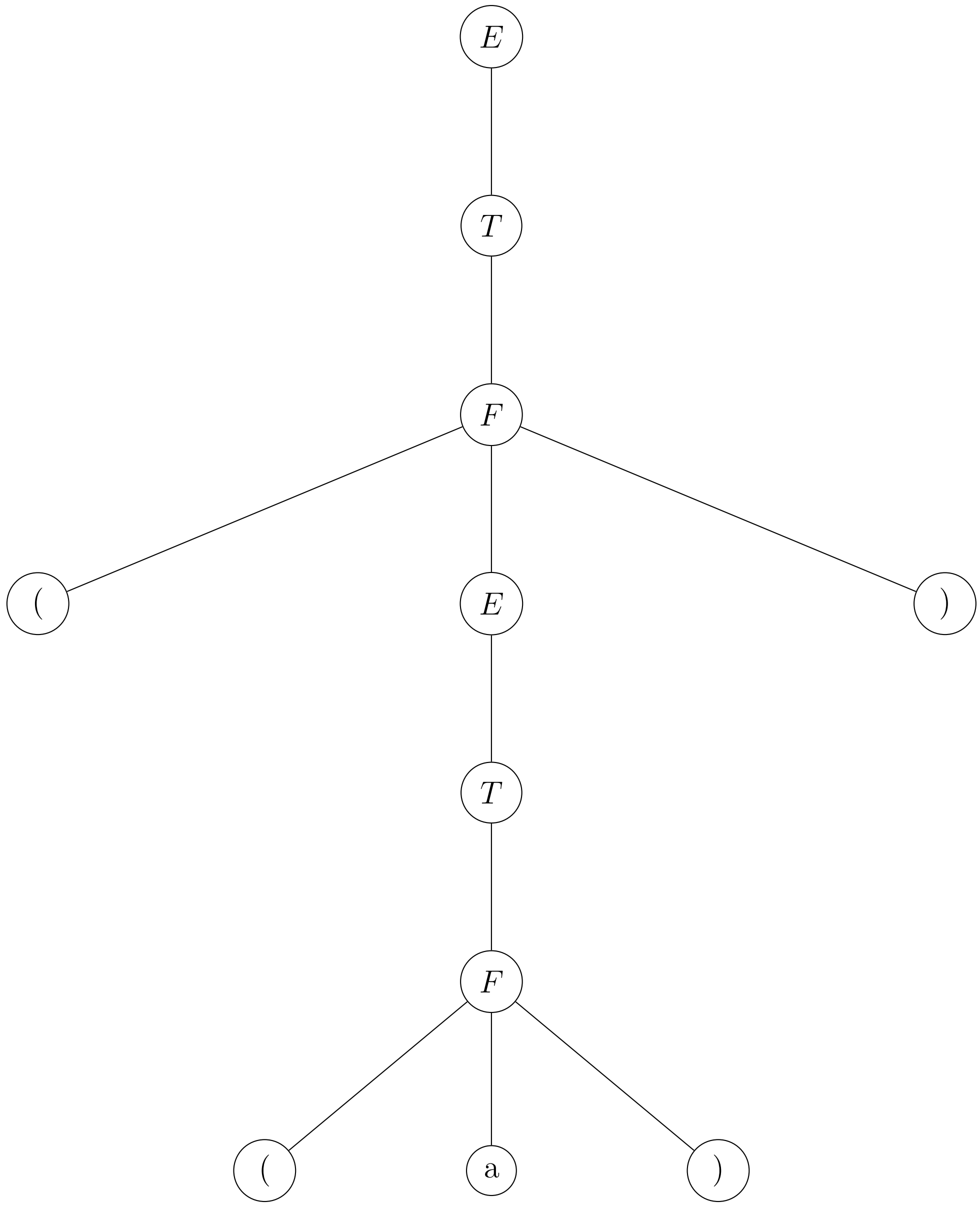
$$E \Rightarrow E + T \Rightarrow T + T \Rightarrow F + T \Rightarrow \mathrm{a} + T \Rightarrow \mathrm{a} + F \Rightarrow \mathrm{a} + \mathrm{a}.$$



c. $a \times (a \times a)$

d. $((a))$

$$E \Rightarrow T \Rightarrow F \Rightarrow (E) \Rightarrow (T) \Rightarrow (F) \Rightarrow ((E)) \Rightarrow T \Rightarrow F \Rightarrow ((\mathrm{a})).$$


E
T
F
(
E
)
T
F
(
a
)