Coding Partitions of Regular Sets[*]

Marie-Pierre Béal[†] Fabio Burderi[‡] Antonio Restivo[‡]

Abstract

A coding partition of a set of words partitions this set into classes such that whenever a sequence, of minimal length, has two distinct factorizations, the words of these factorizations belong to the same class. The canonical coding partition is the finest coding partition that partitions the set of words in at most one unambiguous class and other classes that localize the ambiguities in the factorizations of finite sequences.

We prove that the canonical coding partition of a regular set contains a finite number of regular classes and we give an algorithm for computing this partition. From this we derive a canonical decomposition of a regular monoid into a free product of finitely many regular monoids.

1 Introduction

In this paper, we call code a set of finite words. An important class of codes is the class of Uniquely Decipherable (UD) codes. This property allows the decoding of a sequence of concatenated code words. Nevertheless, some classes of codes are used in information theory although they are not uniquely decipherable (see for instance [8], [10] and [12]). The condition of unique decipherability can also be weakened by considering that it applies only to codes with constraints (see [1]) or to codes with a constraint source (see [4], [7]). In [7], the classification of ambiguities of codes is investigated in the study of natural languages. From a combinatorial point of view, the study of ambiguities helps to understand the structure of a code.

To this purpose, the notions of coding partition and canonical coding partition of a code were introduced in [3] to study some decipherability conditions that are weaker than the unique decipherability. A coding partition is a partition of a code such that if a message $z \in X^+$ has two distinct

[*] Partially supported by Italian MURST Project of National Relevance "Linguaggi Formali e Automi: Metodi, Modelli e Applicazioni"

[†] Université Paris-Est, Laboratoire d'informatique Gaspard-Monge, CNRS, 77454 Marne-la-Vallée Cedex 2, France

[‡] Dipartimento di Matematica ed Applicazioni,Università degli studi di Palermo, Via Archirafi 34,90123 Palermo, Italy

factorizations $z = x_1x_2\cdots x_s = y_1y_2\cdots y_t$ into code words and if z is of minimal length with this property, then the code words x_i, y_j belong to a same class of the partition.

The notion of coding partition generalizes that of *UD* code: indeed *UD* codes correspond to the extremal case in which each class contains exactly one element. In general, for codes that are not *UD*, the notion of coding partition allows to recover "unique decipherability" at the level of classes of the partition. In other words, such a notion gives a tool to *localize* the ambiguities for a code that is not *UD*: indeed the ambiguities are localized inside each class of the partition and a kind of mutual unambiguity holds between the different classes.

By taking into account the natural ordering between the partition of a set X, where finer is higher, we have that the coding partitions form a complete lattice. As a consequence, given a code X, we can define the finest coding partition P of X. It is called the *characteristic* partition of X and it is denoted by $P(X)$.

The structure of $P(X)$ gives useful information about coding properties of X. In particular the extremal case in which each class of $P(X)$ is a singleton corresponds to *UD* codes. The opposite extremal case in which $P(X)$ contains only one class, and the code X contains more than one word, gives rise to the definition of *Totally Ambiguous* (*TA*) codes. Such considerations lead to define a *canonical decomposition* of a code in at most one unambiguous component and in a (possibly empty) set of *TA* components.

Remark that the notion of coding partition is related to some special cases of the notion of $\mathfrak{F}$-factorization, introduced in [9].

In [3] it is given a Sardinas-Patterson like algorithm for computing the canonical coding partition of a finite code.

In this paper, we firstly prove that the canonical coding partition of a regular code has a finite number of classes, each one being regular. This result was conjectured in [3]. We give an exponential time algorithm for computing all classes of the partition which is based on automata constructions. At last we give an algebraic setting of this result: we show, exhibiting a canonical decomposition, that a regular monoid can be expressed as a free product of at most one regular free monoid and finitely many (possibly zero) regular freely indecomposable monoids.

2 Partitions of a code

Let A be a finite alphabet. We denote by A^* the set of finite words over the alphabet A, and by A^+ the set of non-empty finite words. A *code* X is here a subset of A^+. Its elements are called *code words*, the elements of X^* *messages* .

Let X be a code and let

$$P = \{X_i \mid i \in I\}$$

be a partition of X *i.e.*: $\bigcup_{i\in I} X_i = X$ and $X_i \cap X_j = \emptyset$, for $i \neq j$.

A P-*factorization* of a message $w \in X^+$ is a factorization $w = z_1 z_2 \cdots z_t$, where

- $\forall i \;\; z_i \in X_k^+$, for some $k \geq 1$
- if $t > 1$, $z_i \in X_k^+ \Rightarrow z_{i+1} \notin X_k^+$, for all $1 \leq i \leq t-1$.

The partition P is called a *coding partition* if any element $w \in X^+$ has a *unique* P-*factorization*, *i.e.* if

$$w = z_1 z_2 \cdots z_s = u_1 u_2 \cdots u_t,$$

where $z_1 z_2 \cdots z_s$, $u_1 u_2 \cdots u_t$ are P-*factorizations* of w, then $s = t$ and $z_i = u_i$ for $i = 1, \ldots, s$.

We say that a partition P is *concatenatively independent* if, for $i \neq j$,

$$X_i^+ \cap X_j^+ = \emptyset.$$

A necessary condition for a partition P to be a coding partition, is that P is concatenatively independent moreover the trivial partition $P = \{X\}$ is always a coding partition.

Le x be a word. A *factorization* of x is a sequence of words $(x_i)_{1\leq i\leq s}$ such that $x = x_1 x_2 \cdots x_s$. Let X be a code. A *relation* is a pair of factorizations $x_1 x_2 \cdots x_s = y_1 y_2 \cdots y_t$ into code words of a same message $z \in X^+$; the relation is said non-trivial if the factorizations are distinct. In the sequel, when no confusion arises, sometimes we will denote by z both the "word" z and the *relation* $x_1 x_2 \cdots x_s = y_1 y_2 \cdots y_t$. We say that the relation $x_1 x_2 \cdots x_s = y_1 y_2 \cdots y_t$ is *prime* if for all $i < s$ and for all $j < t$ one has $x_1 x_2 \cdots x_i \neq y_1 y_2 \cdots y_j$.

In [3] is proved the following theorem.

Theorem 1. *Let $P = \{X_i \mid i \in I\}$ be a partition of a code X. The partition P is a coding partition iff for every prime relation $x_1 x_2 \cdots x_s = y_1 y_2 \cdots y_t$, the code words x_i, y_j belong to the same component of the partition.*

Example 1. We consider the code $X = \{00, 0010, 1000, 11\}$. Clearly the words $00, 0010$ and 1000 belong to a same set in the canonical coding partition since

$$001000 = 00 \cdot 1000 = 0010 \cdot 00$$

is a prime relation.

Recall that there is a natural partial order between the partitions of a set X: if P_1 and P_2 are two partitions of X, $P_1 \leq P_2$ if the elements of P_1 are unions of elements of P_2. In [3] is proved the next theorem.

Theorem 2. *The set of the coding partitions of a code X is a complete lattice.*

As a consequence of previous theorem we can give the next definition. Given a code X, *the* finest coding partition P of X is called the *characteristic* partition of X and it is denoted by $P(X)$.

A code X is called *ambiguous* if it is not *UD*. It is called *totally ambiguous* (*TA*) if $|X| > 1$ and $P(X)$ is the trivial partition: $P(X) = \{X\}$.

So *UD* codes and *TA* codes correspond to the two extremal cases: a code is *UD* if $|P(X)| = |X|$ and a code is *TA* if $|P(X)| = 1$ and $|X| > 1$.

Let X be a code and let $P(X)$ be the characteristic partition of X. Let X_0 be the union of all classes of $P(X)$ having only one element, *i.e.* of all classes $Z \in P(X)$ such that $|Z| = 1$. The code X_0 is a *UD* code and is called the *unambiguous component* of X. From $P(X)$ one then derives another partition of X

$$P_C(X) = \{X_i \mid i \geq 0\},$$

where $\{X_i \mid i \geq 1\}$ is the set of classes of $P(X)$ of size greater than 1. If there are such sets X_i with $i \geq 1$, then they are *TA* (see[3]). They are called the *TA components* of X. By Theorem 1 we have that $P_C(X)$ is a coding partition (indeed $P_C(X) \leq P(X)$) and it is called the *canonical coding partition* of X: it defines a *canonical decomposition* of a code X in at most one unambiguous component and a (possibly empty) set of *TA* components. Roughly speaking, if a code X is not *UD*, then its canonical decomposition, on one hand separates the unambiguous component of the code (if any), and, on the other, localizes the ambiguities inside the *TA* components of the code. On the contrary, if X is *UD*, then its canonical decomposition contains only the unambiguous component X_0. Moreover if X is *UD* then every partition of X is a coding partition.

In [3] is given a Sardinas-Patterson like algorithm for computing the canonical coding partition of a finite code X.

Example 2. We consider again the code $X = \{00, 0010, 1000, 11\}$. The canonical partition of X is $X_0 = \{11\}$, $X_1 = \{00, 0010, 1000\}$. Note that X is not a *UD* code.

In [3] it is proved the next result.

Theorem 3. *Given a regular code X and a partition $P = \{X_1, \ldots, X_n\}$ of X such that X_i, for $i = 1, \ldots, n$, is a regular set, it is decidable whether P is a coding partition of X.*

Moreover, again in [3], it was conjectured that *if X is regular, the number of classes of $P_C(X)$ is finite and each class of $P_C(X)$ is a regular set.*

The conjecture will be proved in the next section and, as corollary, we will extend Theorem 3 proving that is decidable whether a partition, verifying the same hypothesis of Theorem 3, is the canonical coding partition.

3 Coding partition of a regular code

In this section, we consider a regular code X.

We say that a coding partition of a code is *finite* if it has a finite number of components. We say that a coding partition of a code is *regular* if all the components of the partition are regular. The following theorem gives a positive answer to the previous conjecture.

Theorem 4. *The canonical partition of a regular code is finite and regular. Its classes can be effectively computed.*

Given a coding partition $P = \{X_i \mid i \in I\}$ of a code $X \subseteq A^+$, the condition that every word $w \in X^+$ admits a unique P-factorization has a natural algebraic interpretation in terms of free product of submonoids.

Let M be a monoid generated by submonoids $M_\lambda, \lambda \in \Lambda$. If every element of M has a unique expression of the form $m_1 m_2 \cdots m_r$ where $r \geq 0$, $\varepsilon \neq m_i \in M_{\lambda_i}$, $\lambda_i \neq \lambda_{i+1}$, then M is the free product of the M_λ's.

We say that a monoid M is *freely indecomposable* if M cannot be expressed as a free product of nontrivial monoids.

Since any submonoid of M of A^* has a unique minimal set of generators $X = (M - 1) - (M - 1)^2$, where 1 is the empty word (see [2]), we get an equivalent formulation of Theorem 4:

Theorem 5. *Any regular submonoid $M \subseteq A^*$ admits a canonical decomposition into a free product of at most one regular free submonoid and finitely many (possibly zero) regular freely indecomposable submonoids.*

As a corollary of Theorem 4, we get the following extension of Theorem 3.

Corollary 6. *Given a regular code X and a regular partition $P = \{X_1, X_2, \ldots, X_n\}$ of X, it is decidable whether P is the canonical coding parition of X.*

Note that this corollary extends the problem of decidability of the uniquely decipherability of a regular set (see [2], [6]).

In order to prove Theorem 4, we give some definitions on finite automata (see for instance [5], [11], [13, 14] for more details).

A (finite) *automaton* $\mathcal{A} = (Q, I, E, T)$ is made of a finite set of states Q, a set of edges E labelled on an alphabet A, a set of initial states I and a set of final states T. We shall also consider automata labelled in A^*. A *successful path* is a path going from a state of I to a state of T. The set of labels of successful paths is the *language accepted* (or *language recognized*) by the automaton. An automaton is *trim* if, for any state p, there is a path from an initial state to p and there is path from p to some final state.

An automaton is *unambiguous* if for any word z, any states p, q, there is at most one path going from p to q and labelled by z.

A *normalized automaton* is an unambiguous automaton $\mathcal{A} = (Q, I, E, T)$ with $I = \{0\}$, $T = \{t\}$ with $t \neq 0$, and which has no edge coming in 0 and no edge going out of t. Any regular language is accepted by an unambiguous (resp. normalized) automaton.

Let $\mathcal{A} = (Q, \{0\}, E, \{t\})$ be a normalized finite automaton that accepts the language X. Let $\mathcal{B} = (Q - \{t\}, \{0\}, F, \{0\})$ be the automaton with edges

$$F = \{(p, a, q) \mid (p, a, q) \in E \text{ and } q \neq t\} \cup \{(p, a, 0) \mid (p, a, t) \in E\}.$$

We define the automaton $\mathcal{A}^*$ as the trim part of $\mathcal{B}$. It accepts the language X^*.

Let p be a state of an automaton. We call *p-simple path* a path of the automaton $\mathcal{A}$ from p to p in which the state p is never crossed. Each non p-simple path of $\mathcal{A}$ from p to p can be uniquely decomposed into a concatenation of *p-simple paths.*

Proposition 7. *Let $\mathcal{A}$ be a normalized automaton accepting a code X. The automaton $\mathcal{A}^*$ is unambiguous if and only if X is UD. Moreover if $\mathcal{A}^*$ is ambiguous, each label of two distinct paths from 0 to 0 is a non-trivial relation in X.*

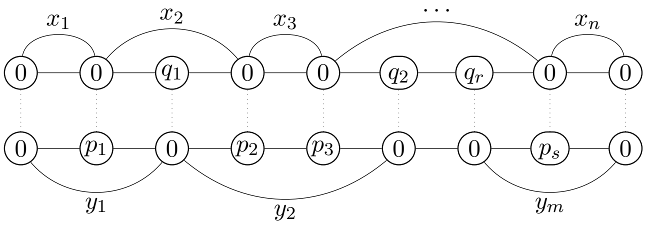
Proof. We refer [2] or [11] for the first statement. Let us show the second statement. Let us assume that $\mathcal{A}^*$ is ambiguous. Let z be the label of two distinct paths from 0 to 0. These two paths have two decompositions into 0-simple paths labelled x_i and y_j respectively, with $n, m \geq 0$ and $z = x_1 x_2 \cdots x_n = y_1 y_2 \cdots y_m$. By construction of $\mathcal{A}^*$, $x_i, y_j \in X$. Let us assume that the sequences $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ are equal. Since $\mathcal{A}$ is unambiguous, this implies that the two paths are equal, a contradiction. Hence $x_1 x_2 \cdots x_s = y_1 y_2 \cdots y_t$ is a non-trivial relation in X. □

Let $\mathcal{A} = (Q, I, E, T)$ be a finite automaton. We define the automaton $\mathcal{A} \times \mathcal{A} = (Q \times Q, I \times I, E', T \times T)$ called the *square automaton* of $\mathcal{A}$, where $E' = \{(p, q) \xrightarrow{a} (p', q') \mid p \xrightarrow{a} p' \text{ and } q \xrightarrow{a} q' \in E\}$.

The property of a trim automaton $\mathcal{A}$ of being unambiguous can be seen in the square automaton of $\mathcal{A}$ as follows: a trim automaton $\mathcal{A}$ is unambiguous if and only if the trim part of $\mathcal{A} \times \mathcal{A}$ is the diagonal of $\mathcal{A} \times \mathcal{A}$ ([13, Lemma 1.16 p. 82]).

Proposition 8. *Let $\mathcal{A}$ be a normalized automaton accepting a code X. A word z is a non-trivial relation in X if and only if z is the label of a path in $\mathcal{A}^* \times \mathcal{A}^*$ from $(0,0)$ to $(0,0)$ in which a state (p,q) with $p = 0$, $q \neq 0$ or $q = 0$, $p \neq 0$ is crossed at least one time. Furthermore, a word z is a prime relation if and only if z is the label of a $(0,0)$-simple path in $\mathcal{A}^* \times \mathcal{A}^*$ from $(0,0)$ to $(0,0)$ in which a state (p,q) with $p = 0$, $q \neq 0$ or $q = 0$, $p \neq 0$ is crossed at least one time.*

Proof. Let $x_1x_2\cdots x_n = y_1y_2\cdots y_m = z$ be a non-trivial relation in X. By definition of $\mathcal{A}^*$, there are in $\mathcal{A}^*$ two paths labelled by z from 0 to 0 of the form



Two paths between vertically aligned states have the same label and x_i (resp. y_j) labels a 0-simple path in $\mathcal{A}^*$. Two vertically aligned states (q,p) which are not represented are such that $p \neq 0$ and $q \neq 0$. If $p_i \neq 0$ for any $1 \leq i \leq s$ and $q_j \neq 0$ for any $1 \leq j \leq r$), then $r = s = 0$ and z is not a non-trivial relation. Thus one has $r \geq 1$ or $s \geq 1$ and there is a path in $\mathcal{A}^* \times \mathcal{A}^*$ from $(0,0)$ to $(0,0)$ in which a state (p,q) with $p = 0$, $q \neq 0$ or $q = 0$, $p \neq 0$ is crossed at least one time. Furthermore, if this path is not $(0,0)$-simple, z is not a prime relation,

Conversely, let us assume that z is the label of a path c in $\mathcal{A}^* \times \mathcal{A}^*$ from $(0,0)$ to $(0,0)$ in which a state (p,q) with $p = 0$, $q \neq 0$ or $q = 0$, $p \neq 0$ is crossed at least one time. This path projects onto two paths of $\mathcal{A}^*$ from 0 to 0. Let $(x_i)_{1 \leq i \leq n}$ (resp. $(y_j)_{1 \leq j \leq m}$) be the sequence of labels of the 0-simple paths of the 0-simple-path decomposition of the first (resp. the second) path. Let us assume that at some time the first path attains the state 0 and the second one the state $p \neq 0$. Then $(x_i)_{1 \leq i \leq n} \neq (y_j)_{1 \leq j \leq m}$, and thus, as $z = x_1x_2\cdots x_n = y_1y_2\cdots y_m$, z is a non-trivial relation. Furthermore, if the relation is not prime, the path c is not $(0,0)$-simple.

Note that we have not used the fact that the automaton $\mathcal{A}$ is unambiguous. □

Proof of Theorem 4. Let $\mathcal{A} = (Q, \{0\}, E, \{t\})$ be a normalized automaton accepting the regular code X. We build the square automaton $\mathcal{A}^* \times \mathcal{A}^*$ and split the state $(0,0)$ into two states $(0,0)_s$ and $(0,0)_t$ such that the edges previously going out of $(0,0)$ go out of $(0,0)_s$, and the edges previously coming in $(0,0)$ come in $(0,0)_t$. Note that $(0,0)_s$ has no incoming edges and

$(0,0)_t$ has no outgoing edges. Hence for each path from $(0,0)_s$ to $(0,0)_t$, no internal state is equal to $(0,0)_s$ or $(0,0)_t$.

We keep in $\mathcal{A}^* \times \mathcal{A}^*$ only the states belonging to a path from $(0,0)_s$ to $(0,0)_t$ in which a state (p,q) with $p=0, q \neq 0$ or $p \neq 0, q=0$ is crossed at least once. Hence a word z is of a prime relation if and only if z is the label of a path from $(0,0)_s$ to $(0,0)_t$.

By using the state-elimination technique due to J. Brzozowski and E. McCluskey (see for instance [13, p. 142]), we remove the states (p,q) with p and q distinct from 0 and get an automaton $\mathcal{B}$ labelled in regular subsets of A^* whose states are $(0,0)_s$, $(0,0)_t$, or a state (p,q) with $p=0, q\neq 0$ or $p \neq 0, q=0$. There is at most one edge between two states and each label is a regular non-empty subset of A^*. States (p,q) with $p=0$ are called *left-zero states* while states (p,q) with $q=0$ are called *right-zero states*. Hence $(0,0)_s$ and $(0,0)_t$ are both left and right-zero states.

A prime relation $z = x_1x_2\cdots x_n = y_1y_2\cdots y_m$ is the label of a path c from $(0,0)_s$ to $(0,0)_t$. We get from Proposition 8 that this path can be factorized in a product of paths $c_1c_2\ldots c_n$ such that each c_i is a path from a left-zero state to a left-zero state (with n maximal with respect to this condition) and x_i is the label of c_i for $1 \leq i \leq n$. Analogously, c can be factorized in a product of paths $d_1d_2\ldots d_m$ such that each d_j is a path from a right-zero state to a right-zero state (with m maximal with respect to this condition) and y_j is the label of d_j for $1 \leq j \leq m$.

For each path c from $(0,0)_s$ to $(0,0)_t$, let X_c be the set of above defined words $x_1,\ldots x_n, y_1,\ldots,y_m$. Therefore X_c is a subset of a same class of the canonical partition of X. Furthermore, if c, c' are two paths from $(0,0)_s$ to $(0,0)_t$ sharing the same first edge , then $X_c \cup X_{c'}$ is a subset of a same class of the canonical partition of X. As a consequence, the number of classes of the canonical partition of X is finite. It is bounded above by the number of edges going out of $(0,0)_s$ plus one. We show below that these classes are regular and we give an algorithm to compute them.

We denote by $E_{(p,q)(p',q')}$ the regular set which is the label of the edge $(p,q) \to (p',q')$. With a slight abuse of language, we sometimes say that there is an edge labelled by a word w from a state (p,q) to a state (p',q') whenever $w \in E_{(p,q)(p',q')}$.

We denote by LR (resp. RL) the set of edges going from a left-zero state to a right-zero state (resp. going from a right-zero state to a left-zero state).

Let q, q' be two states distinct from 0. We denote by

- $L_{(q,0)(q',0)}$ the regular set of labels of paths from $(q,0)$ to $(q',0)$ with *all their states being right-zero states.*

- $S_{(q,0)(q',0)}$ the union of the labels of all edges contained in a path from $(q,0)$ to $(q',0)$ with *all their states being right-zero states.*

Note that we may have $q = q'$. In this case, $L_{(q,0)(q',0)}$ contains the empty word and $S_{(q,0)(q',0)}$ may be the empty set.

Let $e = (0,p) \to (q,0) \in LR$, $f = (q',0) \to (0,p') \in RL$, with $q \neq 0$ and $q' \neq 0$. We define the regular sets

$$L = E_{(0,p)(q,0)} \cdot L_{(q,0)(q',0)} \cdot E_{(q',0)(0,p')} \ \cup \ S_{(q,0)(q',0)},$$
$$S_{ef} = \begin{cases} L & \text{if } p \neq 0, p' \neq 0, \\ L \cup E_{(0,0)_s(q,0)} & \text{if } p = 0, p' \neq 0, \\ L \cup E_{(q',0)(0,0)_t} & \text{if } p \neq 0, p' = 0, \\ L \cup E_{(0,0)_s(q,0)} \cup E_{(q',0)(0,0)_t} & \text{if } p = p' = 0. \end{cases}$$

where the dot symbol is the concatenation symbol.

Let

$$e = (0,p) \to (q,0) \in LR,$$
$$f = (r,0) \to (0,s) \in RL,$$
$$g = (0,t) \to (u,0) \in LR.,$$

with q, r, s, t distinct from 0. We define the regular set

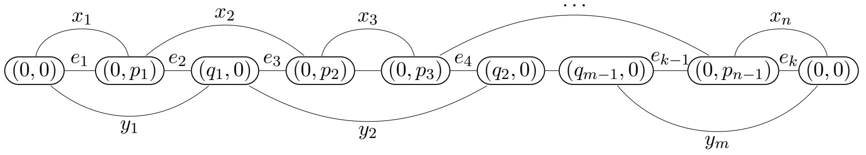
$$S_{efg} = E_{(0,p)(q,0)} \cdot L_{(q,0)(r,0)} \cdot E_{(r,0)(0,s)} \ \cup \ E_{(r,0)(0,s)} \cdot L_{(0,s)(0,t)} \cdot E_{(0,t)(u,0)}.$$

We define similar sets S_{ef} and S_{efg} when $e, g \in RL$ and $f \in LR$ by exchanging the roles played by the left and right states.

We obtain a finite number of regular subsets of X. Some of these subsets may have a non-empty intersection. We replace two parts having a non-empty intersection by their union. Hence, after a finite number of steps we get a finite number of regular subsets of X whose two by two intersections are empty. We denote these sets by $X_1, X_2, \ldots, X_r$. We define the set $X_0 = X - \bigcup_{i=1}^{r} X_i$. We claim that $(X_i)_{0 \leq i \leq r}$ is the canonical coding partition of X, which proves the proposition.

To prove our claim, we show that any two code words that belong to a same prime relation belong to a same component X_i.

Let $z = x_1 x_2 \ldots x_n = y_1 y_2 \ldots y_m$ be a prime relation where x_i, y_j are code words. It is the label of a path c from $(0,0)_s$ to $(0,0)_t$ in $\mathcal{B}$ which is factorized in a product of paths $c_1 c_2 \ldots c_n$ such that each c_i is a path labelled by x_i from a left-zero state to a left-zero state (with n maximal with respect to this condition), and in a product of paths $d_1 d_2 \ldots d_m$ such that each d_j is a path labelled by y_j from a right-zero state to a right-zero state (with m maximal with respect to this condition) as follows



with $p_i, q_j \neq 0$. We set $p_0 = p_n = 0$ and $q_0 = q_m = 0$. We denote by $(e_i)_{1 \leq i \leq k}$ the sequence of edges of this path which belong either to RL or to LR. By definition of the sets $S_{e_i e_{i+1}}$ and the sets $S_{e_i e_{i+1} e_{i+2}}$, we get that all x_i and all y_j belong to a same set X_k.

Conversely, we prove that if two words x and y belong to a same set X_k, then they belong to a same class of the canonical coding partition. Let us assume that x and y belong to a same set X_k.

Let q, q' be two non-null states in Q. We first show that if two words $y, y' \in S_{(q,0)(q',0)}$, then y, y' belong to a same class of the canonical coding partition.

Since $y, y' \in S_{(q,0)(q',0)}$, there are in $\mathcal{B}$ two paths labelled xyz and $x'y'z'$, with $x, x', z, z,' \in A^*$, containing respectively an edge labelled by y and an edge labelled by y', with the following form

$$\begin{aligned}
&(q,0) \xrightarrow{x} (r,0) \xrightarrow{y} (s,0) \xrightarrow{z} (q',0),\\
&(q,0) \xrightarrow{x'} (t,0) \xrightarrow{y'} (u,0) \xrightarrow{z'} (q',0).
\end{aligned}$$

Since $(q,0)$ is accessible from $(0,0)_s$ and $(q',0)$ is co-accessible from $(0,0)_t$, these paths can be extended in $\mathcal{B}$ with a path from $(0,0)_s$ to $(q_1,0)$ labelled by a word v, and with a path from $(q',0)$ to $(0,0)_t$ labelled by a word w. The resulting paths are

$$\begin{aligned}
&(0,0)_s \xrightarrow{v} (q,0) \xrightarrow{x} (r,0) \xrightarrow{y} (s,0) \xrightarrow{z} (q',0) \xrightarrow{w} (0,0)_t,\\
&(0,0)_s \xrightarrow{v} (q,0) \xrightarrow{x'} (t,0) \xrightarrow{y'} (u,0) \xrightarrow{z'} (q',0) \xrightarrow{w} (0,0)_t.
\end{aligned}$$

Let for instance $(0,0)_s \xrightarrow{v_1} (q_1,0)$ be the first edge of the path $(0,0)_s \xrightarrow{v} (q,0)$. Hence v_1 and y belong to a same prime relation, and v_1 and y' belong to a same prime relation. As a consequence y, y' belong to a same class of the canonical coding partition.

Let now x and y be two words in S_{ef}, with $e = (0,p) \to (q,0) \in LR$ and $f = (q',0) \to (0,p') \in RL$. We consider the first case in the definition of S_{ef}. For instance, one may assume that

$$\begin{aligned}
&x \in E_{(0,p)(q,0)} \cdot L_{(q,0)(q',0)} \cdot E_{(q',0)(0,p')},\\
&y \in S_{(q,0)(q',0)}.
\end{aligned}$$

It follows that there is in $\mathcal{B}$ a path labelled by x starting with e, ending with f, and containing an edge labelled by $y' \in S_{(r,0)(s,0)}$ which has the following form

$$(0,p) \to (q,0) \to \ldots \to (r,0) \xrightarrow{y'} (s,0) \to \ldots \to (q',0) \to (0,p').$$

Since $(0,p)$ is accessible from $(0,0)_s$ and $(0,p')$ is co-accessible from $(0,0)_t$, this path can be extended in $\mathcal{C}$ by a path from $(0,0)_s$ to $(0,p)$ labelled by a word v and, by a path from $(0,p')$ to $(0,0)_t$ labelled by a word w. The resulting path is

$$(0,0)_s \xrightarrow{v} (0,p) \to (q,0) \to \ldots \to (r,0) \xrightarrow{y'} (s,0) \ldots (q',0) \to (0,p') \xrightarrow{w} (0,0)_t.$$

This defines a prime relation containing the words x and y'. Since y and y' belong to a same class of the canonical coding partition, x and y also.

We consider similarly all cases in the definitions of S_{ef} and S_{efg} to conclude that a each set S_{ef} (resp. each set S_{efg}) is included in a class of the canonical coding partition. □

Note that the computation of the sets S_{ef} and S_{efg} can be performed in polynomial time. Nevertheless, since it is necessary to compute some intersections to get the automata accepting the classes X_i, the computation of the components X_i for $i \neq 0$ cannot be achieved in polynomial time.

Example 3. We consider the code $X = a + bb + c + ad^*b + bc^*bb$. The unambiguous finite state automaton $\mathcal{A}^* = (Q, \{0\}, E, \{0\})$ accepting the set X^* is described in Figure 1. The square automaton $\mathcal{A}^* \times \mathcal{A}^*$ is described in

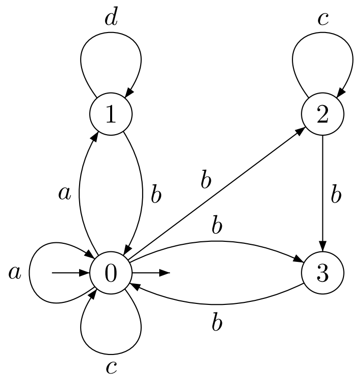


Figure 1: An automaton accepting X^*. This automaton is obtained by merging the initial and final states of a normalized unambiguous automaton accepting X.

Figure 2, and the automaton $\mathcal{B}$ in Figure 3. By the definition of the sets S_{ef}, for

$$e = (0,0)_s \xrightarrow{a} (1,0) \text{ and } f = (1,0) \xrightarrow{b} (0,3),$$

we get

$$S_{ef} = \{a, ab\}.$$

With

$$e = (0,1) \xrightarrow{b} (2,0) \text{ and } f = (2,0) \xrightarrow{bb} (0,0)_t,$$

we have

$$S_{ef} = bc^*bb + c + bb,$$

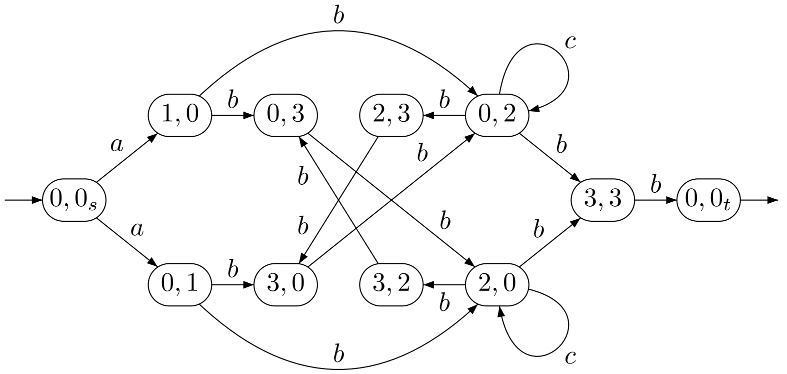


Figure 2: The square automaton $\mathcal{A}^* \times \mathcal{A}^*$ labelled in A. We only keep the states belonging to paths going from $(0,0)_s$ to $(0,0)_t$ through at least one state (p,q) with $p = 0, q \neq 0$ or $p \neq 0, q = 0$.

where the symbol + denotes the union of regular languages.

With

$$e = (0,0)_s \xrightarrow{a} (0,1) \quad f = (0,1) \xrightarrow{b} (2,0), \text{ and } g = (2,0) \xrightarrow{bb} (0,0)_t,$$

We get

$$S_{efg} = ab + bc^*bb.$$

The computation gives the canonical partition (X_0, X_1) of X with

$$X_0 = ad^+b,$$
$$X_1 = a + ab + bb + c + bc^*bb.$$

When the code X is not regular, even when context-free, the canonical coding partition may have an infinite number of classes, as shows the following example.

Example 4. Let

$$X = \cup_{n\geq 1} (a^n b + a^n bc^n + c^n a^n b).$$

The code X is context free and its canonical coding partition is $(X_i)_{i\geq 1}$ with $X_i = a^i b + a^i bc^i + c^i a^i b$ for $i \geq 1$ and $X_0 = \emptyset$. Indeed, $\forall i \geq 1$, $a^i bc^i a^i b = a^i b \cdot c^i a^i b = a^i bc^i \cdot a^i b$ is a prime relation. Furthermore one can easily verify that there are not other prime relations.

It is also possible to get a finite canonical coding partition with non-regular classes.

Example 5. Let X be a code, for instance a non-regular uniquely decipherable code. Let Y be the code

$$Y = \{ax, xb \mid x \in X\} + \{a, b\},$$

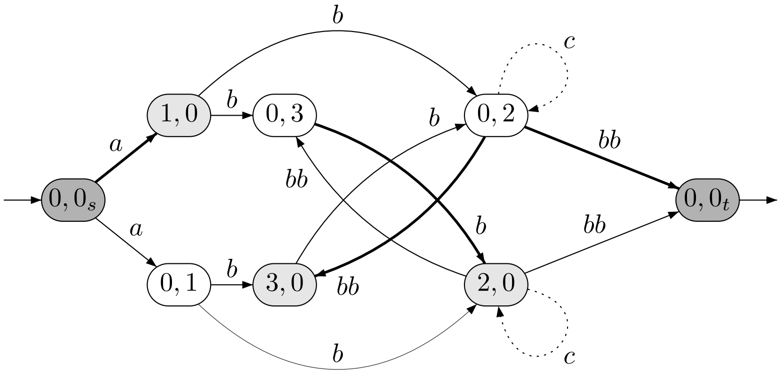


Figure 3: The automaton $\mathcal{B}$ labelled in the set of non-empty regular subsets of A^*. The right-zero states distinct from $(0,0)_s$ and $(0,0)_t$ are represented in light gray. The two states $(0,0)_s$ and $(0,0)_t$ are represented in dark gray. Edges in RL are represented with thin lines while edges in LR are represented with thick lines. Edges going from a left-zero state to a left-zero state or from a right-zero state to a right-zero state are dashed. By definition of $\mathcal{B}$, there is no edge from $(0,0)_s$ and $(0,0)_t$.

where a, b are two symbols which do not appear in the words of X. The canonical coding partition of Y is made of a unique class since $axb = ax \cdot b = a \cdot xb$. Such a code is TA. Let for instance $X = \{a^n b^n c^n \mid n \geq 1\}$. It is context-free and not regular. The code Y, equal to the unique class of its canonical coding partition, is TA and not regular.

4 Conclusion

We have proved that the components of the canonical coding partition of a regular code are regular and we have given an algorithm based on automata for computing these components. In [3] is given an algorithm, which follows a Sardinas-Patterson scheme, for computing the components of the canonical coding partition of a finite code. The hypothesis that the code X is finite intervene in the algorithm. Similarly, there are mainly two types of algorithms for checking whether a regular code is UD. One is the Sardinas-Patterson algorithm. Another one is based on automata and uses the square of an automaton like in the algorithm presented in this paper. The same situation holds for checking whether a code is a circular code (see [2]). So, it is natural to conjecture that there is also a Sardinas-Patterson like algorithm for computing the components of the canonical coding partition of a regular code. This algorithm would extend the one described in [3] which is valid for finite codes.

5 Acknoledgments

We author would like to thank the reviewers for many helpful suggestions to improve the presentation of the paper.

References

[1] M.-P. Béal and D. Perrin, *Codes, unambiguous automata and sofic systems*, Theoret. Comput. Sci., 356 (2006), pp. 6–13.

[2] J. Berstel and D. Perrin, *Theory of codes*, vol. 117 of Pure and Applied Mathematics, Academic Press Inc., Orlando, FL, 1985. http://www-igm.univ-mlv.fr/~berstel/LivreCodes/Codes.html.

[3] F. Burderi and A. Restivo, *Coding partitions*, Discret. Math. Theor. Comput. Sci., Vol 9, No 2 (2007), pp. 227–240.

[4] M. Dalai and R. Leonardi, *Non prefix-free codes for constrained sequences*, in International Symposium on Information Theory, 2005. ISIT 2005, IEEE, 2005, pp. 1534–1538.

[5] S. Eilenberg, *Automata, languages, and machines. Vol. A*, Academic Press, New York, 1974. Pure and Applied Mathematics, Vol. 58.

[6] H. Fernau, K. Reinhardt, and L. Staiger, *Decidability of code properties*, in Developments in Language Theory, 1999, pp. 153–163.

[7] G. Gönenç, *Unique decipherability of codes with constraints with application to syllabification of Turkish words*, in COLING 1973: Computational And Mathematical Linguistics: Proceedings of the International Conference on Computational Linguistics, vol. 1, 1973, pp. 183–193.

[8] F. Guzmán, *Decipherability of codes*, J. Pure Appl. Algebra, 141 (1999), pp. 13–35.

[9] J. Karhumäki, W. Plandowsky, and W. Rytter, *Generalized factorizations of words and their algorithmic properties*, Theoret. Comput. Sci., 218 (1999), pp. 123–133.

[10] A. Lempel, *On multiset decipherable codes*, IEEE Trans. Inform. Theory, 32 (1986), pp. 714–716.

[11] D. Perrin, *Finite automata*, in Handbook of theoretical computer science, Vol. B, Elsevier, Amsterdam, 1990, pp. 1–57.

[12] A. Restivo, *A note on multiset decipherable codes*, IEEE Trans. Inform. Theory, 35 (1989), pp. 662–663.

[13] J. Sakarovitch, *Éléments de théorie des automates*, Vuibert, Paris, 2003. English translation to appear, Cambridge University Press.

[14] ——, *Elements of Automata Theory*, Cambridge University Press, 2009.