Transducers for bidirectional decoding of codes with a finite deciphering delay

L. Giambruno[1], S. Mantaci [2], J. Néraud[3] and C. Selmi [3]

GREYC - Université de Caen Basse-Normandie

Dipartimento di Matematica e Informatica - Università di Palermo (Italy)

LITIS - Université de Rouen

Journées COMATEGE-SeqBio 2012
LIGM, Université Paris-Est Marne-la-Vallée

Codes

- Let A be a finite set, which we call an alphabet
- Let A^* be the set of words on A
- Let $A^+ = A^* \setminus \{\epsilon\}$ be the set of nonempty words on A

Definition

A subset X of A^* is said a code over A if any word in X has a unique factorization in words in X

Example

- Let $A = \{0, 1\}$ and let $X = \{10, 01, 0\}$. X is not a code since 010 admits two different factorizations in elements of X.
- Let $A = \{0, 1\}$ and let $X = \{10, 001, 010\}$. Then X is a code, in particular it is a prefix code.

Encoding

Let B and A be two alphabets that we call respectively source alphabet and channel alphabet.

Definition

A mapping $\gamma : B \longrightarrow A^+$ which is extended to words by $\gamma(b_1 \cdots b_n) = \gamma(b_1) \cdots \gamma(b_n)$ is an encoding or is uniquely decipherable if $\gamma(w) = \gamma(w') \Longrightarrow w = w'$ for each pair of words w, w' over B,

Proposition

If γ is an encoding then $\gamma(B)$ is a code.

- for $b \in B$, $\gamma(b)$ is said a codeword
- any word in $\gamma(B)^*$ is said a message on $\gamma(B)$

Example

$B = \{b_1, b_2, b_3\}$. The map γ from B to $A^* = \{0, 1\}^*$ defined as $\gamma(b_1) = 10, \gamma(b_2) = 001, \gamma(b_3) = 010$ is an encoding .

Decoding

Definition

A decoding of an encoding γ is the function γ^{-1} restricted to $\gamma(B^*)$.

The decoding of a message w in A^+ is its decomposition in elements of B by applying γ^{-1} i.e. $\gamma^{-1}(w)$

MAIN PURPOSE

To study coding methods that allow bidirectional decodings of the encoded messages

Why such codings are interesting? Data-integrity: correcting errors that can occur while encoding or transmitting a message

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 00000\textcolor{green}{00001}$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Prefix codes

Definition

A code X is called prefix (resp. suffix) if no element of X is a prefix (resp. a suffix) of another element of X

Remark

A prefix code can be decoded without any delay in a left-to-right parsing. On the other hand, it can not be as easily decoded from right to left.

$$X = \{10, 001, 010\} \qquad w = 00101010010$$

$$X = \{1, 00, 01\} \qquad w = 0000000001$$

Remark

We are not able to decode the second message from right to left until we have read all the message.

Codes with deciphering delay

Intuitively, codes with finite deciphering delay can be decoded, from left to right, with a finite lookahead.

Definition

A set X of nonempty words in A is said to have finite deciphering delay (in short f.d.d.) d, if there exists an integer $d > 0$ such that, for all $x, x' \in X$, $y \in X^d$, $y' \in X^*$, xy is a prefix of $x'y'$ implies that $x = x'$. Otherwise we say that X has infinite deciphering delay.

Example

- Any prefix code is a code with a f.d.d. $d = 0$
- For any $d = 0$, $X^d = \{01, (01)^d 1\} \in A^*$, $A = \{0, 1\}$ is a code with f.d.d. d
- $X = \{1, 00, 10\}$ has infinite deciphering delay

On Maximal Codes

Definition

A code is maximal if it is not contained in any other code.

In the case of maximal prefix codes, we have a very strong result due to Schutzenberger in 1961:

Theorem

A finite maximal code with finite deciphering delay is prefix.

This means that the only maximal codes that can be bidirectionally decoded with finite delay are the ones that are both prefix and suffix.

On reliability of messages: bifix codes

Definition

A bifix code is a prefix and suffix code.

Example

- A prefix code which is equal to its reversal is bifix. For example $X = 01^*0$.
- Any set of words of the same length is a bifix code.

- Bifix codes are used for checking errors that may occur in the encoding or in the transmission of the message.
- A single error that occurs into a compressed stream can propagate and cause many consecutive errors when the stream is decompressed.

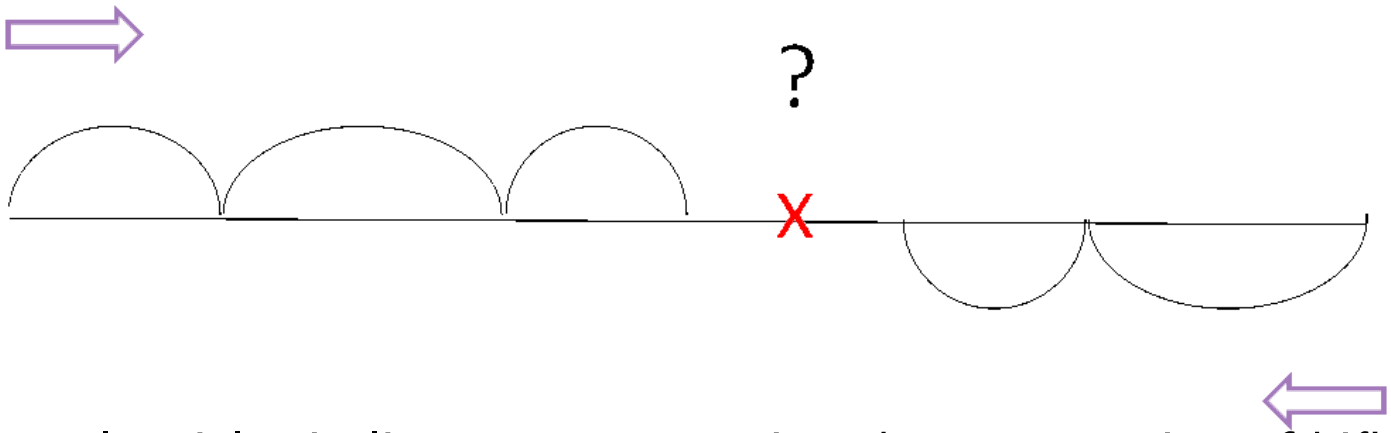
Remark

Bifix codes are used for the compression of moving pictures. The Advanced Video Coding (AVC) standard recommends the use of these codes.

Bidirectional codes

Bifix codes are used for checking errors that may occur in the encoding or in the transmission of the message.

The compressed file is usually divided into records of codewords that are bidirectionally decoded.



There are several articles in literature concerning the construction of bifix codes starting from codes. For instance in [Fraenkel and Klein, 1990] the authors study how to construct affix Huffman codes

It is important to have other coding methods that allow a bidirectional decoding of the messages.

Another simpler solution: Girod's method

- In 1999 Girod introduced a method where any sequence of codewords on a finite prefix code X on a binary alphabet is transformed using the XOR operation into a bitstring that can be bidirectionally decoded.

B. Girod. Bidirectionally decodable streams of prefix code words. IEEE Communications Letters,1999.

Girod's method depends on

- a binary alphabet A
- a finite prefix code X on A
- the keyword 0^L, with L the length of one of the longest words in X
- the XOR operation

Our results

Our results

- We define a generalization of Girod's encoding method depending on
  - a finite alphabet within a special operation
  - a finite code with finite deciphering delay (in short f.d.d.) d
  - a keyword depending on X and on d

Motivations

- Data integrity : we get a bidirectional deciphering delay in decoding equal to the minimal deciphering delay between the code and its reverse.
- Codes with a finite deciphering delay play a prominent part in literature

Our results

Our results

- We define a deterministic transducer for such generalized method
- We prove interesting properties as:
  - Transducers associated to different operations are isomorphic as unlabelled graphs
  - Transducers associated to different keys have an isomorphic non trivial strongly connected component

Motivations

- Importance of studying transducers with particular properties (bi-determinism, minimality) also from an algorithmic point of view
- In symbolic dynamics: to every code with f.d.d. we can associate an irreducible sofic shift.

Girod's generalized method: Our hypotheses

Let A, B be finite alphabets Generalized Girod's method depends on

- a finite alphabet A
- a finite code X with finite deciphering delay d on A
- a latin square map f on A
- a keyword x_L, where x_L is a word in A^+ of length $L = (d+1)l$ with l the length of the longest word in X

We define the generalization of Girod's encoding from B^* to A^* depending on these hypotheses.

Latin square maps

Definition

Let A be a finite alphabet. A map $f : A \times A \longrightarrow A$ is called a latin square map on A if its components functions are bijective.

Latin square maps are represented as matrices of size $card(A) + 1$ *where each line and each column contains only one occurrence of* i*, for all* $0 \leq i \leq card(A)$

If $A = \{0, 1, 2\}$, the following matrix defines a latin square map:

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

Remark

There exists only one solution to the equation $f(a, b) = c$, when one element among a, b or c is unknown.

Generalized Girod's Encoding - Example

- $X = \{01, 012\}$ code with f.d.d. 1 on $A = \{0, 1, 2\}$
  (X defined by an encoding γ on $B = \{b_1, b_2\}$)
- $\widetilde{X} = \{10, 210\}$ set of reverse words of X
- $x_L = 011011$
- g latin square map

g	**0**	**1**	**2**
0	0	2	1
1	1	0	2
2	2	1	0

We define the generalized Girod's encoding δ on $y = x_2x_1x_2 = 012.01.012$
Let $y' = \tilde{x_2}\tilde{x_1}\tilde{x_2} = 210.10.210$

Generalized Girod's Encoding - Example

- $X = \{01, 012\}$ code with f.d.d. 1 on $A = \{0, 1, 2\}$
  (X defined by an encoding γ on $B = \{b_1, b_2\}$)
- $\widetilde{X} = \{10, 210\}$ set of reverse words of X
- $x_L = 011011$
- g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

We define the generalized Girod's encoding δ on $y = x_2x_1x_2 = 012.01.012$
Let $y' = \tilde{x_2}\tilde{x_1}\tilde{x_2} = 210.10.210$

$$x = yx_L \quad = \quad 012010120110 11$$

Generalized Girod's Encoding - Example

- $X = \{01, 012\}$ code with f.d.d. 1 on $A = \{0, 1, 2\}$
  (X defined by an encoding γ on $B = \{b_1, b_2\}$)
- $\widetilde{X} = \{10, 210\}$ set of reverse words of X
- $x_L = 011011$
- g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

We define the generalized Girod's encoding δ on $y = x_2x_1x_2 = 012.01.012$
Let $y' = \tilde{x_2}\tilde{x_1}\tilde{x_2} = 210.10.210$

$$x = yx_L = 012010120110 11$$
$$x' = \tilde{x_L}y' = 110110210 10210$$

Generalized Girod's Encoding - Example

- $X = \{01, 012\}$ code with f.d.d. 1 on $A = \{0, 1, 2\}$
  (X defined by an encoding γ on $B = \{b_1, b_2\}$)
- $\widetilde{X} = \{10, 210\}$ set of reverse words of X
- $x_L = 011011$
- g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

We define the generalized Girod's encoding δ on $y = x_2x_1x_2 = 012.01.012$
Let $y' = \tilde{x_2}\tilde{x_1}\tilde{x_2} = 210.10.210$

$$\begin{aligned} x = yx_L &= 012010120 11011 \\ x' = \tilde{x_L}y' &= 110110210 10210 \\ z = g(x, x') &= 202200210 01101 \end{aligned}$$

z is the generalized Girod's encoding of $w = b_2b_1b_2$

Generalized Girod's Left-to-Right decoding - Example

$X = \{01, 012\}$ - $x_L = 011011$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

$$z = 20220021001101$$
$$x' = 110110$$
$$x =$$

$$\delta^{-1}(z) =$$

Generalized Girod's Left-to-Right decoding - Example

$X = \{01, 012\}$ - $x_L = 011011$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

$$z = 20220021001101$$
$$x' = 110110$$
$$x = 012010$$

$\delta^{-1}(z) =$

Generalized Girod's Left-to-Right decoding - Example

$X = \{01, 012\}$ - $x_L = 011011$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

$$z = 20220021001101$$
$$x' = 110110$$
$$x = \mathbf{012}010$$

$\delta^{-1}(z) = \mathbf{b_2}$

Generalized Girod's Left-to-Right decoding - Example

$X = \{01, 012\}$ - $x_L = 011011$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

$$z = 20220021001101$$
$$x' = 110110\mathbf{210}$$
$$x = \mathbf{012}010$$

$\delta^{-1}(z) = \mathbf{b_2}$

Generalized Girod's Left-to-Right decoding - Example

$X = \{01, 012\}$ - $x_L = 011011$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

$$z = 20220021001101$$
$$x' = 110110210$$
$$x = 012\mathbf{01}0120$$

$\delta^{-1}(z) = \mathbf{b_2}$

Generalized Girod's Left-to-Right decoding - Example

$X = \{01, 012\}$ - $x_L = 011011$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

$$z = 20220021001101$$
$$x' = 11011021\mathbf{10}$$
$$x = 012\mathbf{01}0120$$

$\delta^{-1}(z) = \mathbf{b_2 b_1}$

Generalized Girod's Left-to-Right decoding - Example

$X = \{01, 012\}$ - $x_L = 011011$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

$$z = 20220021001101$$
$$x' = 11011021010\mathbf{210}$$
$$x = 01201\mathbf{012}0$$

$\delta^{-1}(z) = \mathbf{b_2 b_1 b_2}$

Generalized Girod's Left-to-Right decoding - Example

$X = \{01, 012\}$ - $x_L = 011011$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0

$$z = 20220021001101$$
$$x' = 11011021010210$$
$$x = 01201012\mathbf{011011}$$

$\delta^{-1}(z) = \mathbf{b_2 b_1 b_2}$

Definition

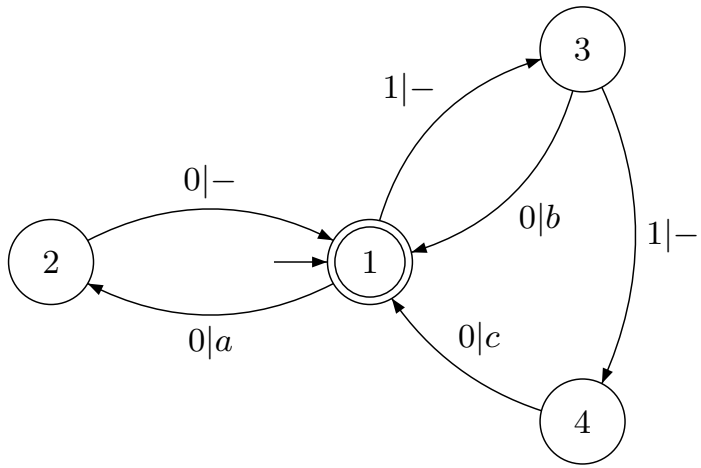
A finite transducer $\mathcal{T}$ uses an input alphabet A and an output alphabet B. It is composed of

- a finite set Q of states
- two distinguished subsets I and T called the sets of initial and terminal states
- a set E of edges composed of quadruples (p, u, v, q) where p and q are states and u is a word over A and v is a word over B. An edge is also denoted by $p \xrightarrow{u|v} q$.

Definition

- A path is a sequence of consecutive edges.
- The label of the path is the concatenation of the labels of the edges.
- A path is successful if it starts in an initial state and ends in a terminal state.

We can represent a transducer by a labelled graph:



Definition

Definition

- A transducer is literal if input labels are letters.
- A literal transducer is deterministic (resp. codeterministic) if for each state p and for each input letter a there exists at most one edge that starts (resp. ends) at p with input letter a.

Definition

A sequential transducer is composed of a deterministic transducer and an output function that maps terminal states into words on the output alphabet.

Proposition

There exists a unique minimal sequential transducer equivalent to a given one.

Encoders and Decoders

Definition

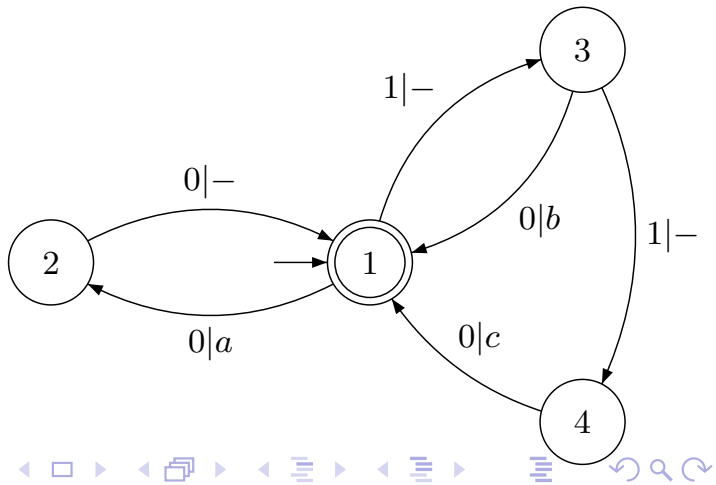
A transducer $\mathcal{T}$ defines a binary relation between words, called the relation realized by $\mathcal{T}$. It is composed of the pairs which are the labels of successful paths.

Transducers are mainly used for decoding. In this case, A is the channel alphabet and B is the source alphabet.

Definition

An encoder (resp. decoder) is a transducer that realizes an encoding (resp. decoding).

A decoder for messages encoded by $X = \{00, 10, 110\}$



Transducer for decoding

Hypotheses

- Let A be a finite alphabet and X be a code with f.d.d. d on A
- Let f be a latin square map on A and x_L in A^+ of length $L = (d+1)l$ with l the length of the longest word in X

Let δ be the Girod's generalized encoding depending on these hypotheses.

We define a transducer $\mathcal{T}(X)_{f,x_L}$ for the left-to-right decoding and a transducer $\widetilde{\mathcal{T}}(X)_{f,x_L}$ for the right-to-left decoding.

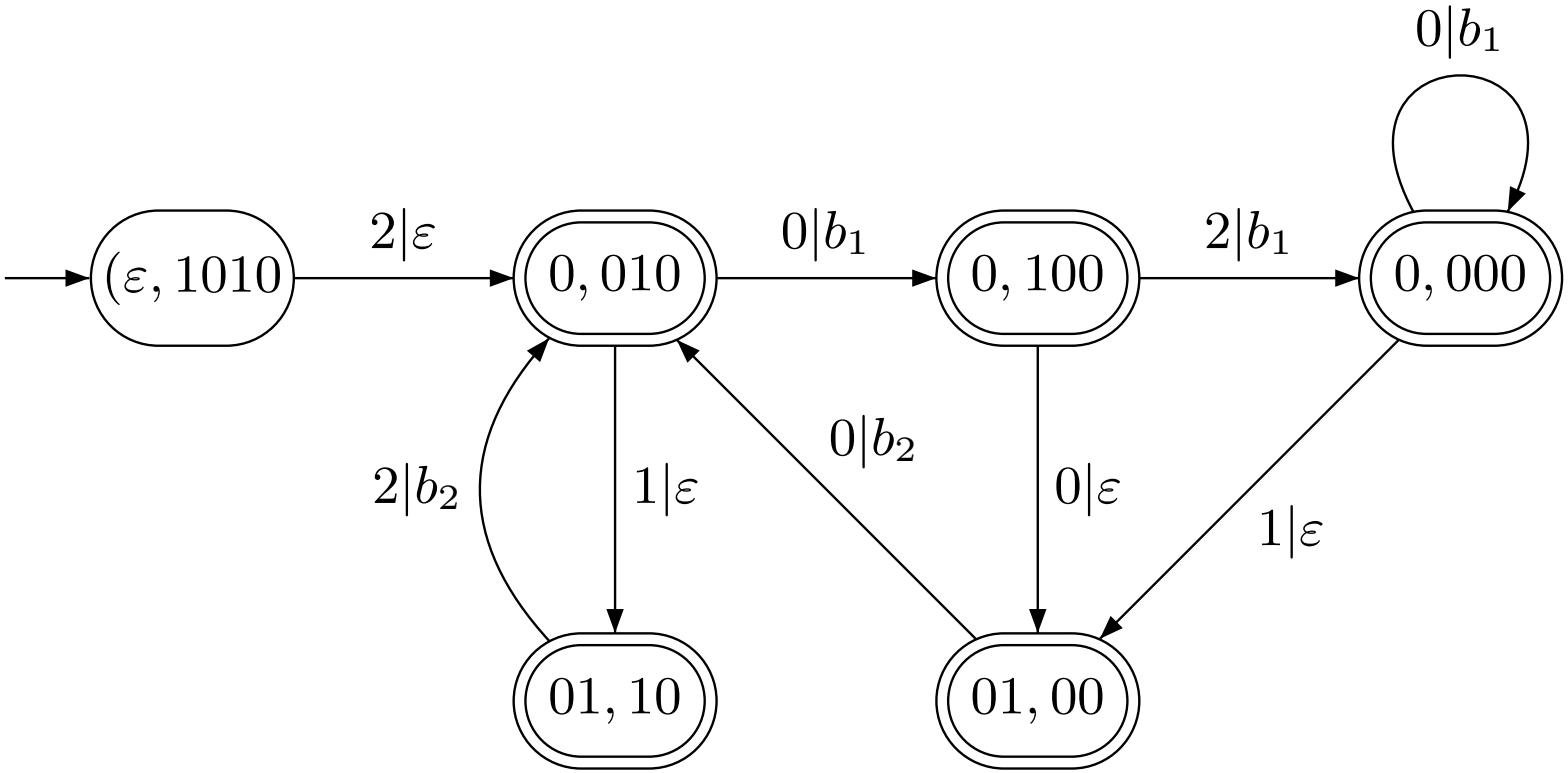
Proposition

We have that $\mathcal{T}(X)_{f,x_L}$ (resp. $\widetilde{\mathcal{T}}(X)_{f,x_L}$) realizes the left-to-right (resp. right-to-left) decoding δ^{-1} on the encoded word z, , except their suffix of length L. Moreover these transducers are deterministic.

Transducer for left-to-right decoding - Example

$X = \{0, 01\}$ - $x_L = 0101$ - g latin square map

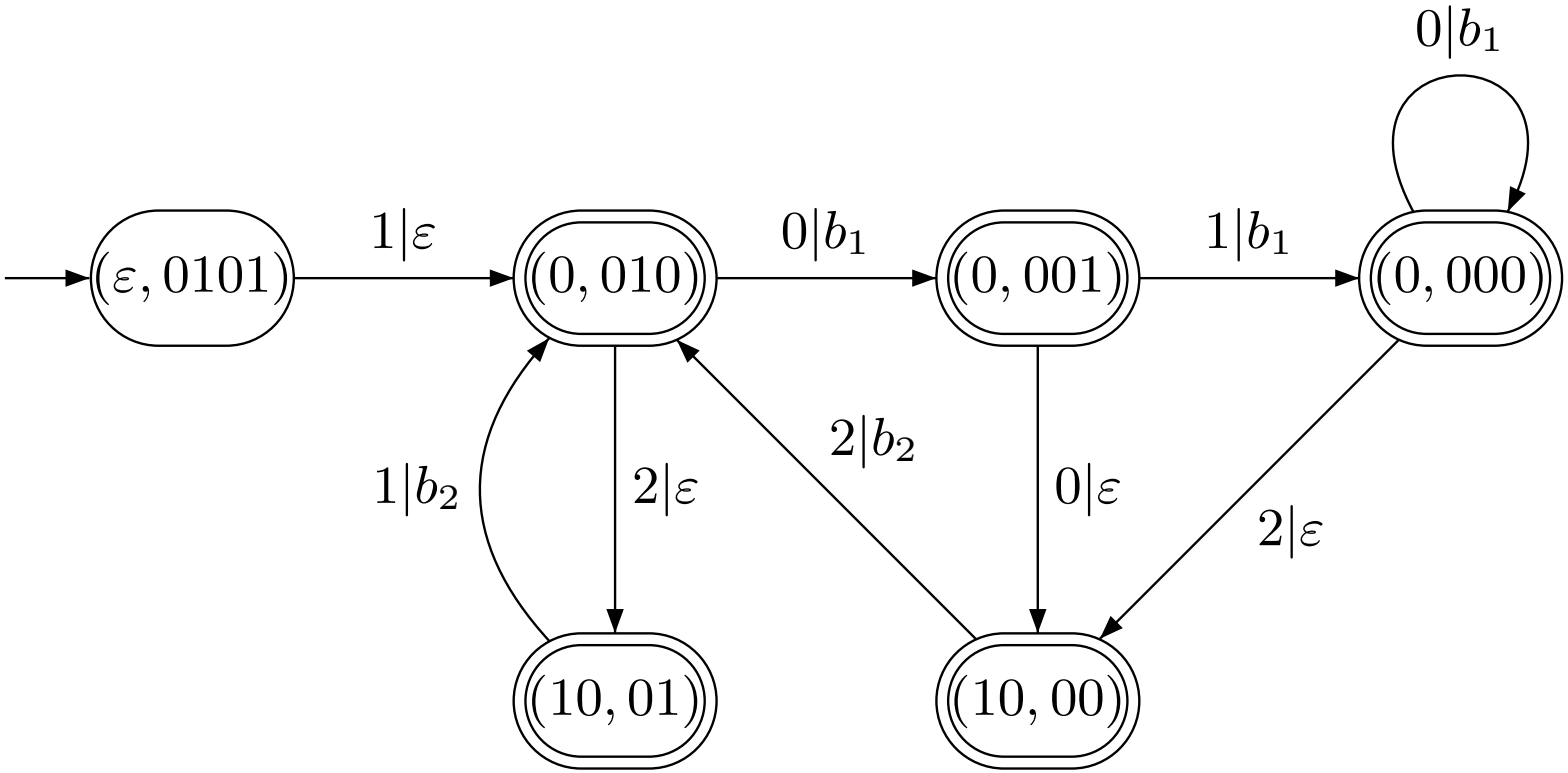
g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0



Transducer for right-to-left decoding - Example

$X = \{0, 01\}$ - $x_L = 0101$ - g latin square map

g	0	1	2
0	0	2	1
1	1	0	2
2	2	1	0



Isomorphism properties

Proposition

$\mathcal{T}(X)_{f,x_L}$ and $\widetilde{\mathcal{T}}(X)_{f,x_L}$ are isomorphic as unlabelled graphs. Moreover if f is a commutative latin square map then they are also isomorphic as transducers.

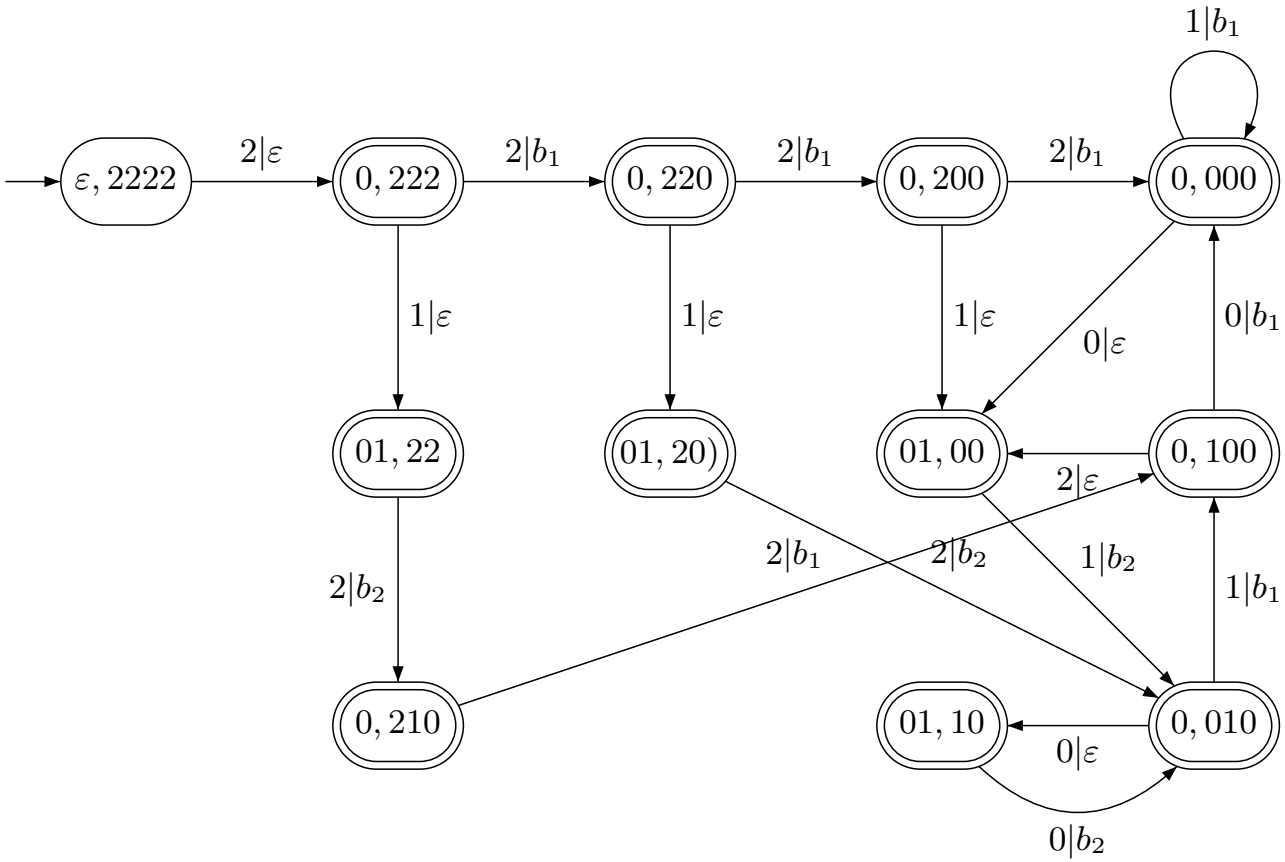
Proposition

The transducers $\mathcal{T}(X)_{f,x_L}$ and $\mathcal{T}(X)_{g,x_L}$ are isomorphic as unlabelled graphs, for all pairs of latin square maps f and g on A.

Example

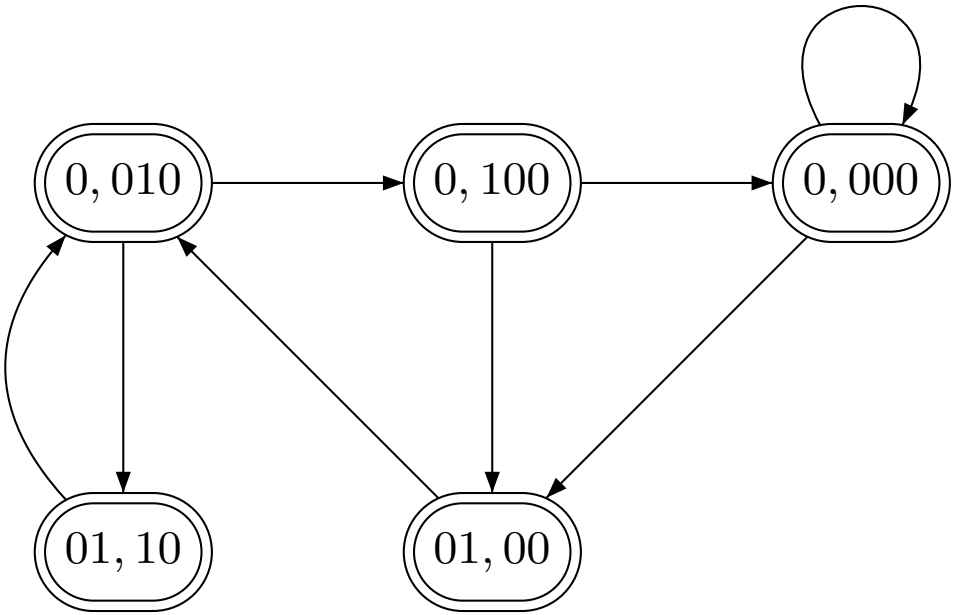
$X = \{0, 01\}$ - $x'_L = 2222$ - f latin square map

f	0	1	2
0	1	0	2
1	0	2	1
2	2	1	0



A unique strongly connected component - Example

Each of the unlabelled graphs associated to transducers $\mathcal{T}(X)_{g,x_L}$ and $\mathcal{T}(X)_{f,x'_L}$ has only one non trivial strongly connected component:



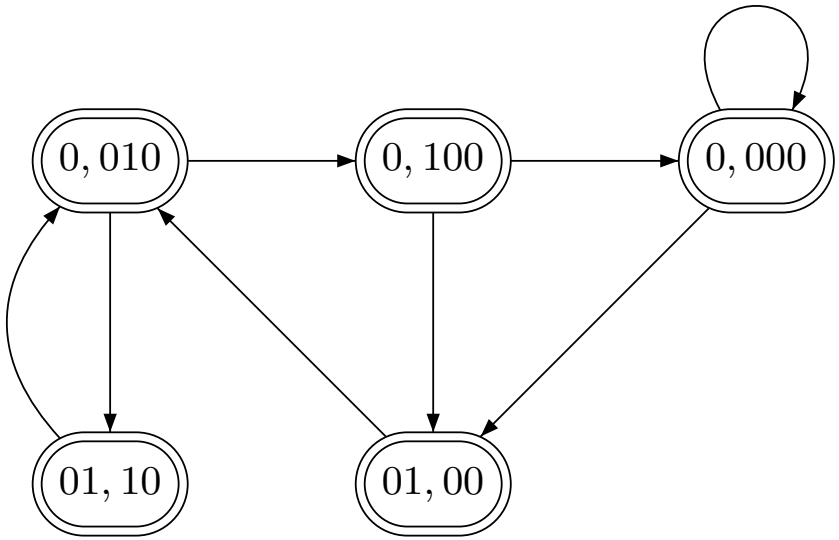
A unique strongly connected component

Let $G(X)_{x_L}$ be the graph obtained by removing the labels from the transitions of $\mathcal{T}(X)_{g,x_L}$

We consider now $C(G(X)_{x_L})$, the subgraph of $G(X)_{x_L}$ where

- the vertices are elements of $X^d(Pref(X) \setminus X) \times Suff(\widetilde{X}^+)$
- we consider the edges in $G(X)_{x_L}$ connecting vertices in the subgraph.

$X = \{0, 01\}$



A unique strongly connected component

We prove the following:

Proposition

Given a code with a finite deciphering delay X, and given a key x_L, $C(G(X)_{x_L})$ is the unique non trivial strongly connected component of $G(X)_{x_L}$ which is accessible from any vertex of $G(X)_{x_L}$.

As another remarkable property, the component $C(G(X)_{x_L})$ does not depend on the key x_L:

Theorem

Given a code X with a finite deciphering delay d, a unique graph, namely $C(X)$ exists such that $C(X) = C(G(X)_{x_L})$, for any key x_L.

Generalizations

- Generalize the technique of Girod to codes with infinite deciphering delay. Construct and study the associated transducers.
- Transducers for Girod's decoding can suggest a generalization of the method to rational prefix codes?

Link with symbolic dynamics

- To every code with f.d.d. we can associate an irreducible sofic shift. We propose to study the properties of the sofic shifts obtained from this correspondence.

MERCI BEAUCOUP!