Models of Computation

1: Basics, Languages

Basics, terminology

- **Alphabet**: a finite, non-empty set of symbols/letters.
- **Words** or **strings** over *V*: Finite sequences of the elements of an alphabet *V*.
- V^* : the **set of words** over *V* including the **empty word** (ε).
- $V^+ = V^* \setminus \{\varepsilon\}$: the **set of non-empty words** over *V.*
- The **length** of a word $u = t_1 \ldots t_n$ is the numer of letters in *u,* denoted by $|u| = n$.
  - Length of the empty set ε is 0 ($|\varepsilon| = 0$).

- Example:
  Let $V = \{a, b\}$, then ab and *baaabb* are words over *V.*

Basics, terminology

- Let V be an alphabet and let u and v be words over V (i.e., $u,v \in V^*$). Then the word uv is the **concatenation** of u and v.

- $|uv| = |u| + |v|$.

- Example:
  Let $V = \{a, b\}$, $u = ab$ and $v = baabb$ words over V.
  Then $uv = abbaabb$.

Basics, terminology

Properties

- The concatenation is associative, but in general not commutative.
  - if $u, v \in V^*$, $u \neq v$, then uv differs from vu, unless V consists of only one letter (not commutative).
  - if $u, v, w \in V^*$, then $u(vw) = (uv)w$ (associative).
- V^* is **closed** for the operation of concatenation (i.e. for any $u, v \in V^*$, $uv \in V^*$ holds).
- The concatenation is an operation with **identity element**, or **neutral element**, the neutral element is ε (i.e., for any $u \in V^*$, $u = u\varepsilon = \varepsilon u$).

Basics, terminology

- Let i be a non-negative integer and u be a word over V ($u \in V^*$). The ***i*-th power** u^i of the word u is the concatenation of i instances of u.
- Convention: $u^0 = \varepsilon$.

- Example:
  Let $V = \{a, b\}$ and $u = abb$ be a word above V.
  Then $u^0 = \varepsilon$, $u^1 = \text{abb}$, $u^2 = abbabb$, $u^3 = abbabbabb$, ...

Basics, terminology

- Let u and v be words over V. The words u and v are **equa**l, if as sequences of letters, they are equal element-by-element, i.e., $|u|=|v|$ and for all $i = 1,\ldots,|u|$, the i-th letter of u and the i-th letter of v are equal.
- Let V be an alphabet and u and v be words over V.
  The word u is a **subword** (or **substring**) of v,
  if $v = xuy$, for some $x, y \in V^*$.
- A word u is a **proper subword** (or **proper substring**) of a word v if at least one of x or y is not empty, i.e. if $xy \neq \varepsilon$.
- If $x = \varepsilon$, then u is the **prefix** of v.
- If $y = \varepsilon$, then u is the **suffix** of v.

Basics, terminology

- Example:
  Let $V = \{a, b\}$ and $u = abb$.
  - Subwords of u: $\varepsilon, a, b, ab, bb, abb$.
  - Proper subwords of u: $\varepsilon, a, b, ab, bb$.
  - Prefixes of u: ε, a, ab, abb.
  - Suffixes of u: ε, b, bb, abb.

Basics, terminology

- Let u be a word over the alphabet V. The **reverse** (or **mirror**) word u^{-1} of u is the word obtained, s.t. the letters of u are written in reverse order.
- Let $u = a_1 \dots a_n$, $a_i \in V$, $1 \le i \le n$. Then $u^{-1} = a_n \dots a_1$.
- $(u^{-1})^{-1} = u$.
- $(u^{-1})^i = (u^i)^{-1}$ also holds, where $i = 1, 2, \dots$

- Example:
  Let $V = \{a, b\}$ and $u = abba$ and $v = aabbba$
  Then $u^{-1} = abba$ (palindrome) and $v^{-1} = abbbaa$.

Basics, terminology

- Let V be an alphabet and L be an arbitrary subset of V^*. L is called a **language** over V.
- An **empty language** (a language that does not contain any words) is denoted by $\emptyset$.
- A language L over V is a **finite language** if it has a finite number of words. Otherwise, L is an **infinite language**.

- Example:
  Let $V = \{a, b\}$ be an alphabet.

  $L_1 = \{a, b, \varepsilon\}$.

  $L_2 = \{a^i b^i \mid i \geq 0\}$.

  $L_3 = \{uu^{-1} \mid u \in V^*\}$.

  $L_4 = \{(a^n)^2 \mid n \geq 1\}$.

  $L_5 = \{u \mid u \in \{a, b\}^+ , N_a(u) = N_b(u)\}$, where $N_a(u)$ and $N_b(u)$ denote the number of occurrences of symbols a and b in u, respectively.

  L_1 is a finite language, the others are infinite.

Basics, terminology

- A **generative grammar** G is a 4-tuple (N, T, P, S), where
  - N and T are disjoint finite alphabets (i.e. $N \cap T = \varnothing$).
  - The elements of N are called **nonterminal** symbols.
  - The elements of T are called **terminal** symbols.
  - $S \in N$ is the **start symbol** (axiom).
  - P is a finite set of ordered (x,y) pairs, where $x,y \in (N \cup T)^*$ and x contains at least one non-terminal symbol.
  - The elements of P are called **rewriting rules** (**rules** for short) or **productions**. $x \to y$ can be used instead of (x,y), where $\to \notin (N \cup T)$.

Basics, terminology

- Example:
  - $G_1 = (\{S, A, B\}, \{a, b, c\}, \{S \to c, S \to AB, A \to aA, B \to \varepsilon, abb \to aSb\}, S)$ is not a generative grammar.

  - $G_2 = (\{S, A, B, C\}, \{a, b, c\}, \{S \to a, S \to AB, A \to Ab, B \to \varepsilon, aCA \to aSc\}, S)$ is a generative grammar.

Basics, terminology

- Let $G = (N, T, P, S)$ be a generative grammar and let $u, v \in (N \cup T)^*$. The word v can be **derived directly** or **in one step** from u in G, denoted as $u \Rightarrow_G v$,
  if $u = u_1 x u_2$ and $v = u_1 y u_2$, where $u_1, u_2 \in (N \cup T)^*$ and $x \to y \in P$.

- Let $G = (N, T, P, S)$ be a generative grammar and $u, v \in (N \cup T)^*$. The word v can be **derived** from u in G , denoted as $u \Rightarrow^*_G v$,
  - if $u = v$, or
  - there exists a word $z \in (N \cup T)^*$, for which $u \Rightarrow^*_G z$ and $z \Rightarrow_G y$.
  - $\Rightarrow^*$ is the reflexive, transitive closure of $\Rightarrow$.
  - $\Rightarrow^+$ is the transitive closure of $\Rightarrow$.

Basics, terminology

- Let $G = (N, T, P, S)$ be a generative grammar and $u, v \in (N \cup T)^*$.
  The word v can be **derived in *k* steps** from u in G, $k \geq 1$, if there exists a sequence of words $u_1, \ldots, u_{k+1} \in (N \cup T)^*$, s.t. $u=u_1$, $v=u_{k+1}$, and $u_i \Rightarrow_G u_{i+1}$, $1 \leq i \leq k$.
- A word v can be **derived** from a word u in G if either $u = v$, or there is a number $k \geq 1$, s.t. v can be derived from u in k steps.

Basics, terminology

- Let $G = (N, T, P, S)$ be an arbitrary generative grammar. The **generated language** $L(G)$ by the grammar G is: $L(G) = \{w \mid S \Rightarrow^*_G w, w \in T^*\}$

- This means that $L(G)$ consists of words that are in T^* and can be derived from S by grammar G.

Basics, terminology

- Example:
  Let $G = (N, T, P, S)$ be a generative grammar, where
  $N = \{S, A, B\}$, $T = \{a, b\}$ and
  $P = \{S \rightarrow aSb, S \rightarrow ab, S \rightarrow ba\}$.
  Then $L(G) = \{a^n abb^n , a^n bab^n \mid n \geq 0\}$.

- Example:
  Let $G = (N, T , P, S)$ be a generative grammar, where
  $N = \{S, X , Y\}$, $T = \{a, b, c\}$ and
  $P = \{S \rightarrow abc, S \rightarrow aXbc, Xb \rightarrow bX , Xc \rightarrow Ybcc, bY \rightarrow Yb,$
  $aY \rightarrow aaX , aY \rightarrow aa\}$.
  Then $L(G) = \{a^n b^n c^n \mid n \geq 1\}$.

Basics, terminology

- Each grammar generates a language, but the same language can be generated by several different grammars.

- Two grammars are **equivalent** if they generate the same language.
- Two languages are **weakly equivalent**, if they differ only in the empty word.

Chomsky hierarchy

- Let $G = (N, T, P, S)$ be a generative grammar. G is generative grammar is of i-type, $i = 0, 1, 2, 3$, if the rule set P satisfies the following:
  - $i = 0$: no restriction.
  - $i = 1$: All rules of P have the form $u_1Au_2 \to u_1vu_2$, where $u_1, u_2, v \in (N \cup T)^*$, $A \in N$, and $v \neq \varepsilon$, except for a rule $S \to \varepsilon$, when such a rule exists in P.
    If P contains the rule $S \to \varepsilon$, then S does not occur on the right side of any rule.
  - $i = 2$: All rules of P are of the form $A \to v$, where $A \in N$ and $v \in (N \cup T)^*$.
  - $i = 3$: All rules of P are of the form either $A \to uB$ or $A \to u$, where $A, B \in N$ and $u \in T^*$.

Chomsky hierarchy

- A language L is of **type** i, where $i = 0, 1, 2, 3$, if it can be generated by a type i grammar.
- $\mathcal{L}_i$, $i = 0, 1, 2, 3$, denotes the class (family) of type i languages.

Chomsky hierarchy

- Type 0 grammars are called **phrase-structured** grammars.
- Type 1 grammars are **context-sensitive** grammars, since some occurrence of the nonterminal *A* can only be substituted with the word *v* in the presence of contexts u_1 and u_2.
- Type 2 grammars are **context-free** grammars, because the substitution of a nonterminal *A* with *v* is allowed in any context.
- Type 3 grammars are **regular** or **finite state** grammars.

- The classes of languages of type 0,1,2,3 are called **recursively enumerable**, **context-sensitive**, **context-free**, and **regular**, respectively.

Chomsky hierarchy

Linguistic background

”The cunning fox hastily ate the leaping frog.”

- $S \rightarrow A + B$ (S: sentence, A: noun phrase, B: verb phrase)
- $A \rightarrow C + D + E$ (C : article, D: adjective, E : noun)
- $B \rightarrow G + B$ (G : adverb)
- $B \rightarrow F + A$ (F : verb)
- $C \rightarrow$ the
- $D \rightarrow$ cunning
- $E \rightarrow$ fox
- $G \rightarrow$ hastily
- $F \rightarrow$ ate
- $D \rightarrow$ leaping
- $E \rightarrow$ frog

Chomsky hierarchy

Linguistic background

- + (space) – terminal symbol
- cunning ← → leaping , fox ← → frog (they are interchangeable, but the meanings are different)
- Sentence is syntactically correct

- It is not possible to describe the complete syntax of natural languages

Chomsky hierarchy

- It is oblivious that $\mathcal{L}_3 \subseteq \mathcal{L}_2 \subseteq \mathcal{L}_0$ and $\mathcal{L}_1 \subseteq \mathcal{L}_0$.
- It can also be shown that (Chomsky's hierarchy) following hold: $\mathcal{L}_3 \subset \mathcal{L}_2 \subset \mathcal{L}_1 \subset \mathcal{L}_0$.
- The inclusion relation between language class $\mathcal{L}_2$ and $\mathcal{L}_1$ is not is oblivious from the definition of the corresponding grammars. However, $\mathcal{L}_1$ can be also generated by so called length-non-decreasing grammars. For all rules $p \to q$ of a length-non-decreasing grammar, $|p| \leq |q|$ is fulfilled, except $S \to \varepsilon$. If $S \to \varepsilon \in P$, then S does not occur in the right side of any rule of P.

Operations on Languages

- Let V be an alphabet and L_1, L_2 be languages over V (that is, $L_1 \subseteq V^*$ and $L_2 \subseteq V^*$)
  - **union:** $L_1 \cup L_2 = \{u \mid u \in L_1 \text{ or } u \in L_2\}$.
  - **intersection:** $L_1 \cap L_2 = \{u \mid u \in L_1 \text{ and } u \in L_2\}$.
  - **difference:** $L_1 - L_2 = \{u \mid u \in L_1 \text{ and } u \notin L_2\}$.

- Example:

  Let $V = \{a, b\}$ be an alphabet and $L_1 = \{a, b\}$ and $L_2 = \{\varepsilon, a, bbb\}$ languages over V. Then

  $L_1 \cup L_2 = \{\varepsilon, a, b, bbb\}$

  $L_1 \cap L_2 = \{a\}$

  $L_1 - L_2 = \{b\}$

Operations on Languages

- The **complement** of the language $L \subseteq V^*$ with respect to the alphabet V is the language $\overline{L} = V^* - L$.

- Example:
  Let $V = \{a\}$ be an alphabet and let $L = \{a^{4n} \mid n \geq 0\}$. Then $\overline{L} = V^* - \{a^{4n} \mid n \geq 0\}$.

Operations on Languages

- Let V be an alphabet and L_1, L_2 be languages over V (i.e. $L_1 \subseteq V^*$ and $L_2 \subseteq V^*$). The **concatenation** of L_1 and L_2 is $L_1L_2 = \{u_1u_2 \mid u_1 \in L_1, u_2 \in L_2\}$.

- Remark:
  The following equalities hold for every language L:
  $\emptyset L = L\emptyset = \emptyset$ and
  $\{\varepsilon\}L = L\{\varepsilon\} = L$.

Operations on Languages

- L^i denotes the ***i*-th iteration** of L (for the operation of concatenation), where $i \geq 1$. By convention, $L^0 = \{\varepsilon\}$.
- The **iterative closure** (or **Kleene closure**) of a language L is: $L^* = \bigcup_{i\geq 0} L^i$.
- The **positive closure** of L is: $L^+ = \bigcup_{i\geq 1} L^i$.

- Remark:
  Obviously, if $\varepsilon \in L$, then $L^+ = L^*$. Otherwise, $L^+ = L^* - \{\varepsilon\}$.

Operations on Languages

- Example (concatenation):
  Let $V = \{a, b\}$ and let
  $L_1 = \{a, b\}$, $L_2 = \{\varepsilon, a, bbb\}$,
  $L_3 = \{a^{4n}b^{4n} \mid n \geq 0\}$ and $L_4 = \{a^{7n}b^{7n} \mid n \geq 0\}$.
  Then
  - $L_1L_2 = \{a, b, aa, ba, abbb, bbbb\}$,
  - $L_3L_4 = \{a^{4n}b^{4n}a^{7m}b^{7m} \mid n \geq 0, m \geq 0\}$.

Operations on Languages

- Let V be an alphabet and $L \subseteq V^*$. Then the language $L^{-1} = \{u^{-1} \mid u \in L\}$ is the **mirror** (or **reversal**) of L.

- Remarks:
  - $(L^{-1})^{-1} = L$,
  - $(L_1 L_2 \ldots L_n)^{-1} = L_n^{-1} \ldots L_2^{-1} L_1^{-1}$,
  - $(L^i)^{-1} = (L^{-1})^i$, where $i \geq 0$, and
  - $(L^*)^{-1} = (L^{-1})^*$.

Operations on Languages

- Example (mirror, reversal):
  Let $V = \{a, b\}$ and $L = \{\varepsilon, a, abb\}$ be a language over V. Then
  $L^{-1} = \{\varepsilon, a, bba\}$.

Operations on Languages

- The **prefix of a language** $L \subseteq V^*$ is the language
  $\mathrm{PRE}(L) = \{ u \mid u \in V^* , uv \in L$ for some $v \in V^* \}$.

- Remark:
  By definition, $L \subseteq \mathrm{PRE}(L)$ for any language $L \in V^*$.

- The **suffix of a language** $L \subseteq V^*$ is the language
  $\mathrm{SUF}(L) = \{ u \mid u \in V^* , vu \in L$ for some $v \in V^* \}$.

Operations on Languages

- Let V_1 and V_2 be two alphabets. The mapping $h : V_1^* \rightarrow V_2^*$ is called a **homomorphism** if the following conditions hold:
  - for every word $u \in V_1^*$ there is exactly one word $v \in V_2^*$ for which $h(u) = v$.
  - $h(uv) = h(u)h(v)$, for all $u, v \in V_1^*$.

- Remarks:
  - It follows from the above conitions that $h(\varepsilon) = \varepsilon$. Namely, for all $u \in V_1^*$ holds $h(u) = h(\varepsilon u) = h(u\varepsilon)$.
  - For all words $u = a_1a_2 \ldots a_n$, $a_i \in V_1$, $1 \leq i \leq n$, it holds that $h(u) = h(a_1)h(a_2) \ldots h(a_n)$. I.e. it is sufficient to define the mapping h on the elements of V_1, this is automatically extended to V_1^*.

Operations on Languages

- A homomorphism $h : V_1^* \to V_2^*$ is **ε-free** if for all $u \in V_1^+$, $h(u) \neq \varepsilon$.

- Let $h : V_1^* \to V_2^*$ be a homomorphism. The ***h*-homomorphic image** of a language $L \in V_1^*$ is the language $h(L) = \{w \in V_2^* \mid w = h(u), u \in L\}$

- Example (homomorphism): Let $V_1 = V_2 = \{a, b\}$ be two alphabets. Let $h : V_1^* \to V_2^*$ be a homomorphism, s.t. $h(a) = bbb$, $h(b) = ab$ and $L = \{a, abba\}$. Then $h(L) = \{bbb, bbbababbbb\}$.

Operations on Languages

- A homomorphism h is called an **isomorphism**
  if following holds:
  for any $u, v \in V_1^*$, if $h(u) = h(v)$, then $u = v$.

- Example (isomorphism – binary representation of decimal numbers):
  $V_1 = \{0, 1, 2, \ldots, 9\}$, $V_2 = \{0, 1\}$,
  $h(0) = 0000$, $h(1) = 0001$, . . . , $h(9) = 1001$

Literature

- Handbook of Formal Languages, G. Rozenberg, A. Salomaa, (eds.), Springer–Verlag, Berlin–Heidelberg, 1997.
- Gy. E. Révész, Introduction to Formal Languages, Dover Publications, Inc., New York, 2012.
- G. Rozenberg, A. Salomaa, The mathematical theory of L systems, Vol. 90., Academic Press, 1980.
- J. Dassow, Gh. Paun. Regulated rewriting in formal language theory, Springer Publishing Company, Inc., 2012.