

Theory of Computation 7

Normalforms and Algorithms

Frank Stephan

Department of Computer Science

Department of Mathematics

National University of Singapore

fstephan@comp.nus.edu.sg

Repetition 1

Example

$\text{Ascii}(\text{Year 2019}) = 596561722032303139$ represents each letter of “Year 2019” by its two-digit hexadecimal ASCII representation.

Definition 6.1

A homomorphism is a mapping \mathbf{h} with domain Σ^* for some alphabet Σ which preserves concatenation:

$$\mathbf{h}(\mathbf{v} \cdot \mathbf{w}) = \mathbf{h}(\mathbf{v}) \cdot \mathbf{h}(\mathbf{w}).$$

Proposition 6.2

The homomorphism is determined by the images of the single letters and $\mathbf{h}(\mathbf{w}) = \mathbf{h}(\mathbf{a}_1) \cdot \mathbf{h}(\mathbf{a}_2) \cdot \dots \cdot \mathbf{h}(\mathbf{a}_n)$ for a word $\mathbf{w} = \mathbf{a}_1\mathbf{a}_2 \dots \mathbf{a}_n$; $\mathbf{h}(\varepsilon) = \varepsilon$.

Definition 6.18

If \mathbf{h} is a homomorphism then $\mathbf{h}^{-1}(\mathbf{L}) = \{\mathbf{w} : \mathbf{h}(\mathbf{w}) \in \mathbf{L}\}$ is the inverse image of \mathbf{L} under \mathbf{h} .

Repetition 2

Theorem 6.5

The homomorphic images of regular and context-free languages are regular and context-free, respectively.

Theorem 6.11

Every language generated by some grammar is the homomorphic image of a context-sensitive language; in particular the class of context-sensitive languages is not closed under homomorphism.

Theorem 6.19

If \mathbf{L} is on level \mathbf{k} of the Chomsky hierarchy and \mathbf{h} is a homomorphism then $\mathbf{h}^{-1}(\mathbf{L})$ is also on level \mathbf{k} .

Generalised Homomorphisms map regular sets to regular sets such that they preserve unions, concatenations, Kleene star, \emptyset and $\{\varepsilon\}$. Whenever \mathbf{h} is a homomorphism then $\mathbf{L} \mapsto \{\mathbf{h}(\mathbf{u}) : \mathbf{u} \in \mathbf{L}\}$ is a generalised homomorphism.

Repetition 3

One can reduce the number of stars in $\bigcup_{a \in \Sigma} aa^*$ to two using intersection:

$$\begin{aligned} &00^* \cup 11^* \cup 22^* \cup 33^* = \\ &(\{0, 1, 2, 3\} \cdot \{00, 11, 22, 33\}^* \cdot \{\varepsilon, 0, 1, 2, 3\}) \cap \\ &(\{00, 11, 22, 33\}^* \cdot \{\varepsilon, 0, 1, 2, 3\}). \end{aligned}$$

The general result needs also a homomorphism.

Theorem 6.9

Let L be a regular language. Then there are two regular expressions σ, τ each containing only one Kleene star and some finite sets and concatenations and there is one homomorphism h such that L is described by $h(\sigma \cap \tau)$.

The idea is to encode states of a dfa into the symbols; expressions σ and τ test state-transitions at even and odd positions, respectively; h removes the state markers from the symbols.

Normal Forms

The normal forms are named after Noam Chomsky (born 7.12.1928) and Sheila Greibach (born 6.10.1939).

Assume that the language does not contain ε .

Chomsky Normal Form

All rules are of the form $A \rightarrow BC$ or $A \rightarrow d$ where A, B, C are non-terminals and d is a terminal.

Greibach Normal Form

All rules are of the form $A \rightarrow bw$ where b is a terminal and A a non-terminal and w a (possibly empty) string of non-terminals.

If the language contains ε , one allows in both normal forms $S \rightarrow \varepsilon$ for the start symbol S which then is not allowed to appear on the right side of a rule.

CNF Algorithm Steps 1 and 2

Given context-free grammar (N_0, Σ, P_0, S) .

1. Dealing with ε : Let $N_1 = N_0 \cup \{S'\}$ and

$$P_1 = P_0 \cup \{S' \rightarrow S\};$$

While there are rules $A \rightarrow vBw, B \rightarrow \varepsilon$ in P_1 with

$A \rightarrow vw$ not in P_1 Do Begin $P_1 = P_1 \cup \{A \rightarrow vw\}$ End;

Remove all rules $A \rightarrow \varepsilon$ with $A \in N_0$ from P_1 ;

Continue with grammar (N_1, Σ, P_1, S') .

2. Dealing with single terminal letters: Let $N_2 = N_1, P_2 = P_1$;

While there are $a \in \Sigma$ and rule $A \rightarrow vaw$ in P_2 with

$vw \neq \varepsilon$ Do Begin Choose a new non-terminal $B \notin N_2$;

Replace in all rules in P_2 all occurrences of a by B ;

Update $N_2 = N_2 \cup \{B\}$ and $P_2 = P_2 \cup \{B \rightarrow a\}$ End;

Continue with grammar (N_2, Σ, P_2, S') .

CNF Algorithm Steps 3 and 4

- 3. Breaking long right hand sides:** Let $N_3 = N_2$ and $P_3 = P_2$;
While there is $A \rightarrow Bw$ in P_3 with $|Bw| \geq 3$
Do Begin Choose a new non-terminal $C \notin N_3$ and let
 $N_3 = N_3 \cup \{C\}$; Add the rules $A \rightarrow BC, C \rightarrow w$ into P_3
and remove the rule $A \rightarrow Bw$ from P_3 End;
Continue with grammar (N_3, Σ, P_3, S') .
- 4. Removing rules $A \rightarrow B$:** Make a table of all $(A, B), (A, c)$
such that $A, B \in N_3, c \in \Sigma$ and, in the grammar
 (N_3, Σ, P_3, S') , $A \Rightarrow^* B$ and $A \Rightarrow^* c$, respectively;
Let $N_4 = N_3$ and P_4 contain the following rules:
 $S' \rightarrow \varepsilon$ in the case that this rule is in P_3 ;
 $A \rightarrow a$ in the case that (A, a) is in the table;
 $A \rightarrow BC$ in the case that there is $D \rightarrow EF$ in P_3 with
 $(A, D), (E, B), (F, C)$ in the table;
The grammar (N_4, Σ, P_4, S') is in Chomsky Normalform.

Example for Chomsky Normalform

The grammar $(\{S\}, \{0, 1\}, \{S \rightarrow 0S0 \mid 1S1 \mid 00 \mid 11\}, S)$ generates all palindromes of even nonzero length.

Chomsky Normal Form

Non-terminals: $\{S, T, U, V, W\}$; terminals: $\{0, 1\}$; start symbol: S ; rules: $S \rightarrow TV \mid UW$, $T \rightarrow VS \mid 0$, $U \rightarrow WS \mid 1$, $V \rightarrow 0$, $W \rightarrow 1$.

The derivation $S \Rightarrow 0S0 \Rightarrow 01S10 \Rightarrow 010010$ in the old grammar is equivalent to

$S \Rightarrow TV \Rightarrow VSV \Rightarrow 0SV \Rightarrow 0S0 \Rightarrow 0UW0 \Rightarrow 0WSW0 \Rightarrow 0WS10 \Rightarrow 01S10 \Rightarrow 01TV10 \Rightarrow 010V10 \Rightarrow 010010$ in the new grammar.

Exercises 7.4 – 7.6

Bring the following grammars into Chomsky Normal Form.

Exercise 7.4

$(\{S, T\}, \{0, 1\}, \{S \rightarrow TTTT, T \rightarrow 0T1|\varepsilon\}, S)$.

Exercise 7.5

$(\{S, T\}, \{0, 1\}, \{S \rightarrow ST|T, T \rightarrow 0T1|01\}, S)$.

Exercise 7.6

$(\{S\}, \{0, 1\}, \{S \rightarrow 0SS11SS0, 0110\}, S)$.

Always choose exercises of the right difficulty level: Students who are good in the field should take challenging exercises; students who have difficulties should take exercises they can master. So everyone should learn a bit from the exercises done by oneself; students should also try out exercises they do not present.

Removing Useless Non-Terminals

Let (N_0, Σ, P_0, S) be in Chomsky Normal Form.

- 1. Removing non-terminating non-terminals:** Let $N_1 = \{A \in N_0 : \text{there is a rule } A \rightarrow a \text{ or } A \rightarrow \varepsilon \text{ in } P_0\}$;
While there is a rule $A \rightarrow BC$ in P_0 with $A \in N_0 - N_1$ and $B, C \in N_1$ Do Begin $N_1 = N_1 \cup \{A\}$ End;
Let $P_1 = \{A \rightarrow w \text{ from } P_0 \text{ with } A \in N_1 \text{ and } w \in N_1 \cdot N_1 \cup \Sigma \cup \{\varepsilon\}\}$;
If $S \notin N_1$ then terminate with empty grammar else continue with grammar (N_1, Σ, P_1, S) .
- 2. Selecting all reachable non-terminals:** Let $N_2 = \{S\}$;
While there is a rule $A \rightarrow BC$ in P_1 with $A \in N_2$ and $\{B, C\} \not\subseteq N_2$ Do Begin $N_2 = N_2 \cup \{B, C\}$ End;
Let $P_2 = \{A \rightarrow w \text{ from } P_1 : A \in N_2\}$;
The grammar (N_2, Σ, P_2, S) does not contain any useless non-terminal.

Quiz 7.8

$\Sigma = \{0\}$; $N = \{Q, R, S, T, U, V, W, X, Y, Z\}$;
rules $S \rightarrow TU|UV$, $T \rightarrow UT|TV|TW$, $R \rightarrow VW|QQ|0$,
 $Q \rightarrow 0$, $U \rightarrow VW|WX$, $V \rightarrow WX|XY|0$, $W \rightarrow XY|YZ|0$;
start symbol S .

Determine the set of reachable and terminating non-terminals.

Exercises 7.9 and 7.10

Exercise 7.9: Consider the grammar

$(\{S_0, S_1, \dots, S_9\}, \{0\}, \{S_0 \rightarrow S_0S_0, S_1 \rightarrow S_2S_3, S_2 \rightarrow S_4S_6|0, S_3 \rightarrow S_6S_9, S_4 \rightarrow S_8S_2, S_5 \rightarrow S_0S_5, S_6 \rightarrow S_2S_8, S_7 \rightarrow S_4S_1|0, S_8 \rightarrow S_6S_4|0, S_9 \rightarrow S_8S_7\}, S_1)$.

Exercise 7.10: Consider the grammar

$(\{S_0, S_1, \dots, S_9\}, \{0\}, \{S_0 \rightarrow S_1S_1, S_1 \rightarrow S_2S_2, S_2 \rightarrow S_3S_3, S_3 \rightarrow S_0S_0|S_4S_4, S_4 \rightarrow S_5S_5, S_5 \rightarrow S_6S_6|S_3S_3, S_6 \rightarrow S_7S_7|0, S_7 \rightarrow S_8S_8|S_7S_7, S_8 \rightarrow S_7S_6|S_8S_6, S_9 \rightarrow S_7S_8|0\}, S_1)$.

For each of the two grammars, determine the set of reachable and terminating non-terminals and explain the steps on the way to this set. What is the shortest word generated by the grammar?

Emptiness Check

The removing of useless non-terminals results in an empty grammar in the case that the grammar does not generate any word. Here an algorithm for a context-free grammar (N, Σ, P, S) which is not in any normalform.

Initialisation: Let $N' = \emptyset$;

Loop: While there are $A \in N - N'$ and a rule $A \rightarrow w$ with $w \in (N' \cup \Sigma)^*$
Do Begin $N' = N' \cup \{A\}$ End;

Decision: If $S \notin N'$ then the language of the grammar is empty else the language of the grammar contains some word.

Finiteness Check

Let (N, Σ, P, S) be in Chomsky Normal Form. One computes $N''(A)$ as the set of non-terminals occurring in a derivation of a terminal word from A after at least one step.

Initialisation 1: Let $N' = \emptyset$;

Loop 1: While there are $A \in N - N'$ and a rule $A \rightarrow w$ with $w \in (N' \cup \Sigma)^*$
Do Begin $N' = N' \cup \{A\}$ End;

Initialisation 2: For all $A \in N$, let $N''(A) = \emptyset$;

Loop 2: While there are $A, B, C, D \in N'$ and a rule $B \rightarrow CD$ with $B \in N''(A) \cup \{A\}$ and $\{C, D\} \not\subseteq N''(A)$
Do Begin $N''(A) = N''(A) \cup \{C, D\}$ End;

Decision: If there is $A \in N''(S) \cup \{S\}$ with $A \in N''(A)$ then the language of the grammar is infinite else it is finite.

Exercise 7.13: Time Complexity

Polynomial time means that the algorithm uses time $p(n)$ for inputs of length n to decide some property. If a grammar is in Chomsky Normal Form, n can just be the number of non-terminals.

The checks whether a grammar in CNF generates some word or generates infinitely many words are in polynomial time. How complex is it to not only check whether some word is generated but also to output one such witness? Possible answers are polynomial time, exponential time and double exponential time. Give reasons for the answer.

Exercises 7.14 – 7.17

For grammar $(\{S, T, U, V, W\}, \{0, 1, 2\}, P, S)$ with below rules determine how many words the grammar generates: (a) None, (b) One, (c) Two, (d) Three, (e) Finitely many and at least four, (f) Infinitely many?

Exercise 7.14: The rules are $S \rightarrow TT$, $T \rightarrow UU$,
 $U \rightarrow VW|WV$, $V \rightarrow 0$, $W \rightarrow 1$.

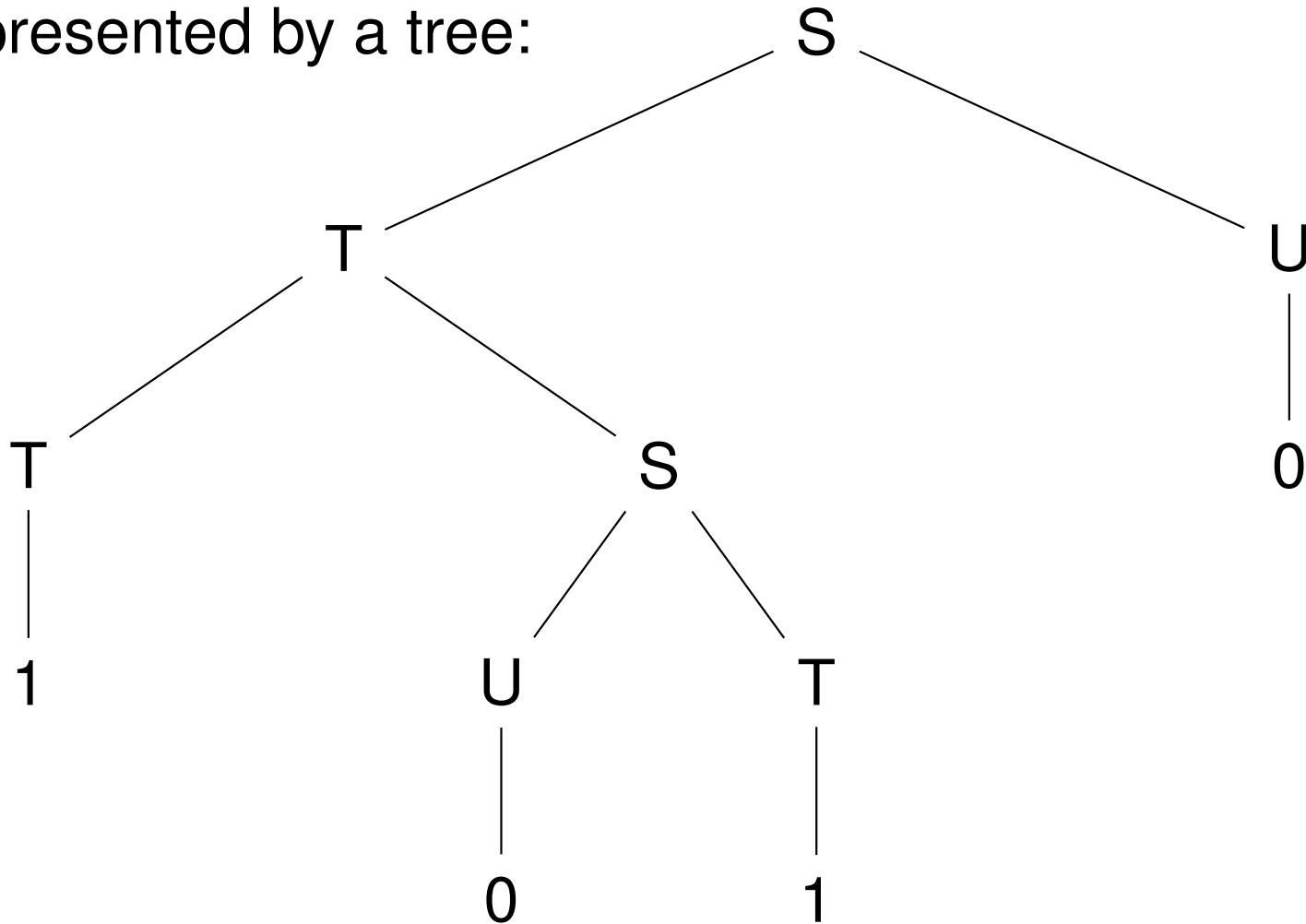
Exercise 7.15: The rules are $S \rightarrow ST$, $T \rightarrow TU$, $U \rightarrow UV$,
 $V \rightarrow VW$, $W \rightarrow 0$.

Exercise 7.16: The rules are $S \rightarrow UT|TU|2$, $T \rightarrow VV$,
 $U \rightarrow WW$, $V \rightarrow 0$, $W \rightarrow 1$.

Exercise 7.17: The rules are $S \rightarrow SS|TT|UU$, $T \rightarrow VV$,
 $U \rightarrow WW$, $V \rightarrow 0$, $W \rightarrow WW$.

Derivation Tree

For grammar $(\{S, T, U\}, \{0, 1\}, \{S \rightarrow SS|TU|UT, U \rightarrow 0|US|SU, T \rightarrow 1|TS|ST\}, S)$, a derivation $S \Rightarrow TU \Rightarrow TSU \Rightarrow TUTU \Rightarrow 1UTU \Rightarrow 10TU \Rightarrow 101U \Rightarrow 1010$ can be represented by a tree:



Properties of Derivation Trees

Each node has a symbol in it.

If the symbol is a non-terminal then the successor nodes have the symbols of the rule applied at this symbol; if the symbol is a terminal then the node is a leaf.

The derivation does not depend on the order of rules on incomparable nodes are applied; however, the rule of each node has to be applied before it is applied to any successor.

The leaves below a node have the part of the word generated from the corresponding symbol.

Trees are more precise than derivations by lists of rules, as they explicitly say which non-terminal derives into which symbols and these are on the successor nodes.

Sometimes a derivation is ambiguous: For $TU \Rightarrow TSU$, the rule can be $T \rightarrow TS$ or $U \rightarrow SU$.

Number of Derivation Trees

As the rules to derive a word are not always unique but can be ambiguous, there can be different derivation trees for the same word. Indeed, one can prove that for some grammars it is impossible to have the derivation tree all the time unique. Consider the grammar

$(\{S, T, U\}, \{0, 1\}, \{S \rightarrow SS|TU|UT, U \rightarrow 0|US|SU, T \rightarrow 1|TS|ST\}, S)$, how many derivation trees has the word below?

Exercise 7.19: How many derivation trees are there for **011001**?

Exercise 7.20: How many derivation trees are there for **000111**?

Exercise 7.21

One can generalise the notion of derivation tree to all context-free grammars and might then have nodes with more than two successors. Consider the grammar

$$(\{S, T\}, \{0, 1, 2\}, \{S \rightarrow TT, T \rightarrow 0T1|2\}, S).$$

and draw the derivation tree for the word **00211021**. Prove that all words in the language of this grammar have a unique derivation tree.

Pumping Lemma

Theorem 2.15 (b)

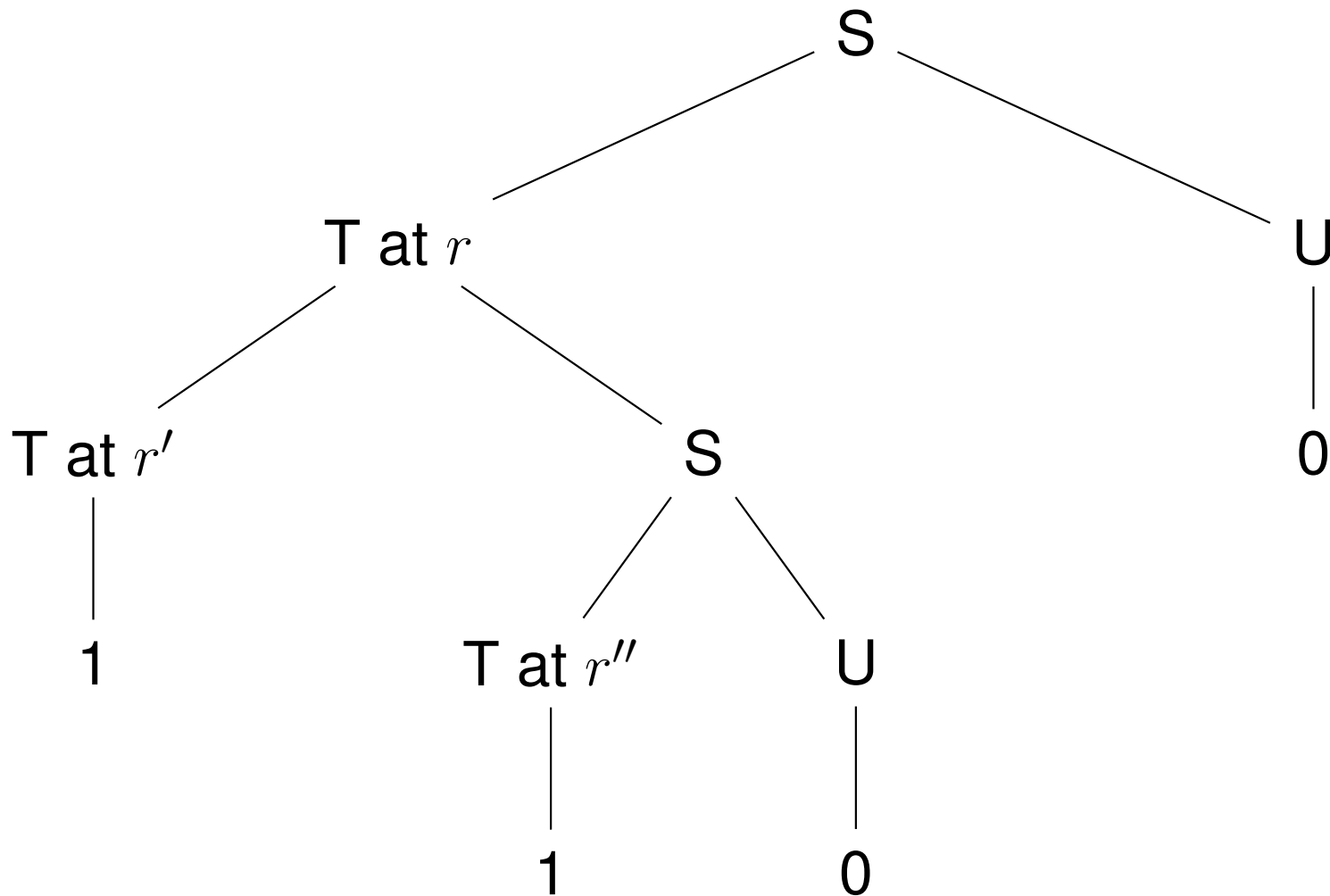
Let $\mathbf{L} \subseteq \Sigma^*$ be an infinite context-free language generated by a grammar $(\mathbf{N}, \Sigma, \mathbf{P}, \mathbf{S})$ in Chomsky Normal Form with \mathbf{h} non-terminals. Then the constant $\mathbf{k} = 2^{\mathbf{h}+1}$ satisfies that for every $\mathbf{u} \in \mathbf{L}$ of length at least \mathbf{k} there is a representation $\mathbf{vwxyz} = \mathbf{u}$ such that $|\mathbf{wxy}| \leq \mathbf{k}$, ($\mathbf{w} \neq \varepsilon$ or $\mathbf{y} \neq \varepsilon$) and $\mathbf{vw}^\ell \mathbf{xy}^\ell \mathbf{z} \in \mathbf{L}$ for all $\ell \in \mathbb{N}$.

Proof Idea

In the derivation tree of a word \mathbf{u} longer than $2^{\mathbf{h}+1}$, find the lowest node having a non-terminal \mathbf{A} which is on a branch somewhere below. Below \mathbf{A} , no branch repeats a non-terminal and therefore each branch has at most the length \mathbf{h} . Thus there are at most $2^{\mathbf{h}+1}$ leaves below \mathbf{A} .

Since \mathbf{A} itself repeats, one has that $\mathbf{S} \Rightarrow^* \mathbf{vAz} \Rightarrow^* \mathbf{vwAyz} \Rightarrow^* \mathbf{vwxyz}$ and $|\mathbf{wxy}| \leq 2^{\mathbf{h}+1}$ and $\mathbf{A} \Rightarrow^* \mathbf{wAy}$ and $\mathbf{wy} \neq \varepsilon$.

Example of Derivation Tree



Lowest node r with non-terminal repeated below at r' or r'' .
Possible Pumpings: $1(10)^l 0$, $1^l 10^l 0$.

Ogden's Lemma

Marked Symbols

Mark the first four **1** in word **0000011111** as **0000011111**.

The word can be pumped such that at least one but at most four marked symbols are pumped or between the pumped parts: **000 0^ℓ 01 1^ℓ 111**.

Theorem 7.22

Let $\mathbf{L} \subseteq \Sigma^*$ be an infinite context-free language generated by a grammar $(\mathbf{N}, \Sigma, \mathbf{P}, \mathbf{S})$ in Chomsky Normal Form with \mathbf{h} non-terminals. Then the constant $\mathbf{k} = 2^{\mathbf{h}+1}$ satisfies that for every $\mathbf{u} \in \mathbf{L}$ with at least \mathbf{k} marked symbols, there is a representation $\mathbf{vwxyz} = \mathbf{u}$ such that \mathbf{wxy} contains at most \mathbf{k} marked symbols, \mathbf{wy} contains at least $\mathbf{1}$ marked symbol and $\mathbf{vw}^\ell \mathbf{xy}^\ell \mathbf{z} \in \mathbf{L}$ for all $\ell \in \mathbb{N}$.

Example 7.23

The Language

Let \mathbf{L} be the language of all words $w \in 1^+(0^+1^+)^+$ with no two runs of zeroes of equal lengths. That is, words in \mathbf{L} start and end with $\mathbf{1}$ and do not contain a subword 10^h1 twice for any $h > 0$.

\mathbf{L} contains words 1000101 and 1100110001 and 1111111 .
 \mathbf{L} neither contains 100111001 nor 10100110100001 .

Traditional Context-Free Pumping Lemma Satisfied

More precisely: Each word longer than $\mathbf{2}$ symbols in \mathbf{L} can somewhere be pumped by a single one-symbol pump.

If $w \in 1^+0^+1^+$ then pump some $\mathbf{0}$ in the middle.

If w contains a border of at least two $\mathbf{1}$ then pump a $\mathbf{1}$ in this border.

If w has a longest run of $\mathbf{0}$ separated by a single $\mathbf{1}$ from another run of $\mathbf{0}$ then pump this separating $\mathbf{1}$.

L does not satisfy Ogden's Lemma

If k is a constant supposed to work for Ogden's Pumping Lemma then consider $u = 1010^210^31 \dots 10^{4k}1$ and mark the zeroes of the subword 10^k1 .

Let $vwxyz$ split u into five parts.

If w or y in $\{0, 1\}^+ - \{0\}^* - \{1\}^*$ then vw^4xy^4z contains a subword 10^h1 at least twice.

One of w and y must be in 0^+ and contain some of the marked zeroes.

In the word $vwwxyyz$ pumping has replaced 10^k1 by $10^{k+h}1$ for $h \in \{1, 2, \dots, k\}$ and thus the subword $10^{k+h}1$ in u must also be pumped and replaced by $10^{k+h+l}1$ for $l \in \{1, 2, \dots, k+h\}$. However, $10^{k+h+l}1$ is also a subword of u which occurs in $vwwxyyz$ and thus $vwwxyyz \notin L$ (as $h \leq k$ and $l \leq 2k$ and $h + k + l \leq 4k$).

Square-Containing Words

Theorem 7.24 [Ehrenfeucht and Rozenberg 1983; Ross and Winklmann 1982]

The language L of square-containing words over $\{0, 1, 2\}$ is not context-free.

Proposition 7.25

The language L of square-containing words over $\{0, 1, 2\}$ satisfies Ogden's Lemma with constant 6.

If $u = 012012102012012102$ then one can see that $012(01)^*2102012012102 \subseteq L$. Details on next page.

Proof of Proposition

For given word $u \in L$ with at least six letters marked, consider those splittings $vwxyz = u$ where the following conditions are met:

- v ends with the same letter a with which z starts;
- w contains at least one marked letter;
- x, y are ε .

Such splittings exist obviously when there is $a \in \{0, 1, 2\}$ such that there are three marked letters a or there are two marked letters a with one other marked letter in between.

The remaining case is wlog of form $r_0 0 r_1 0 r_2 1 r_3 1 r_4 2 r_5 2 r_6$. If r_1 is in 0^* then one let $w = 1$ and v everything left of the first 1 and z everything right of it. If r_1 contains a $b \neq 0$ then one let w be all the part right of this b and left of the first marked b and w contains the marked letter 0 .

Exercises 7.27 and 7.28

Exercise 7.27

Prove that the language

$$\mathbf{L} = \{ \mathbf{a}^{\mathbf{h}} \cdot \mathbf{w} : \mathbf{a} \in \{0, 1, 2\}, \mathbf{w} \in \{0, 1, 2\}^*, \mathbf{w} \text{ is square-free and } \mathbf{h} \in \mathbb{N} \}$$

satisfies Theorem 2.15 (b) but does not satisfy Ogden's Pumping Lemma. The fact that there are infinitely many square-free words can be used without proof.

Exercise 7.28

Use the Block Pumping Lemma to prove the following variant of Ogden's Lemma for regular languages: If a language \mathbf{L} satisfies the Block Pumping Lemma with constant $\mathbf{k} + 1$ then one can, for each word \mathbf{u} having at least \mathbf{k} marked symbols, find a splitting of the word into parts $\mathbf{x}, \mathbf{y}, \mathbf{z}$ such that $\mathbf{u} = \mathbf{xyz}$ and $\mathbf{xy}^* \mathbf{z} \subseteq \mathbf{L}$ and \mathbf{y} contains at least $\mathbf{1}$ and at most \mathbf{k} marked symbols.

Greibach Normal Form

Example 7.29

Consider $(\{S\}, \{0, 1\}, \{S \rightarrow 0S0|1S1|00|11|0|1\}, S)$. For the Greibach Normal Form, one needs two additional non-terminals T, U and updates the rules as follows:

$$S \rightarrow 0ST|1SU|0T|1U|0|1, T \rightarrow 0, U \rightarrow 1.$$

Consider $(\{S\}, \{0, 1\}, \{S \rightarrow SS|0S1|1S0|10|01\}, S)$. For the Greibach Normal Form, one needs two additional non-terminals T, U and updates the rules as follows:

$$S \rightarrow 0SU|0U|1ST|1T, T \rightarrow 0|0S, U \rightarrow 1|1S.$$

Exercises 7.30 and 7.31

Construct Greibach Normal Form for the intersection of $0^*1^*0^*1^*$ with each of these two languages, respectively.

Languages and Derivatives

Exercise 7.32. Prove the following rules of the derivative with $\mathbf{a} \in \Sigma$ and $\mathbf{x} \in \Sigma^*$:

- $(\mathbf{L} \cup \mathbf{H})_{\mathbf{x}} = \mathbf{L}_{\mathbf{x}} \cup \mathbf{H}_{\mathbf{x}}$ and $(\mathbf{L} \cap \mathbf{H})_{\mathbf{x}} = \mathbf{L}_{\mathbf{x}} \cap \mathbf{H}_{\mathbf{x}}$;
- If $\varepsilon \in \mathbf{L}$ then $(\mathbf{L} \cdot \mathbf{H})_{\mathbf{a}} = \mathbf{L}_{\mathbf{a}} \cdot \mathbf{H} \cup \mathbf{H}_{\mathbf{a}}$ else $(\mathbf{L} \cdot \mathbf{H})_{\mathbf{a}} = \mathbf{L}_{\mathbf{a}} \cdot \mathbf{H}$;
- $(\mathbf{L}^*)_{\mathbf{a}} = \mathbf{L}_{\mathbf{a}} \cdot \mathbf{L}^*$.

The following theorem characterises the context-free languages by stating that all derivatives have to be formed from a finite set of languages using concatenation and union; furthermore, all the derivatives of these finitely many languages satisfy the same condition.

Theorem 7.33. A language \mathbf{L} is context-free iff there is a finite list of languages $\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_n$ with $\mathbf{L} = \mathbf{H}_1$ such that for every word \mathbf{x} and every \mathbf{H}_m , $(\mathbf{H}_m)_{\mathbf{x}}$ is a finite union of finite products of some \mathbf{H}_k .

Exercise 7.34. Prove this theorem using Exercise 7.32.