A Generalization of Girod's Bidirectional Decoding Method to Codes with a Finite Deciphering Delay

Laura Giambruno[1], Sabrina Mantaci[2], Jean Néraud[3], and Carla Selmi[3]

[1] LIPN UMR CNRS 7030, Université Paris-Nord, 93430 Villetaneuse, France
[2] Dipartimento di Matematica e Informatica, Universitá di Palermo, 90133, Italy
[3] LITIS, Université de Rouen, 76801 Saint Etienne du Rouvray, France
giambruno@lipn.univ-paris13.fr, sabrina@math.unipa.it,
{Jean.Neraud,Carla.Selmi}@univ-rouen.fr

Abstract. In this paper we generalize an encoding method due to Girod (cf. [6]) using prefix codes, that allows a bidirectional decoding of the encoded messages. In particular we generalize it to any finite alphabet A, to any operation defined on A, to any code with finite deciphering delay and to any key $x \in A^+$, on a length depending on the deciphering delay. We moreover define, as in [4], a deterministic transducer for such generalized method. We prove that, fixed a code $X \in A^*$ with finite deciphering delay and a key $x \in A^*$, the transducers associated to different operations are isomorphic as unlabelled graphs. We also prove that, for a fixed code X with finite deciphering delay, transducers associated to different keys have an isomorphic non trivial strongly connected component.

Introduction

Coding methods that allow bidirectional decoding of messages are used in order to guarantee data integrity. When we use a variable length code for source compression (cf. [9], Chapter 3), a single bit error in the transmission of the coded word may cause catastrophic consequences during decoding, since the wrongly decoded symbol generate loose of synchronization and the error is propagated till the end of the file. In order to limit this error propagation, the compressed file is usually divided into records. If just one error occurred in the coding of the record and we are able to decode in both directions, it is possible to check the error position, isolate it and then avoid the error propagation. For this purpose, bifix codes, that allows instantaneous bidirectional decoding, are generally used but they are usually too big (so do not guarantee compression), and they are difficult to construct (cf. [3]). Prefix codes are usually very small instead, but in spite they allow an instantaneous left-to-right decoding, the right-to-left decoding requires some deciphering delay. Due to a Schützenberger famous result (cf. [2], Chapter 3) such a delay is even infinite for maximal finite prefix codes that are not suffix. In 1999 B. Girod (cf. [6]) introduced a encoding/decoding method, that, even if it makes use of prefix codes, it allows an instantaneous decoding

H.-C. Yen and O.H. Ibarra (Eds.): DLT 2012, LNCS 7410, pp. 471–476, 2012.

also from right to left by paying just few additional memory bits. In previous papers (cf. [4], [5]) a bideterministic transducer is defined for the bidirectional deciphering of words by the method introduced by Girod [6]. A generalization of this method and the corresponding transducer are also introduced.

In this paper we consider two different generalizations of the Girod's method. The first one concerns the cardinality of the code alphabet, realized by replacing the bitwise sum with a reversible operation defined by a *latin square map*. The second one concerns codes with finite deciphering delay (f.d.d.). The prominent part played by these codes is largely illustrated by deep combinatorial results, such as the two famous theorems of Schützenberger concerning their minimality [cf. [2], Chapter 5]. For this reason, it is natural to wonder what happens when, in a Girod's encoding/decoding suitable generalization, the prefix code is replaced by a code with some f.d.d. Our results allow to get a bidirectional f.d.d. equal to the minimal f.d.d. between the code and its reverse. We also show that the transducers for the same code associated to different keys have an isomorphic non trivial strongly connected component. This property emphasizes the deep connections between variable lengths codes and irreducible shifts (cf. [8]).

In Section 1 we introduce some preliminary notions on codes and transducers. In Section 2 we define latin square maps and we describe the generalization of Girod's method to f.d.d. codes. In Section 3 we give the algorithm for constructing the deterministic transducer that realizes the left-to-right and the right-to-left decoding by the generalized method. We show that the transducers realizing left-to-right and right-to-left decoding of the same inputs are isomorphic as graphs. We moreover show that the transducers over a given f.d.d. code $X \subset A^+$ and key $x \in A^*$, are isomorphic as unlabeled graphs, independently from the latin square maps used for encoding. In Section 4 we show that, for a given f.d.d. code X over a fixed alphabet A, the transducers associated to different keys have an isomorphic non trivial strongly connected component.

1 Preliminaries: Codes and Transducers

Let B and A be two alphabets, that we call respectively *source* alphabet and *channel* alphabet. Let $\gamma\colon B \to A^*$ be a map that associates to each element b in B a nonempty word over A. We extend this map to words over B by $\gamma(b_1 \dots b_n) = \gamma(b_1) \dots \gamma(b_n)$. We say that γ is an *encoding* if $\gamma(w) = \gamma(w')$ implies that $w = w'$. For each b in B, $\gamma(b)$ is said a *codeword* and the set of all codewords is said a *variable length code*, or simply a *code*. In what follows we denote by $x_i = \gamma(b_i)$ and by $X = \{x_1, \dots, x_m\}$ the code defined by γ. A *decoding* is the inverse operation than encoding i.e. the decoding of γ is the function γ^{-1} restricted to $\gamma(B^*)$. A set Y over A^* is said a *prefix set* (resp. *suffix set*) if no element of Y is a prefix (resp. a suffix) of another element of Y. Since one can prove that any prefix and any suffix set different from $\{\varepsilon\}$ is a code, we call them prefix and suffix codes, respectively. Words obtained by encoding a word in the source alphabet by a prefix code, can be decoded without delay in a left-to-right parsing.

Let $X \subset A^*$ be a code and let d be a nonnegative integer. X is a *code with finite deciphering delay* (*f.d.d.* in short) d if for any $x, x' \in X,\ x \neq x'$, we have

$$xX^dA^* \bigcap x'X^* = \emptyset.$$

The *delay* of X is the smallest integer d for which this property is verified. If such an integer does not exist we say that X has infinite deciphering delay.

For instance any prefix code is a code with a f.d.d. $d = 0$. As another example, the code $X_d = \{01, (01)^d1\} \subset A^*, A = \{0, 1\}$ is a code with f.d.d. d for any $d \geq 0$. In fact, $(01)X^dA^* \bigcap (01)^d1X^* = \emptyset$. The code $X = \{0, 01, 11\} \subset A^*$ is a code with infinite deciphering delay. In fact, $0(11)^d1 \in 0X^dA^* \bigcap 01X^*$ for all $d \geq 1$. For u in A^* we denote by $\widetilde{u}$ the reverse of u. For $X = \{x_1, x_2, \ldots, x_n\}$, we define the set $\widetilde{X} = \{\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_n\}$. We denote by $Pref(X)$ ($Suff(X)$) the set of prefixes (suffixes) of words in X. For $u \in A^*$, for $k \leq |u|$, we denote by $pref_k(u)$ ($suff_k(u)$) the prefix (suffix) of u of length k and by $suff_{-k}(u)$ the suffix of u of length $|u|-k$.

A finite *sequential transducer* T (cf. [10], [7] Chapter 1) is defined over an input alphabet A and an output alphabet B. It consists of a quadruple $T = (Q, i, F, \delta)$, where Q is a finite set of *states*, i is the unique *initial state*, δ is a partial function $Q \times A \longrightarrow B^* \times Q$ which breaks up into a *next state* function $Q \times A \longrightarrow Q$ and an output function $Q \times A \longrightarrow B^*$. The elements $(p, a, \delta(p, a)) = (p, a, v, q)$, with $p, q \in Q$, $a \in A$ and $v \in B^*$, are called *edges*. In this case we call a the *input label* and v the *output label*. F is a partial function $F : Q \longrightarrow B^*$ called the *terminal function*.

2 Generalization of Girod's Method to f.d.d. Codes

It is well known that a prefix code can be decoded without delay in a left-to-right parsing while it can not be as easily decoded from right to left. In 1999 B. Girod (cf. [6]) introduced a very interesting alternative coding method using finite prefix codes on binary alphabets. It applies a transformation to a concatenation of codewords in a prefix code X that allows the deciphering of the coded word in both directions, by adding to the messages just as many bits as the length the longest word in X. We generalize this method to codes over alphabets with cardinality greater than two and with a f.d.d.

Let $A = \{0, \ldots, n\}$ and f be a map from $A \times A$ into A. For all $a \in A$, we denote by $f_{(a,\cdot)}$ (resp. $f_{(\cdot,a)}$) the map from A into A defined by $f_{(a,\cdot)}(x) = f(a, x)$ (resp. $f_{(\cdot,a)}(x) = f(x, a)$), $\forall x \in A$. They are called the *components* of f. A map $f : A \times A \longrightarrow A$ is said a *latin square map* on A if, for each $a \in A$, its components $f_{(a,\cdot)}$ and $f_{(\cdot,a)}$ are bijective. A latin square map on A is defined by a square array of size $n+1$ where each line and each column contain one and only one occurrence of i, for all $0 \leq i \leq n$.

Example 1. 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

We remark that the bijectivity of the components of a latin square map f implies that there exists only one solution to the equation $f(a,b)=c$, when one element among a, b or c is unknown. We can define two different "inverse" latin square maps associated to a given map f. For $a,b,c \in A$ such that $f(a,b)=c$, we define the inverse maps f_1^{-1}, f_2^{-1} as $f_1^{-1}(c,b)=a$ and $f_2^{-1}(a,c)=b$. The corresponding square arrays can be easily computed in this way:

- for each $c,b \in A$, we define $a=f_1^{-1}(c,b)$ as the index of the row in the b-th column that contains the element c;
- for each $c,a \in A$, we define $b=f_2^{-1}(a,c)$ as the index of the column in the a-th row that contains the element c;

Let $X=\{x_1,\ldots,x_m\}$ be a code with f.d.d. d on an alphabet A, encoding the alphabet $B=\{b_1,\ldots,b_m\}$ by $\gamma(b_i)=x_i$. The words of X are named *codewords*. Let f be a latin square map on A. We extend f to the elements of $A^k \times A^k$, $k \geq 1$, by $f(a_1 \ldots a_k, a'_1 \ldots a'_k) = f(a_1,a'_1)\ldots f(a_k,a'_k)$. Let l be the length of the longest word in X and let x_L be a word in A^+ of length $L=(d+1)l$. Let $b=b_{i_1}\ldots b_{i_t} \in B^*$ and let $y=\gamma(b)=x_{i_1}\ldots x_{i_t}$, with $x_{i_j} \in X$, its encoding. Consider $y'=\widetilde{x}_{i_1}\ldots\widetilde{x}_{i_t}$ where $\widetilde{x}_{i_j}$ represents the reverse of x_{i_j}. Consider the word $z=f(yx_L, \widetilde{x}_L y')$. We define an encoding δ from B^* to A^* by $\delta(b)=z$. We can realize a left-to-right decoding of z by proceeding in the following way:

1. We first consider the words $v:=\widetilde{x}_L$ and $t:=z$, $t':=pref_L(t)$.
2. The word $u=f_1^{-1}(t',v)$ is a prefix of y with length L, then has a prefix which is product of at least $(d+1)$ codewords of X, that is $u=x_{i_1}\ldots x_{i_{d+1}}u'$, with $x_{i_j} \in X$, $u \in A^*$. Since X is a code with a f.d.d. d, we can state that the factorization of u begins with x_{i_1}. Then the decoding of z begins with b_{i_1}.
3. Let $v:=suff_L(v\widetilde{x}_{i_1})$ and let $t:=suff_{-|x_{i_1}|}(t)$ and $t'=pref_L(t)$. Back to step 2 we compute $u=f_1^{-1}(t',v)$, and since u has length L, we are able to recognize the first codeword of u, x_{i_2}. Then the decoding of z begins with $b_{i_1}b_{i_2}$.
4. The algorithm stops when $|t|=L$.

For instance consider the code $X=\{01,012\}$ with f.d.d. $d=1$ encoding $B=\{b_1,b_2\}$. We use the latin square map g of Example 1 and its inverse g_1^{-1} and we choose as key $x_L=011011$. Let $y=(012)(01)(01)(012)(012)$ and $y'=(210)(10)(10)(210)(210)$. The encoding, performed by applying g to the pair $(yx_L, \widetilde{x}_L y')$, gives the sequence $z=2022002211002101101$.

In order to decode z from left to right, we first consider $u=g_1^{-1}(202200,110110)=012010$. Since u has length 6, we are able to recognize the first codeword 012: the decoding of z begins with b_2. After, we consider $v=110210$ and $t'=200221$ and $u=g_1^{-1}(110210,200221)=010101=(01)0101$: the decoding of z begins with b_2b_1. By proceeding this way we get the entire decoding of the message.

We remark that, by reversing the roles of yx_L and $\widetilde{x}_L y'$, we can decode the word form right to left, just by using the information that the first word ends with x_L and that the inverse latin square map g_2^{-1} allows to get b from c and a.

3 Transducers for Decoding

Let $X = \{x_1, \ldots, x_m\} \in A^+$ be a code with f.d.d. d encoding the alphabet $B = \{b_1, \ldots, b_m\}$. Let $x_L \in A^*$ with $|x_L| = L = (l+1)d$, where l is the length of the longest word in X, and let f be a latin square map on A. For any sequence z of codewords in X, consider the encoding δ given by the generalization of Girod's method. The left-to-right decoding method given in the previous section can be described by the transducer, $\mathcal{T}(X)_{f,x_L} = (Q, i, \delta, F)$ defined as follows:

1. Q contains pairs of words (u, v) such that: **a)** $u \in Pref(X^{d+1}) \setminus X^{d+1}$; **b)** v is a a word in $Suff(\widetilde{x}_L)Suff(\widetilde{X}^{d+1})$ of length $L - |u|$.
2. The initial state $i = (\varepsilon, \widetilde{x}_L)$.
3. δ is defined as: **a)** $\delta((u, av), c) = (\varepsilon, (ub, v))$ if $b = f_1^{-1}(c, a)$, $ub \in Pref(X^{d+1})$ and $ub \notin X^{d+1}$; **b)** $\delta((u, av), c) = (b_{i_1}, (x_{i_2} \ldots x_{i_{d+1}}, v\widetilde{x}_{i_1}))$ if $b = f_1^{-1}(c, a)$ and $ub = x_{i_1}x_{i_2} \ldots x_{i_{d+1}} \in X^{d+1}$. In all remaining cases the transitions are undefined.
4. F is defined only for the accessible states of the form (u, v), $u = x_1 \ldots x_d \in X^d$, as the word $b_1 \ldots b_d \in B^d$.

For instance, if $X = \{0, 01\}$ is a code with $d = 1$, encoding $B = \{b_1, b_2\}$ and we use for decoding the map g of Example 1 and the key $x_L = 0101$, we obtain the left-to-right decoding Girod's transducer associated to X in Figure 1:

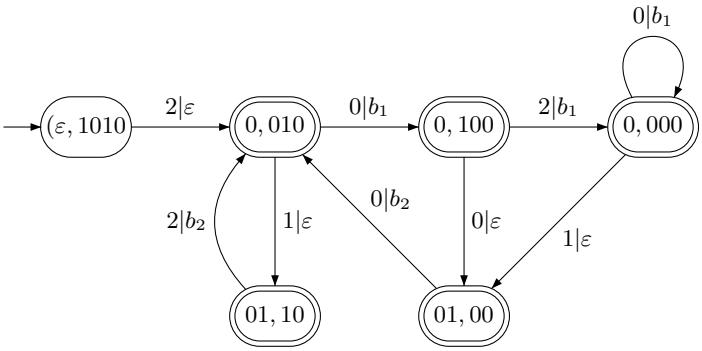

Fig. 1. Transducer T for the left-to-right decoding of $X = \{0, 01\}$ for $x_L = 0101$ over $B = \{b_1, b_2\}$

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

Proposition 31 *Let $\mathcal{T}(X)_{f,x_L}$ (resp. $\widetilde{\mathcal{T}}(X)_{f,x_L}$) be the transducers defined as above. The following results hold:*

1. *$\mathcal{T}(X)_{f,x_L}$ (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, by reading the prefix (resp. the suffix) of length $|z| - L$ of z. Moreover this transducer is deterministic.*
2. *$\mathcal{T}(X)_{f,x_L}$ and $\widetilde{\mathcal{T}}(X)_{f,x_L}$ are isomorphic as unlabelled graphs. If f is a commutative latin square map then they are also isomorphic as transducers.*

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

4 A Remarkable Strongly Connected Component

In this section we examine from the graph theory point of view the transducer $\mathcal{T}(X)_{f,x_L} = (Q, i, F, \delta)$ defined in the previous sections. According to the results of Section 3, given two edges $((u,v), i, c, (u', v'))$, $((u,v), j, d, (u", v"))$ in $\mathcal{T}(X)_{f,x_L}$, if $i \neq j$, then $(u', v') \neq (u", v")$. Then the graph $G(X)_{x_L}$, obtained by removing the labels from the transitions of $\mathcal{T}(X)_{f,x_L}$ is well defined, i.e. there do not exist two different arrows connecting two fixed nodes.

We define $C(G(X)_{x_L})$ as the subgraph of $G(X)_{x_L}$ whose vertices are elements of X^d $(\mathit{Pref}(X) \setminus X) \times (\mathit{Suff}(\widetilde{X}^+) \setminus \{\varepsilon\})$, and whose edges are those in $G(X)_{x_L}$ connecting vertices in the subgraph. It can be proved that given a vertex (u, v) of $C(G(X)_{x_L})$, any vertex accessible from (u, v) is a vertex of $C(G(X)_{x_L})$.

Proposition 41 *The following facts hold:*

1. *$C(G(X)_{x_L})$ is a strongly connected component of $G(X)_{x_L}$.*
2. *Given a code with a f.d.d. X, and 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 $G(X)_{x_L}$.*

Theorem 42 *Given a code X with a f.d.d., a unique graph $C(X)$ exists such that $C(X) = C(G(X)_{x_L})$, for any key x_L.*

References

1. Béal, M.-P., Berstel, J., Marcus, B.H., Perrin, D., Reutenauer, C., Siegel, P.H.: Variable-length codes and finite automata. In: Woungang, I. (ed.) Selected Topics in Information and Coding Theory. World Scientific (to appear)
2. Berstel, J., Perrin, D., Reutenauer, C.: Codes and Automata. Cambridge University Press (2010)
3. Fraenkel, A.S., Klein, S.T.: Bidirectional Huffman Coding. The Computer Journal 33, 296–307 (1990)
4. Giambruno, L., Mantaci, S.: Transducers for the bidirectional decoding of prefix codes. Theoretical Computer Science 411, 1785–1792 (2010)
5. Giambruno, L., Mantaci, S.: On the size of transducers for bidirectional decoding of prefix codes. Rairo-Theoretical Informatics and Applications (2012), doi:10.1051/ita/2012006
6. Girod, B.: Bidirectionally decodable streams of prefix code words. IEEE Communications Letters 3(8), 245–247 (1999)
7. Lothaire, M.: Applied combinatorics on words. Encyclopedia of mathematics and its applications, vol. 104. Cambridge University Press (2005)
8. Lind, D., Marcus, B.: An introduction to Symbolic Dynamics and Coding. Cambridge University Press (1995)
9. Salomon, D.: Variable-Length Codes for Data Compression. Springer (2007)
10. Sakarovitch, J.: Éléments de théorie des automates. Vuibert Informatique (2003)