A CONSTRUCTION AND REPRESENTATION OF SOME VARIABLE LENGTH CODES

Nacer Ghadbane

Laboratory of Pure and Applied Mathematics, Department of Mathematics, University of M'sila, Algeria

Corresponding author: Nacer Ghadbane, nacer.ghadbane@yahoo.com

ABSTRACT: Let Σ be an alphabet. A subset X of the free monoid Σ^* is a code over Σ if for all $m, n \geq 1$ and $x_1, \dots, x_n, y_1, \dots y_m \in X$, the condition : $x_1 \dots x_n = y_1 \dots y_m$ implies $n = m$ and $x_i = y_i$ for $i = 1, \dots, n$. In other words, a set X is a code if any word in X^+ can be written uniquely as a product of words in X ([BP84]). It is not always easy to verify a given set of words is a code. In this paper, we give the construction and representation by deterministic finite automata of some variable length codes.

KEYWORDS: Words and languages, the free monoid and relatives, morphisme of monoids, deterministic finite automata.

1. INTRODUCTION

¶

Let Σ^* be the free monoid generated by a finite alphabet Σ. A langage $X \subseteq \Sigma^*$ is called a Variable Length Codes if X^* is a free submonoid of Σ^* with base X. The theory of variable length codes takes its origin in the framework of the theory of information, since Shannon's early works in the 1950's. An algebraic theory of codes was subsequently initiated by M. P. Schutzenberger ([Sch55]). Variable length codes occur frequently in the domain of data compression.

Let M be a submonoid of Σ^* and X be its minimal generating, then M is free iff any equality $x_1 \dots x_n = y_1 \dots y_m$, $n, m \geq 1$ $x_i, y_i \in X$ implies $n = m$ and $x_i = y_i, 1 \leq i \leq n.$

The minimal generating of a free submonoid M of Σ^* is called a variable length code.

The remainder of this paper is organized as follows. In Section 2, some mathematical preliminaries. In Section 3, we use the algorithm of Sardinas and Patterson ([SP53]), to giving some test for variable length codes. In Section 4, we give the representation by deterministic finite automata of some variable length codes. Finally, we draw our conclusions in Section 5.

2. PRELIMINARIES

A semigroup is a set S together with an associative binary operation $\circ$ defined on it. We shall write $(S,\circ)$ or simply S for a semigroup. If $s \circ t = t \circ s$ holds for all $s, t \in S$, we call a commutative semigroup. If the semigroup $(S,\circ)$ has an identity element, then $(S,\circ)$ is called a monoid. If $X \subseteq S$, we write X^* for the submonoid of M generated by X, that is the set of finite products $x_1 \dots x_n$ with $x_1, \dots, x_n \in X$, including the empty product 1. It is the smallest submonoid of S containing X. For example, $(\mathbb{N}, +)$ is generated by $\{1\}$, while $(\mathrm{N},\times)$ is generated by $\{1\} \cup \mathrm{P}$, where P is the set of all primes.

Let S and T be semigroups. A function $h: S \longrightarrow T$ is called a homomorphism if $h(s_1 s_2) = h(s_1)h(s_2)$ for all $s_1, s_2 \in S$.

If S and T are both monoids then we usually require in addition that the identity of S is mapped to the identity of T

Let Σ be a set, with we call an alphabet. A word w on the alphabet Σ is a finite sequence of elements of Σ, $w = (a_1, a_2, \dots, a_n)$ $a_i \in \Sigma$, $1 \leq i \leq n.$

The set of all words on the alphabet Σ is denoted by Σ^* and is equipped with the associative operation defined by the concatenation of two sequences

$$(a_1, a_2, \dots, a_n)(b_1, b_2, \dots, b_m) = (a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_m)$$

This operation is associative. This allows us to write $w = a_1 a_2 \dots a_n$. The string consisting of zero letters is called the empty word, written ε. Thus, ε, 0, 1, 011, 1111 are words over the alphabet $\{0, 1\}$. The set Σ^* of words is equipped with the structure of a monoid. The monoid Σ^* is called the free monoid on Σ. The reverse of a word $w = a_1 a_2 \dots a_n$, is $w^{-1} = a_n a_{n-1} \dots, a_1.$

Note that for all $u, v \in \Sigma^*, (uv)^{-1} = v^1 u^{-1}.$

The length of a word u, in symbols $|u|$, is the number of letters in u when each letter is counted as many times as it occurs. Again by definition, |ε|=0.

The length function possesses some of the formal properties of Logarithm: $|uv| = |u| + |v|, |u^i| = i|u|$, for any words u and v and integers $i \geq 0$. For example $|011| = 3$ and $|1111| = 4$. For a subset

B of Σ, we let $|w|_B$ denote the number of letters of w which are in B. Thus $|w| = \sum_{a\in\Sigma}|w|_a$.
A language L over Σ^* is any subset of Σ^*.
Let $K, L \subseteq \Sigma^*$. A equation of the form $X = KX + L$, where $\varepsilon \notin K$, has a unique solution given by $X = K^*L$.
For $X, Y \subseteq \Sigma^*$, the set $XY = \{xy, x \in X, y \in Y\}$. In particular, we define $X^0 = \{\varepsilon\}, X^{n+1} = X^nX$ $(n \geq 0)$.
Given a set $X \subseteq \Sigma^*$, the star of X is as in any monoid, the set $X^* = \{x_1 \dots x_n,\ x_i \in X\} = \bigcup_{n\geq 0} X^n$.
Any submonoid M of Σ^* has a unique minimal generating $X = (M - \varepsilon) - (M - \varepsilon)^2$.
For $x, y \in \Sigma^*$, we define $x^{-1}y = \{z \in \Sigma^* : zx = y\}$ and $xy^{-1} = \{z \in \Sigma^* : zy = x\}$.
For subsets X, Y of Σ^*, this notation is extended to $X^{-1}Y = \bigcup_{x\in X}\bigcup_{y\in Y} x^{-1}y$ and $XY^{-1} = \bigcup_{x\in X}\bigcup_{y\in Y} xy^{-1}$.
A mapping $h: \Sigma^* \longrightarrow \Delta^*$, where Σ and Δ are alphabets, satisfying the condition $h(uv) = h(u)h(v)$, for all words u and v of Σ^* is called a morphism, define a morphism h, it suffices to list all the words $h(\sigma)$, where a ranges over all the (finitely many) letters of Σ. If M is a monoid, then any mapping $f: \Sigma \to M$ extends to a unique morphism $\hat{f}: \Sigma^* \longrightarrow M$ For instance, if M is the additive monoid $\mathbb{N}$, and f is defined by $f(\sigma) = 1$ for each $a \in \Sigma$, then $\hat{f}(u)$ is the length $|u|$ of the word u.
The theory of codes provides some jewels of combinatorics on words.
A subset X of the free monoid Σ^* is a code over Σ if for all $m, n \geq 1$ and $x_1, \dots, x_n, y_1, \dots y_m \in X$, the condition : $x_1 \dots x_n = y_1 \dots y_m$ implies $n = m$ and $x_i = y_i$ for $i = 1, \dots, n$. In other words, a set X is a code if any word in X^+ can be written uniquely as a product of words in X.
The words of X are called code words, the elements of X^* are messages. It is not always easy to verify that a given set of words is a code.
Deterministic finite automaton is a type of finite automaton in which the transitions are deterministic, in the sense that there will be exactly one transition from a state on an input symbol. Formally, a deterministic finite automata (DFA) is a quintuple $A = (Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set called the set of states,
- Σ is a finite set called the input alphabet,
- $q_0 \in Q$, called the initial state,
- $F \subseteq Q$, called the set of final states, and
- $\delta: Q \times \Sigma \longrightarrow Q$ is a function called the transition function.

Recall that the transition function δ assigns a state for each state and an input symbol.
This naturally can be extended to all strings in Σ^*, i. e. assigning a state for each state and an input string.
The extended transition function $\delta: Q \times \Sigma^* \longrightarrow Q$ is defined recursively as follows:
For all $q \in Q$, $w \in \Sigma^*$ and $a \in \Sigma$, $\delta(q, \varepsilon) = \varepsilon$ and $\delta(q, wa) = \delta(\delta(q, w), a)$.
The langage accepted by A is $L(A) = \{w \in \Sigma^* : \delta(q_0, w) \in F\}$

3. TEST FOR VARIABLE LENGTH CODES

The basic question to be asked is "When is a given X of Σ^* a variable length code?". This was answered by Sardinas and Patterson ([SP53]). Define recursively subsets U_n of Σ^* as follows:

$$\begin{cases} U_0 = X^{-1}X - \{\varepsilon\} \\ U_{n+1} = {U_n}^{-1}X \cup X^{-1}U_n, \quad for\ n \geq 0 \end{cases}$$

where ε denotes the identity of Σ^* and $X^{-1}X = \bigcup_{x\in X} x^{-1}X$. We have:
If $\varepsilon \in U_n$, then X is not a variable length code.
If $U_{n+1} = U_n$, then X is a variable length code.

Example 3.1

Let $\Sigma = \{0,1\}$ and $X = \{00,01,10,11\}$ we have, $U_0 = X^{-1}X - \{\varepsilon\}$, $X^{-1}X = \bigcup_{x\in X} x^{-1}X$, where $x^{-1}X = \{y \in \{0,1\}^* : xy \in X\}$.

- $(00)^{-1}X = \{y \in \{0,1\}^* : (00)y \in X\} = \{\varepsilon\}$
- $(01)^{-1}X = \{y \in \{0,1\}^* : (01)y \in X\} = \{\varepsilon\}$
- $(10)^{-1}X = \{y \in \{0,1\}^* : (10)y \in X\} = \{\varepsilon\}$
- $(11)^{-1}X = \{y \in \{0,1\}^* : (11)y \in X\} = \{\varepsilon\}$

Then $U_0 = X^{-1}X - \{\varepsilon\} = \emptyset$, with $\emptyset$ designates the empty set. $U_1 = {U_0}^{-1}X \cup X^{-1}U_0$, ${U_0}^{-1}X = \bigcup_{u_0\in U_0} u_0^{-1}X$ and $X^{-1}U_0 = \bigcup_{x\in X} x^{-1}U_0$.
We have $U_1 = U_0 = \emptyset$, Finally, the set X is a variable length code.

Example 3.2

Let $\Sigma = \{0,1\}$ and $X = \{00,010,101,11\}$ we have, $U_0 = X^{-1}X - \{\varepsilon\}$, $X^{-1}X = \bigcup_{x\in X} x^{-1}X$, where $x^{-1}X = \{y \in \{0,1\}^* : xy \in X\}$.

- $(00)^{-1}X = \{y \in \{0,1\}^* : (00)y \in X\} = \{\varepsilon\}$
- $(010)^{-1}X = \{y \in \{0,1\}^* : (010)y \in X\} = \{\varepsilon\}$
- $(101)^{-1}X = \{y \in \{0,1\}^* : (101)y \in X\} = \{\varepsilon\}$
- $(11)^{-1}X = \{y \in \{0,1\}^* : (11)y \in X\} = \{\varepsilon\}$

Then $U_0 = X^{-1}X - \{\varepsilon\} = \emptyset$, with $\emptyset$ designates the empty set. $U_1 = {U_0}^{-1}X \cup X^{-1}U_0$, ${U_0}^{-1}X = \bigcup_{u_0\in U_0} u_0^{-1}X$ and $X^{-1}U_0 = \bigcup_{x\in X} x^{-1}U_0$.
We have $U_1 = U_0 = \emptyset$, Finally, the set X is a variable length code.

Example 3.3

Consider the alphabet $\Sigma = \{0,1\}$ and $X = \{00,01,0111,100\}$ we have, $U_0 = X^{-1}X - \{\varepsilon\}$, $X^{-1}X = \bigcup_{X \in x} x^{-1} X$, where $x^{-1}X = \{y \in$ 0,1∗:xy∈X.

- $(00)^{-1}X = \{y \in \{0,1\}^*: (00)y \in X\} = \{\varepsilon\}$
- $(01)^{-1}X = \{y \in \{0,1\}^*: (01)y \in X\} = \{\varepsilon, 1\}$
- $(011)^{-1}X = \{y \in \{0,1\}^*: (011)y \in X\} = \{\varepsilon\}$
- $(100)^{-1}X = \{y \in \{0,1\}^*: (100)y \in X\} = \{\varepsilon\}$

Then $U_0 = X^{-1}X - \{\varepsilon\} = \{1\}$.
$U_1 = {U_0}^{-1}X \cup X^{-1}U_0$, ${U_0}^{-1}X = \bigcup_{u_0 \in U_0} u_0^{-1} X$ and $X^{-1}U_0 = \bigcup_{x \in X} x^{-1} U_0$.
We have ${U_0}^{-1}X = \bigcup_{u_0 \in U_0} u_0^{-1} X = (1)^{-1}X$, $(1)^{-1}X = \{y \in \{0,1\}^*: (1)y \in X\} = \{00\}$.
$X^{-1}U_0 = \bigcup_{x \in X} x^{-1} U_0$, there is only cases to be considered:
$(00)^{-1}U_0 = \{y \in \{0,1\}^*: (00)y = 1\} = \emptyset$
$(01)^{-1}U_0 = \{y \in \{0,1\}^*: (01)y = 1\} = \emptyset$
$(011)^{-1}U_0 = \{y \in \{0,1\}^*: (011)y = 1\} = \emptyset$
$(100)^{-1}U_0 = \{y \in \{0,1\}^*: (100)y = 1\} = \emptyset$
Then $U_1 = {U_0}^{-1}X \cup X^{-1}U_0 = \{00\}$.
$U_2 = {U_1}^{-1}X \cup X^{-1}U_1$, we have
${U_1}^{-1}X = (00)^{-1}X = \{y \in \{0,1\}^*: (00)y \in X\}$
$= \{\varepsilon\}$.
Since U_2 contains the empty word ε, then X is not a variable length code.

4. REPRESENTATION OF VARIABLE LENGTH CODES

Let Σ be an alphabet and X be a variable length code over Σ. Our representation of X is based on the automaton theory. This subject was reviewed in [BP84], [Tsu01]. We construct an automaton for X by union of automata of codewords.
If codeword $w = x_1 \ldots x_n$ then automaton $A(w)$ of w is $A(w) = (Q_w, \Sigma, \delta_w, q_i, F_w)$ where :

- $Q_w = \{q_i, q_{x_1}, q_{x_1x_2}, \ldots, q_{x_1x_2\ldots x_{n-1}}\}$.
- $F_w = \{q_i\}$.
- Define the function $\delta_w : Q_w \times \Sigma \longrightarrow Q_w$ by:

$\delta_w(q_i, x_1) = q_{x_1}$, $\delta_w(q_{x_1}, x_2) = q_{x_1x_2}$, …, $\delta_w(q_{x_1x_2\ldots x_{n-2}}, x_{n-1}) = q_{x_1x_2\ldots x_{n-1}}$, $\delta_w(q_{x_1x_2\ldots x_{n-1}}, x_n) = q_i$.
Thus $A(w)$ can recognize $w^* = \bigcup_{k=0}^{k=+\infty} w^k$.
The automaton $A(w_1, w_2, \ldots, w_n)$ of variable length code $X = \{w_1, w_2, \ldots, w_n\}$. We can use notation $A(X)$.
$A(X) = (Q_X, \Sigma, \delta_X, q_i, F_X)$ with :

- $Q_X = Q_{w_1} \cup Q_{w_2} \ldots \cup Q_{w_n}$.
- $F_X = \{q_i\}$.
- $\delta_X = \delta_{w_1} \cup \delta_{w_2} \ldots \cup \delta_{w_□}$.

The automaton $A(X)$ accepts $X^* = \bigcup_{k=0}^{k=+\infty} X^k$.

Example 4.1

Let $\Sigma = \{0,1\}$ and $X = \{00,01,10,11\}$ we have, $A(X) = (Q_X, \Sigma, \delta_X, q_i, F_X)$ with :

- $Q_X = \{q_i, q_0, q_1\}$.
- $F_X = \{q_i\}$.
- δ_X is given by the following table:

δ_X	0	1
q_i	q_0	q_1
q_0	q_i	q_i
q_1	q_i	q_i

We show that, the langage accepted by $A(X)$ is
$L(A(X)) = \{w \in \Sigma^*: \delta(q_i, w) \in F_X\}$
$= \{w \in \Sigma^*: \delta(q_i, w) = q_i\}$
$= X^*$
$= \{00,01,10,11\}^*$.
The characteristic equations for the states q_i, q_0 *and* q_1 respectively, are :

$$\begin{cases} x_i = 0x_0 + 1x_1 + \varepsilon \\ x_0 = (0+1)x_i \\ x_1 = (0+1)x_i \end{cases}$$

With $L(A(X)) = x_i$, we have $x_i = 0x_0 + 1x_1 + \varepsilon = 0(0+1)x_i + 1(0+1)x_i + \varepsilon$
Then $x_i = [0(0+1) + 1(0+1)]x_i + \varepsilon$.
Finally $x_i = [0(0+1) + 1(0+1)]^* = X^*$.

Example 4.2

Let $\Sigma = \{0,1\}$ and $X = \{00,010,101,11\}$ we have, $A(X) = (Q_X, \Sigma, \delta_X, q_i, F_X)$ with :

- $Q_X = \{q_i, q_0, q_1, q_{01}, q_{10}\}$.
- $F_X = \{q_i\}$.
- δ_X is given by the following table :

δ_X	0	1
q_i	q_0	q_1
q_0	q_i	q_{01}
q_1	q_{10}	q_i
q_{01}	q_i	
q_{10}		q_i

We show that, the langage accepted by $A(X)$ is
$L(A(X)) = \{w \in \Sigma^*: \delta(q_i, w) \in F_X\}$
$= \{w \in \Sigma^*: \delta(q_i, w) = q_i\}$
$= X^*$
$= \{00,010,101,11\}^*$.
The characteristic equations for the states q_i, q_0 *and* q_1 respectively, are:

$$\begin{cases} x_i = 0x_0 + 1x_1 + \varepsilon \\ x_0 = 0x_i + 1x_{01} \\ x_1 = 1x_i + 0x_{10} \\ x_{01} = 0x_i \\ x_{10} = 1x_i \end{cases}$$

With $L\big(A(X)\big) = x_i$. We have $x_0 = 0x_i + 1x_{01} = 0x_i + 10x_i$.
and $x_i = 0x_0 + 1x_1 + \varepsilon = 0(0x_i + 1x_{01}) + 1(1x_i + 0x_{10}) + \varepsilon = (00 + 010 + 101 + 11)x_i + \varepsilon$.
Finally $x_i = L\big(A(X)\big) = (00 + 010 + 101 + 11)^*$.

CONCLUSIONS

In this paper, we use the algorithm of Sardinas and Patterson, to giving some test for variable length codes and we give the representation by deterministic automata of some variable length codes.

REFERENCES

[Bed98] **N. Bedon** – *Langages reconnaissables de mots indexes par des ordinaux*, thèse de Doctorat, Université de Marne-La-Valée, 1998.

[Bil14] **M. Billaud** – *Théorie des langages*, Université de Bordeaux, 2014.

[BP84] **J. Berstel, D. Perrin** – *Theory of codes*, Academic Press, 1984.

[B+12] **J. Berstel, C. D. Felice, D. Perrin, C. Reutenaauer, G. Rindone** – *Recent results on syntactic groups of prefix codes*, European Journal of Combinatorics, vol 33, p.1386-1401, 2012.

[Car06] **O. Carton** – *Langage formels, Calculabilité et Complexité*, 2006.

[CP85] **R. Cori, D. Perrin** – *Automates et Commutattions Partielles*, RAIRO-Informatique Théorique, tome 19, n° 1, p. 21-32, 1985.

[Fal08] **J. Falucskai** – *On equivalence of two testes for codes*, Acta Mathematica Academiae Peadagogicae Nyiregyhàziensis, vol 24, p.249-256, 2008.

[FB95] **R. Floyd, R. Beigel** (Traduction de D. Krob) – *Le langage des machines*, International Thomthenn France, Paris, 1995.

[GK10] **D. Goswami, K. V. Krishna** – *Formal Languages and Automata Theory*, 2010.

[GM16] **N. Ghadbane, D. Mihoubi** – *A Construction of Some Group Codes*, International Journal of Electronics and Information Engineering, vol 4, p.50-55, 2016.

[Kli14] **I. Klimann** – *Autour de divers problèmes de décision sur les automates*, Mémoire d'habilitation à diriger des recherches, Université Paris Diderot, Sorbonne, 2014.

[Mau66] **N. Maurice** – *Eléments de la théorie général des codes*, School of Computer Science Carleton, Université de Paris, 1966.

[MS14] **A. Maheshwari, M. Smid** – *Introduction to Theory of Computation*, School of Computer Science Carleton, University Ottawa, Canada, 2014.

[Sch55] **M. P. Schutzenberger** –*Une théorie algébrique du codage*, Séminaire Dubreil, *Vol 09, 1955.*

[Shy79] **H. J. Shyi** – *Free monoids and languages*, Soochow University Taipei, Taiwan, R. O. C., 1979.

[SP53] **A. A. Sardinas, C. W. Patterson** – *A necessary and sufficient condition for the unique decomposition of codes messages*, IRE internat. Conv. Rec, vol 08, p.104-108, 1953.

[Tsu01] **K. Tsuji** – *An automaton for deciding whether a given set of words is a code*, Souririkaisekikenkyusho Kokyuroku, p. 123-127, 2001.