Automata and Grammars

Prof. Dr. Friedrich Otto

Fachbereich Elektrotechnik/Informatik, Universität Kassel
34109 Kassel, Germany

E-mail: f.otto@uni-kassel.de

SS 2018

Lectures and Seminary SS 2018

Lectures:

Thursday 9:00 - 10:30, Room S 11

Start: Thursday, February 22, 2018, 9:00.

Seminary:

Thursday 10:40 - 12:10, Room S 11

Start: Thursday, February 22, 2018, 10:40.

Exercises:

From Thursday to Thursday next week!

To pass seminary:
At least two successful presentations in class (10 %)
and passing of midtem exam on April 5 (10:40 - 11:40) (20 %)

Final Exam: (dates are still preliminary)
Written exam on May 31 (9:00 - 11:00)
or on June 14 (9:00 - 11:00) (50 %)
Oral exam on June 14 or on June 28 (20 %).

Moodle: Course ”Automata and Grammars” NTIN071.

Literature:

P.J. Denning, J.E. Dennis, J.E. Qualitz;
Machines, Languages, and Computation.
Prentice-Hall, Englewood Cliffs, N.J., 1978.

M. Harrison; *Introduction to Formal Language Theory.*
Addison-Wesley, Reading, M.A., 1978.

J.E. Hopcroft, R. Motwani, J.D. Ullman; *Introduction to Automata Theory, Languages, and Computation.*
Addison-Wesley, Boston, 2nd. ed., 2001.

H.R. Lewis, Ch.H. Papadimitriou;
Elements of the Theory of Computation.
Prentice Hall, Englewood Cliffs, N.J., 1981.

G. Rozenberg, A. Salomaa (editors);
Handbook of Formal Languages, Vol.1: Word, Language, Grammar. Springer, Heidelberg, 1997.

1. Introduction

Formal Languages (not natural languages):
A formal language is a set of (finite) sequences of symbols (words, strings) from a fixed finite set of symbols (alphabet).

How to describe a formal language? There are various options:

- by a complete enumeration, but:

$$L = \{ w \in \{a, b, c\}^* \mid |w| = 20 \} : |L| = 3^{20} \sim 3.5 \times 10^9$$

- by a mathematical expression, but: how to check that a given word belongs to the language described?

- by a generative device: a grammar or a rewriting system.
A grammar tells you how to generate (all) the words of a language!
- by an analytical device: an automaton or an algorithm.
An automaton recognizes exactly the words of a given language!

The Roots of the Theory of Formal Languages:

- Combinatorics on Words (A. Thue 1906, 1912)
- Semigroup and Group Theory (M. Dehn 1911)
- Logic (A. Turing 1926, E. Post 1936, A. Church 1936)

Church's Thesis

A function is effectively computable iff there is a Turing machine that computes this function.

Let $f : \Sigma^* \rightsquigarrow \Gamma^*$ be a (partial) function.

$dom(f)$	=	$\{ u \in \Sigma^* \mid f(u) \text{ is defined} \}$:	domain
$range(f)$	=	$\{ v \in \Gamma^* \mid \exists u \in \Sigma^* : f(u) = v \}$:	range
$ker(f)$	=	$\{ u \in \Sigma^* \mid f(u) = \varepsilon \}$:	kernel
$graph(f)$	=	$\{ u\#f(u) \mid u \in dom(f) \}$:	graph

f is computable iff there is an “algorithm” that accepts the language $graph(f)$.

Classes of Automata: Restrictions of the Turing machine

- Linguistics	(N. Chomsky 1956)	:	phrase structure grammars
- Biology	(A. Lindenmayer 1968)	:	L-systems
	(T. Head 1987)	:	DNA-computing, H-systems
	(G. Paun 1999)	:	Membrane computing, P-systems

Overview:

Chapter 2: Regular Languages and Finite Automata
Chapter 3: Context-free Languages and Pushdown Automata
Chapter 4: Context-sensitive Languages, Recursively Enumerable Languages and Turing Machines
Chapter 5: Summary

Chapter 2:

Regular Languages and Finite Automata

2.1. Words, Languages, and Morphisms

An alphabet Σ is a finite set of symbols or letters.
For all $n \in \mathbb{N}$: $\Sigma^n = \{u : [1, n] \to \Sigma\}$: set of words of length n over Σ, that is, $u = (u(1), u(2), \ldots, u(n)) = u_1 u_2 \ldots u_n$.

$$\Sigma^0 = \{\varepsilon\} : \varepsilon = \text{empty word}$$

$$\Sigma^+ = \bigcup_{n \geq 1} \Sigma^n : \text{set of all non-empty words over } \Sigma$$

$$\Sigma^* = \bigcup_{n \geq 0} \Sigma^n : \text{set of all words over } \Sigma$$

The concatenation $\cdot : \Sigma^* \times \Sigma^* \to \Sigma^*$ is defined through $u \cdot v = uv$.
For $u \in \Sigma^m$ and $v \in \Sigma^n$, $uv \in \Sigma^{m+n}$.

The length function $|.| : \Sigma^* \to \mathbb{N}$ is defined through $|u| = n$ for all $u \in \Sigma^n, n \geq 0$.

Lemma 2.1

(a) *The operation of concatenation is associative, that is,* $(u \cdot v) \cdot w = u \cdot (v \cdot w)$.

(b) *For all* $u \in \Sigma^*$, $u \cdot \varepsilon = \varepsilon \cdot u = u$.

(c) *If* $u \cdot v = u \cdot w$, *then* $v = w$ *(Left cancellability)*.

(d) *If* $u \cdot w = v \cdot w$, *then* $u = v$ *(Right cancellability)*.

(e) *If* $u \cdot v = x \cdot y$, *then exactly one of the following cases holds:*

(1) $|u| = |x|$, $u = x$, *and* $v = y$.

(2) $|u| > |x|$, *and there exists* $z \in \Sigma^+$ *such that* $u = x \cdot z$ *and* $y = z \cdot v$.

(3) $|u| < |x|$, *and there exists* $z \in \Sigma^+$ *such that* $x = u \cdot z$ *and* $v = z \cdot y$.

Abbreviations:

uv stands for $u \cdot v$.

$u^0 = \varepsilon, u^1 = u, u^{n+1} = u^n u$ for all $u \in \Sigma^*, n \geq 1$.

Further Basic Notions:

The mirror function $^R : \Sigma^* \to \Sigma^*$ is defined through

$$\varepsilon^R = \varepsilon, (ua)^R = au^R \text{ for all } u \in \Sigma^*, a \in \Sigma.$$

Hence, $(abbc)^R = c(abb)^R = cb(ab)^R = cbba$.

If $uv = w$, then u is a prefix and v is a suffix of w.

If $u \neq \varepsilon$ $(v \neq \varepsilon)$, then v is a proper suffix (proper prefix) of w.

If $uvz = w$, then v is a factor of w.

It is a proper factor, if $uz \neq \varepsilon$.

A language L over Σ is a subset $L \subseteq \Sigma^*$.

The cardinality of L is denoted by $|L|$.

Let $L, L_1, L_2 \subseteq \Sigma^*$.

$L_1 \cdot L_2 = \{ uv \mid u \in L_1, v \in L_2 \}$ is the product of L_1 and L_2.
It is also called the concatenation of L_1 and L_2.

$L^0 = \{\varepsilon\},\; L^1 = L,\; L^{n+1} = L^n \cdot L$ for all $(n \geq 1)$.

$L^+ = \bigcup_{n\geq 1} L^n$ and $L^* = \bigcup_{n\geq 0} L^n$ are the plus closure and the star closure (Kleene closure) of L.

$L^R = \{ w^R \mid w \in L \}$ is the mirror language of L.

Example:

(a) $\Sigma_1 := \{0,1\}$: $\varepsilon, 0, 10, 110 \in \Sigma_1^*$.

$L_1 = \{\, 10^i \mid i \geq 0 \,\}$:

$L_1^R = \{\, 0^j 1 \mid j \geq 0 \,\}$ und $L_1 \cdot L_1^R = \{\, 10^i 1 \mid i \geq 0 \,\}$.

$L_2 = \{\, 0^n 1^n \mid n \geq 1 \,\}$.

(b) $\Sigma_2 := \{a, b, c\}$:

$L_3 = \{\, wcw^R \mid w \in \{a,b\}^* \,\}$: marked palindromes of even length

$L_4 = \{\, ww \mid w \in \{a,b\}^* \,\}$: copy language

$L_5 = \{\, w \mid |w|_a \equiv 0 \bmod 2 \text{ and } |w|_b \equiv 1 \bmod 3 \,\}$.

(c) $\Sigma_3 := \{0, 1, \ldots, 9, ., +, -, e\}$: $112.45, -23e+17 \in \Sigma_3^+$

$L_6 = \{\text{unsigned integer in PASCAL}\}$

Let Σ and Δ be two alphabets.

A mapping $h : \Sigma^* \to \Delta^*$ is a morphism, if
the equality $h(uv) = h(u) \cdot h(v)$ holds for all $u, v \in \Sigma^*$.

Lemma 2.2

If $h : \Sigma^* \to \Delta^*$ is a morphism, then $h(\varepsilon) = \varepsilon$.

For a set S, 2^S denotes the power set of S.

A mapping $\varphi : \Sigma^* \to 2^{\Delta^*}$ is a substitution, if
the two following conditions are satisfied:

– $\forall u, v \in \Sigma^* : \varphi(uv) = \varphi(u) \cdot \varphi(v)$.
– $\varphi(\varepsilon) = \{\varepsilon\}$.

For $L \subseteq \Sigma^*$, $h(L) = \{\, h(w) \mid w \in L \,\}$ and $\varphi(L) = \bigcup_{w \in L} \varphi(w)$.

Example:

Let $\Sigma = \{a, b, c\}$ and $\Delta = \{0, 1\}$.
If $h : \Sigma^* \to \Delta^*$ is defined through $a \mapsto 01,\ b \mapsto 1,\ c \mapsto \varepsilon$,
then $h(baca) = 10101$.

If $\varphi : \Sigma^* \to 2^{\Delta^*}$ is defined through

$$a \mapsto \{01, 001\},\ b \mapsto \{1^i \mid i \geq 1\},\ c \mapsto \{\varepsilon\},$$

then $\varphi(baca) = \{\, 1^i0101, 1^i01001, 1^i00101, 1^i001001 \mid i \geq 1 \,\}$.

A morphism h is called ε-free, if $h(a) \neq \varepsilon$ for all $a \in \Sigma$.

A substitution φ is called ε-free, if $\varepsilon \notin \varphi(a)$ for all $a \in \Sigma$.

A substitution φ is called finite, if $\varphi(a)$ is a finite set for all $a \in \Sigma$.

$h^{-1} : \Delta^* \to 2^{\Sigma^*}$ is defined through $h^{-1}(v) = \{\, u \in \Sigma^* \mid h(u) = v \,\}$.
$\varphi^{-1} : \Delta^* \to 2^{\Sigma^*}$ is defined through $\varphi^{-1}(v) := \{\, u \in \Sigma^* \mid v \in \varphi(u) \,\}$.

These reverse mappings are in general not substitutions!

2.2 Regulare Grammars

A semi-Thue system (string-rewriting system) on an alphabet Σ is a (finite) set S of pairs of words over Σ :

$$S = \{\ell_1 \rightarrow r_1, \ldots, \ell_n \rightarrow r_n\} \; (n \geq 0, \ell_1, \ldots, \ell_n, r_1, \ldots, r_n \in \Sigma^*).$$

S induces a number of binary relations on Σ^*:

- the single-step derivation relation $\rightarrow_S$:
  $u \rightarrow_S v$ iff $\exists(\ell \rightarrow r) \in S \; \exists x, y \in \Sigma^* : u = x\ell y$ and $v = xry$.
- the derivation relation $\rightarrow_S^*$:
  $u \rightarrow_S^0 v$ iff $u = v$.
  $u \rightarrow_S^1 v$ iff $u \rightarrow_S v$.
  $u \rightarrow_S^{n+1} v$ iff $\exists w \in \Sigma^* : u \rightarrow_S^n w$ and $w \rightarrow_S v$.
  $u \rightarrow_S^+ v$ iff $\exists n \geq 1 : u \rightarrow_S^n v$.
  $u \rightarrow_S^* v$ iff $\exists n \geq 0 : u \rightarrow_S^n v$.

Lemma 2.3

(a) *The relation $\rightarrow_S^*$ is the smallest reflexive and transitive binary relation on Σ^* that contains $\rightarrow_S$.*

(b) *If $u \rightarrow_S^* v$, then $xuy \rightarrow_S^* xvy$ for all $x, y \in \Sigma^*$.*

A phrase-structure grammar G is a 4-tuple $G = (N, T, S, P)$, where

- N is a finite alphabet of nonterminals (variables),
- T is a finite alphabet of terminals (terminal symbols), where $N \cap T = \emptyset$,
- $S \in N$ is the start symbol, and
- $P \subseteq (N \cup T)^* \times (N \cup T)^*$ is a finite semi-Thue system on $N \cup T$, the elements of which are called productions.
  For each production $(\ell \rightarrow r) \in P$, it is required that $|\ell|_N \geq 1$.

For $A \in N$,

$$\hat{L}(G, A) = \{ \alpha \in (N \cup T)^* \mid A \rightarrow_P^* \alpha \}$$

is the set of A-sentential forms, and

$$L(G, A) = \{ w \in T^* \mid A \rightarrow_P^* w \}$$

is the set of terminal words that are derivable from A.

$\hat{L}_G = \hat{L}(G, S)$ is the set of sentential forms that are derivable in G and $L_G = L(G, S)$ is the language generated by G.

The grammar $G = (N, T, S, P)$ is called left regular, if $\ell \in N$ and $r \in T^* \cup N \cdot T^*$ for each production $(\ell \rightarrow r) \in P$.

G is called right regular, if $\ell \in N$ and $r \in T^* \cup T^* \cdot N$ for each production $(\ell \rightarrow r) \in P$.

Finally, G is called regular, if it is left or right regular.

Example:

(a) $G_1 = (N, T, S, P)$, where $N = \{S, A\}$, $T = \{0, 1\}$ and
$P = \{S \to 0A, A \to 10A, A \to \varepsilon\}$.
G_1 is right regular, and $L(G_1) = \{\, 0(10)^i \mid i \geq 0 \,\}$.

(b) $G_2 = (N, T, S, P)$, where $N = \{S\}$, $T = \{0, 1\}$ and
$P = \{S \to S10, S \to 0\}$.
G_2 is left regular, and $L(G_2) = \{\, 0(10)^i \mid i \geq 0 \,\} = L(G_1)$.

A language $L \subseteq \Sigma^*$ is called regular, if there exists a regular grammar G such that $L_G = L$.

$\mathrm{REG}(\Sigma)$	=	set of regular languages on Σ
REG	=	class of all regular languages

Remark 2.4

(a) *Each finite language is regular.*

(b) *If L is a regular language, then so is its mirror language L^R.*

(c) *If $L \in \mathrm{REG}(\Sigma)$, and if $h : \Sigma^* \to \Delta^*$ is a morphism, then $h(L) \in \mathrm{REG}(\Delta)$, that is, the class* REG *is closed under morphisms.*

(d) REG *is also closed under finite substitutions.*

A right regular grammar $G = (N, T, S, P)$ is in right normal form if $r \in T \cdot N \cup T$ for each production $(l \to r) \in P$. In addition, G may contain the production $(S \to \varepsilon)$, if S does not occur on the right-hand side of any production.

If G is in right normal form, then it does not contain any productions of the following forms:

- $A \to \varepsilon$ $(A \neq S)$: ε-rule,
- $A \to B$ $(A, B \in N)$: chain rule,
- $A \to wB$ or $A \to w$, where $w \in T^*$, $|w| \geq 2$.

Theorem 2.5

From a right regular grammar G, a grammar $\hat{G}$ in right normal form can be constructed such that $L(\hat{G}) = L(G)$.

Proof of Theorem 2.5.

Let $G = (N, T, S, P)$ be a right regular grammar that is not in right normal form. First we eliminate the ε-rules from P.

(1) Determine the set $N_1 = \{ A \in N \mid A \rightarrow_P^* \varepsilon \}$:
$N_1^{(1)} := \{ A \in N \mid (A \rightarrow \varepsilon) \in P \}$,
$N_1^{(k+1)} := N_1^{(k)} \cup \{ A \in N \mid \exists B \in N_1^{(k)} : (A \rightarrow B) \in P \}$.
Then $N_1 = \bigcup_{k \geq 1} N_1^{(k)} = N_1^{(|N|)}$.

(2) Remove all ε-rules.

(3) For each production $B \rightarrow wA$, where $|w| > 0$ and $A \in N_1$, add the production $B \rightarrow w$.

(4) If $S \in N_1$, then introduce a new start symbol $\hat{S}$ and the productions $\hat{S} \rightarrow \varepsilon$ and $\hat{S} \rightarrow \alpha$ for each production $S \rightarrow \alpha$.

Example:

Let $G = (N, T, S, P)$, where $N = \{S, A, B, C, D\}$, $T = \{a, b\}$, and

$$
\begin{array}{rlllll}
P = \{ & S \to \varepsilon, & S \to abA, & S \to B, & A \to abS, & A \to B, \\
& B \to C, & B \to b^3C, & B \to D, & C \to A, & C \to aab, \\
& D \to \varepsilon, & D \to a, & D \to aab, & D \to abD\}. &
\end{array}
$$

Elimination of ε-rules:

(1) $N_1^{(1)} = \{S, D\}$, $N_1^{(2)} = \{S, D, B\}$, $N_1^{(3)} = \{S, D, B, A\}$, $N_1^{(4)} = \{S, D, B, A, C\} = N = N_1$.

(2) Delete productions $S \to \varepsilon$ and $D \to \varepsilon$.

(3) Add productions $S \to ab$, $A \to ab$, $B \to bbb$, and $D \to ab$.

(4) Introduce $\hat{S}$ and $\hat{S} \to \varepsilon$, $\hat{S} \to abA$, $\hat{S} \to ab$, and $\hat{S} \to B$.

Then $G_1 = (N \cup \{\hat{S}\}, T, \hat{S}, P_1)$ is equivalent to G, where

$$
\begin{array}{rlllll}
P_1 = \{ & \hat{S} \to \varepsilon, & \hat{S} \to abA, & \hat{S} \to ab, & \hat{S} \to B, & S \to abA, \\
& S \to ab, & S \to B, & A \to abS, & A \to ab, & A \to B, \\
& B \to C, & B \to b^3C, & B \to b^3, & B \to D, & C \to A, \\
& C \to aab, & D \to a, & D \to aab, & D \to abD, & D \to ab\}.
\end{array}
$$

Proof of Theorem 2.5 (cont.).

Next we eliminate the chain-rules from P_1.

(1) Two nonterminals $A, B \in N_1$ are called equivalent ($A \leftrightarrow B$) if $A \to^+_{P_1} B$ and $B \to^+_{P_1} A$.
For $A \in N_1$, $[A] = \{ B \in N_1 \mid A \leftrightarrow B \}$.
Pick $A' \in [A]$ and replace all $B \in [A]$ in P_1 by A'.
Remove resulting productions of the form $(A' \to A')$.

(2) Order the remaining nonterminals such that $(A \to B) \in P_1$ implies $A < B$.
If B is the largest nonterminal occurring on the right-hand side of a chain-rule $A \to B$, then delete this chain-rule and add productions $A \to \alpha$ for all productions $B \to \alpha$.

(3) Repeat (2) until no chain-rules are left.

Example (cont.):

P_1 contains the chain-rules

$$\hat{S} \rightarrow B,\ S \rightarrow B,\ A \rightarrow B,\ B \rightarrow C,\ B \rightarrow D, \text{ and } C \rightarrow A.$$

Hence, $A \leftrightarrow B \leftrightarrow C$ and $[A] = \{A, B, C\}$.

We pick A as representative and replace B and C by A:

$$\begin{array}{lllllll} P_2 & = \{\hat{S} \rightarrow \varepsilon, & \hat{S} \rightarrow abA, & \hat{S} \rightarrow ab, & \hat{S} \rightarrow A, & S \rightarrow abA, \\ & S \rightarrow ab, & S \rightarrow A, & A \rightarrow abS, & A \rightarrow ab, & A \rightarrow A, \\ & A \rightarrow A, & A \rightarrow b^3A, & A \rightarrow b^3, & A \rightarrow D, & A \rightarrow A, \\ & A \rightarrow aab, & D \rightarrow a, & D \rightarrow aab, & D \rightarrow abD, & D \rightarrow ab\}. \end{array}$$

To eliminate the chain-rule $A \rightarrow D$,
add the productions $A \rightarrow a$ and $A \rightarrow abD$.

To eliminate the chain-rules $\hat{S} \rightarrow A$ and $S \rightarrow A$, add corresponding productions with left-hand sides $\hat{S}$ and S.

Example (cont.):

This process yields $G_3 = (N_3, T, \hat{S}, P_3)$ with $N_3 = \{\hat{S}, S, A, D\}$ and

$$\begin{aligned} P_3 \;\; = \{ & \hat{S} \to \varepsilon \mid abA \mid ab \mid abS \mid b^3A \mid b^3 \mid aab \mid a \mid abD, \\ & S \to abA \mid ab \mid abS \mid b^3A \mid b^3 \mid aab \mid a \mid abD, \\ & A \to abS \mid ab \mid b^3A \mid b^3 \mid aab \mid a \mid abD, \\ & D \to a \mid aab \mid abD \mid ab \}. \end{aligned}$$

Proof of Theorem 2.5 (cont.).

Finally we eliminate long productions from P_3.

For each production $(A \to a_1 a_2 \cdots a_m B) \in P_3$, where $m \geq 2$, introduce new nonterminals $A_1, A_2, \ldots, A_{m-1}$, and replace the above production by
$A \to a_1 A_1, A_1 \to a_2 A_2, \ldots, A_{m-2} \to a_{m-1} A_{m-1}, A_{m-1} \to a_m B$.

For each production $(A \to a_1 a_2 \cdots a_m) \in P_3$, where $m \geq 2$, introduce new nonterminals $A_1, A_2, \ldots, A_{m-1}$, and replace the above production by
$A \to a_1 A_1, A_1 \to a_2 A_2, \ldots, A_{m-2} \to a_{m-1} A_{m-1}, A_{m-1} \to a_m$.

The resulting grammar $\hat{G}$ is in right normal form and $L(\hat{G}) = L(G)$. □

Example (cont.):

P_3 contains the following A-productions:

$A \to abS$, $A \to b^3A$, $A \to b^3$, $A \to aab$, $A \to ab$, $A \to a$, and $A \to abD$.

To replace the long ones, we introduce the nonterminals $A_1, A_2, \ldots, A_9$ and the following productions:

$$\begin{array}{lllll} A \to aA_1, & A_1 \to bS, & A \to bA_2, & A_2 \to bA_3, & A_3 \to bA, \\ A \to bA_4, & A_4 \to bA_5, & A_5 \to b, & A \to aA_6, & A_6 \to aA_7, \\ A_7 \to b, & A \to aA_8, & A_8 \to b, & A \to aA_9, & A_9 \to bD. \end{array}$$

For example:

$$\begin{array}{lllllll} A & \to & aA_1 & \to & abS, & & \\ A & \to & bA_2 & \to & bbA_3 & \to & bbbA, \\ A & \to & aA_6 & \to & aaA_7 & \to & aab. \end{array}$$

The long $\hat{S}$- and S-productions are replaced analogously. □