Publ. Math. Debrecen
78/3-4 (2011), 535–541
DOI: 10.5486/PMD.2011.4716

A novel test for unique decipherability of codes

By JÁNOS FALUCSKAI (Nyíregyháza)

Abstract. Having a set C of codewords w_i we have to decide whether there are two or more sequences of codewords which form the same chain of characters of codewords. A code C is *UD (uniquely decipherable)* code, if every message has at most one factorization with respect to code C, that is, if $x_1x_2\dots x_n = y_1y_2\dots y_m$ holds, where $x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_m \in C$, then $n = m$ and $x_1 = y_1, \dots, x_n = y_n$. We have developed an algorithm that solves this problem by using finite automata in [1]. In this paper we suppose that there is no empty string in the set of coded messages. Thus, we investigate the language C^+. In these cases the automata have more states, but we get more applicable results.

1. Introduction

The decipherability of codes has been investigated by Sardinas and Patterson [2] foremost. It is known as *Sardinas–Patterson* algorithm. The result involved a number of papers by Markov [3], Bandyopadhyay [4], Levenshtein [5], Riley [6], de Luca [7]. The design of an efficient algorithm is described by Spehner [8], Rodeh [9], Apostolico and Giancarlo [10], McCloskey [11]. The complexity of the algorithm which decides the decipherability of a code is not known, but Hoffmann [12], Galil [13], McCloskey [11] and König [14] have reached useful results. The concept of decipherability was extended to infinite words and their languages in [15], [16], [17] and [18].

In this paper we give a novel algorithm based on automata theory. The unique decipherability of the codes is a special problem in the automata theory, namely we have to test whether a given rational expression is unambiguous.

Mathematics Subject Classification: 94B35, 94A45, 68Q45.
Key words and phrases: decipherability, automaton.

Standard decision procedures exist concerning this question, see Eilenberg [19], or *Aho* et al. The connection between codes and regular expressions has been pointed out by Brzozowsky [20].

In general we can say that a code is a set of sequences of letters. In the course of coding we assign various codewords to the letters of the source messages, this process is called verbatim coding. We can code by non verbatim methods, but the decipherability of a code does not depend on the alphabet of the source message and the methods. So the decipherability only depends on the set of codewords. A verbatim coding is given by $f : \Sigma \to \Delta^+$, where Σ is the alphabet of the source message and the set Δ is the alphabet of the code. The coding defined by f will be decipherable, if the following criterion is fulfilled:

$$f(x_1)\dots f(x_n) = f(y_1)\dots f(y_m) \Rightarrow n = m \quad \text{and} \quad f(x_i) = f(y_i),\ x_i = y_i,$$

where $x_i, y_i \in \Sigma,\ f(x_i), f(y_i) \in \Delta^+$.

The mapping f is homomorphic by the definition of verbatim coding and the properties of the concatenation. Thus, we can formulate with the following way: $g : \Sigma^+ \to \Delta^+$. In this case the decipherability means the following:

$$g(x_1 \dots x_n) = g(y_1 \dots y_m) \text{ implies } n = m \text{ and } x_i = y_i\, \forall i \le n,$$
$$\text{where } x_i, y_i \in \Sigma,\ g(x_1 \dots x_n), g(y_1 \dots y_m) \in \Delta^+$$

That is, the mapping g is isomorphic.

2. A finite automaton of code

Our algorithm is based on theory of finite automata. Using automata of codewords we can construct an automaton for the code C over alphabet Δ. If the codeword $w_i \in C$ is $x_1x_2\dots x_n$, $x_j \in \Delta$, then the automaton $\mathcal{A}(\{w_i\})$ will be $\mathcal{A}(\{w_i\}) = (Q^{(i)}, q_\lambda, Q_F^{(i)}, A, \delta^{(i)})$. The set $Q^{(i)}$ is the set of states where the state q_λ is the initial state of the automaton $\mathcal{A}(\{w_i\})$ and the singleton $Q_F^{(i)}$ is the set of final state. $Q_F^{(i)} = \{i\}$ and $|Q^{(i)}| = length(w_i) + 1$. Since, the transition rules of automaton $\mathcal{A}(\{w_i\})$ are the following:

$$\delta(q_\lambda, x_1) = q_{x_1}$$
$$\delta(q_{x_1}, x_2) = q_{x_1x_2}$$
$$\vdots$$

$$\delta(q_{x_1x_2\dots x_{n-2}}, x_{n-1}) = q_{x_1x_2\dots x_{n-2}x_{n-1}}$$
$$\delta(q_{x_1x_2\dots x_{n-1}}, x_n) = i$$

Thus, the automaton $\mathcal{A}(\{w_i\})$ accepts the codeword w_i.

Let us consider the nondecipherable sequences of codewords. It is obvious, that the different factorizations contain at least two codewords where one of them is prefix part of the other. That is, if $u_1 \dots u_n = w_1 \dots w_m$, then there is a number k such that $u_i = w_i$, but $u_k \neq w_k$ for any $i < k$, where $1 \leq k \leq \min\{n, m\}$. If we join the automata of codewords by the method above, then we will obtain the automaton $\mathcal{A}(\{w_1, \dots, w_n\})$ for the code $C = \{w_1, \dots, w_n\}$. We can use a shorter notation $\mathcal{A}(C)$, too. Denote $x_1^{(l)}$ the first symbol of the l-th codeword. Thus,

$$\mathcal{A}(C) = (q, Q_F = \{1, \dots, n\}, Q = Q^{(1)} \cup \dots \cup Q^{(n)}, A, \delta),$$

where

$$\delta = \delta^{(1)} \cup \dots \cup \delta^{(n)} \cup \{\delta(1, x_1^{(1)}) = q_{x_1^{(1)}}, \dots, \delta(1, x_1^{(n)}) = q_{x_1^{(n)}}, \dots,$$
$$\delta(n, x_1^{(1)}) = q_{x_1^{(1)}}, \dots, \delta(n, x_1^{(n)}) = q_{x_1^{(n)}}\}.$$

Obviously, the automaton $\mathcal{A}(C)$ accepts exactly the language C^+.

Theorem 2.1. *If the automaton $\mathcal{A}(C)$ is deterministic, then the code C will be decipherable.*

Proof. Recall that each prefix code is decipherable. Therefore, it is enough to show, that the code C is prefix whenever $\mathcal{A}(C)$ is deterministic.

Suppose that, contrary of our statement, $\mathcal{A}(C)$ is deterministic and C is not prefix. Then there are two codewords w_i and w_j such that $w_i = w_j\alpha$, where $\alpha \in \Delta^+$. In more details, there are $x_1, \dots, x_{|w_j|}, \dots, x_{|w_i|} \in \Delta$ such that $w_j = x_1 \dots x_{|w_j|}$ and $w_i = x_1 \dots x_{|w_j|} \dots x_{|w_i|}$. By our constructions, this implies that $\delta(q_{x_1\dots x_{|w_j|-1}}, x_{|w_j|}) = q_{x_1\dots x_{|w_j|}} \notin Q_F$ and there exists $j \in Q_F$ with $\delta(q_{x_1\dots x_{|w_j|-1}}, x_{|w_j|}) = j$ for which $j \neq q_{x_1\dots x_{|w_j|}}$. But then $\mathcal{A}(C)$ is nondeterministic. Therefore, if $\mathcal{A}(C)$ is deterministic then C is prefix as we stated. $\square$

There are codes which are nonprefix, but decipherable. For example the code $C = \{01, 0100\}$. That is, the automaton of decipherable codes can be nondeterministic one. Thus, the Theorem 2.1 is not reversible. We demonstrate the graphical presentation of the automaton $\mathcal{A}(\{01, 0100\})$ in Figure 1.

The automaton $\mathcal{A}(\{01, 0100\})$ is nondeterministic because of the rule

$$\delta(q_0, 1) = \{1, q_{01}\}$$

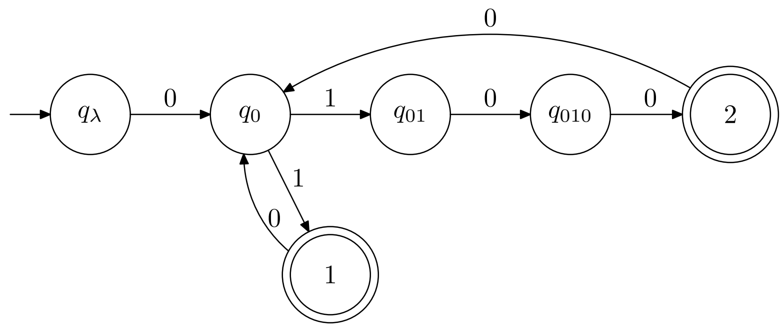

Figure 1. The automaton $\mathcal{A}(\{01, 0100\})$

But, the code $\{01, 0100\}$ is decipherable. If we use our construction, then the automata of the nondecipherable codes will be nondeterministic.

The condition of the Theorem 2.1 is sufficient. Next, we give a necessary and sufficient condition of the decipherability using a well-known algorithm for constructing an appropriate deterministic finite automaton of a nondeterministic finite automaton. (See, for example, [21]).

Let $\mathcal{A} = (Q, q_\lambda, Q_F, A, \delta)$ be a nondeterministic finite automaton with set of states Q, initial state q_λ, set of final states Q_F, input alphabet A, transition function $\delta : Q \times A \to 2^Q$.[1] It is said that $\mathcal{A}$ accepts the word $x_1 x_2 \ldots x_n \in \Sigma^*, x_1, \ldots, x_n \in \Sigma$ if there are $q_1, \ldots, q_{n-1} \in Q, q_n \in Q_F$ with $q_1 \in \delta(q_\lambda, x_1), q_2 \in \delta(q_1, x_2), \ldots, q_{n-1} \in \delta(q_{n-2}, x_{n-1}), q_n \in \delta(q_{n-1}, x_n)$. In particular, we say that $\mathcal{A}$ accepts the empty word if $q_\lambda \in Q_F$. The set of all words accepted by $\mathcal{A}$ is called the language accepted by $\mathcal{A}$. Two nondeterministic automata is called equivalent if they accept the same language.[2]

Theorem 2.2. [21] *Let L be a set accepted by a nondeterministic finite automaton $\mathcal{A}$. Then there exists a deterministic finite automaton $\mathcal{A}'$ that accepts L.*

Consider the automaton $\mathcal{A}$ and define a deterministic finite automaton, $\mathcal{A}' = (Q', q'_\lambda, Q'_F, A, \delta')$ as follows. The states of $\mathcal{A}'$ are all the subsets of the set of states of $\mathcal{A}$. That is, $Q' = 2^Q$. Q'_F is the set of all states in Q' containing a state of Q_F. An element of Q' will be denoted by $[q_1, q_2, \ldots, q_i]$ where $q_1, q_2, \ldots, q_i$ are in Q.

[1] If for every pair $q \in Q, x \in A, \delta(q, x)$ has at most one element then $\mathcal{A}$ can be considered as a deterministic finite automaton.

[2] We can also consider a deterministic finite automaton as a special nondeterministic one. Thus the concept of equivalence can be extended for deterministic finite automata in natural way.

Note that $q'_\lambda = [q_\lambda]$. We define

$$\delta'([q_1, q_2, \ldots, q_i], a) = [p_1, p_2, \ldots, p_j]$$

if and only if

$$\delta(\{q_1, q_2, \ldots, q_i\}, a) = \{p_1, p_2, \ldots, p_j\}.$$

That is, δ' applied to an element R of Q' is computed by applying δ to each state of Q represented by $R = [q_1, q_2, \ldots, q_i]$. On applying δ to each of $q_1, q_2, \ldots, q_i$ and taking the union, we get some new set of states, $p_1, p_2, \ldots, p_j$. This new set of states has a representative, $[p_1, p_2, \ldots, p_j]$ in Q', and that element is the value of $\delta'([q_1, q_2, \ldots, q_i], a)$. We say that the $\mathcal{A}$' is the deterministic finite automaton of the nondeterministic finite automaton $\mathcal{A}$. By this concept, we can derive the following result from Theorem 2.2.

Theorem 2.3. [22] *Every finite automaton equivalent to its deterministic finite automaton.*

If we have the string $v \in C^+$, then the automaton $\mathcal{A}(C)$ will accept v. That is, the automaton $\mathcal{A}(C)$ will read v and get to a final state. If the code C is not uniquely decipherable, then we can follow different paths during reading v. We join these different paths by the equivalent deterministic automaton.

Let us fix an arrangement $w_1, \ldots, w_n$ of the elements of C and for every $w \in C$, put $\mathcal{N}(w) = k$ if and only if $w = w_k$ $(k \in \{1, \ldots, n\})$.

$\mathcal{N}(w)$ is a final state of the automaton $\mathcal{A}(C)$ by our construction. Let $u_1 \ldots u_n = v_1 \ldots v_m$, where $u_i, v_j \in C$. Assume that the index k is the smallest number for which $u_k \neq v_k$. Then $u_{k+1} \ldots u_n \neq v_{k+1} \ldots v_m$ holds. Let d_1 and d_2 be such that if $\forall\, 1 \leq i < d_1$ and $\forall\, 1 \leq j < d_2$, then $u_k \ldots u_{k+i} \neq v_k \ldots v_{k+j}$ holds, but $u_k \ldots u_{k+d_1} = v_k \ldots v_{k+d_2}$. Thus, $u_{k+d_1} \neq v_{k+d_2}$ because of the definitions of d_1 and d_2. Since, if $u_{k+d_1} = v_{k+d_2}$ holds, then $u_k \ldots u_{k+d_1-1} = v_k \ldots v_{k+d_2-1}$. But, this is a contradiction. Denote q_λ the initial state. Thus, the path $q_\lambda \xrightarrow{u_1 \ldots u_{k+d_1}} \mathcal{N}(u_{k+d_1})$ and the path $q_\lambda \xrightarrow{v_1 \ldots v_{k+d_2}} \mathcal{N}(v_{k+d_2})$ are different paths of the automaton $\mathcal{A}(C)$. Let us construct the deterministic automaton $\mathcal{A}_D(C)$ for the automaton $\mathcal{A}(C)$. The two paths are joined in the automaton $\mathcal{A}_D(C)$. They lead to a state which contains the state $\mathcal{N}(u_{k+d_1})$ and the state $\mathcal{N}(v_{k+d_2})$.

Therefore, if we have two (or more) factorization of a string, then there is a state of deterministic automaton which contains at least two final states of nondeterministic automaton. Denote $Q_F^{\mathcal{A}(C)}$ the set of final states of the automaton $\mathcal{A}(C)$.

Theorem 2.4. *A code is decipherable if, and only if in the automaton* $\mathcal{A}_D(C)$ *at most one element of* $Q_F^{\mathcal{A}(C)}$ *appears on the right side of any transaction rule. That is, for any transaction rule the following holds: if the transaction rule* $\delta(\{q_{i_1},\dots,q_{i_n}\},x)=\{q_{j_1},\dots,q_{j_m}\}$ *is in the automaton* $\mathcal{A}_D(C)$*, then there is no* $l\neq k$ *such that,* $q_{j_l}\in Q_F^{\mathcal{A}(C)}$ *and* $q_{j_k}\in Q_F^{\mathcal{A}(C)}$ *hold.*

Proof. The proof is carried out indirectly. Assume that a code is decipherable and there is a state of deterministic automaton that contains at least two final states of nondeterministic automaton. That is, there is a rule $\delta(\{q_{i_1},\dots,q_{i_n}\},x)=\{q_{j_1},\dots,q_{j_m}\}$ in the automaton $\mathcal{A}_D(C)$ and $l\neq k$ exists such that, $q_{j_l}\in Q_F^{\mathcal{A}(C)}$ and $q_{j_k}\in Q_F^{\mathcal{A}(C)}$ hold. Denote v the sequence of symbols which is touched along the path from the initial state q_λ to the state $\{q_{j_1},\dots,q_{j_m}\}$. That is, $q_\lambda\xrightarrow{v}\{q_{j_1},\dots,q_{j_m}\}$. Thus, the paths $q_\lambda\xrightarrow{v}q_{j_l}$ and $q_\lambda\xrightarrow{v}q_{j_k}$ are different successful paths in the nondeterministic automaton. That is, the sequence v has two different factorization. Therefore, the code is nondecipherable. We have a contradiction and the 'if' part is proved.

To prove the 'only if' part we assume the following: There is no state of the deterministic automaton which contains at least two final states of nondeterministic automaton and the code is nondecipherable. Thus, if the code is nondecipherable, then there is at least two sequences of the codewords such that $w_{i_1}\dots w_{i_n}=w_{j_1}\dots w_{j_m}$ and $w_{i_n}\neq w_{j_m}$. If we read the sequences, then we get to the same state because of the automaton is deterministic. Therefore, this state contains the final states i_n and j_m of the nondeterministic automaton because of the codewords w_{i_n} and w_{j_m}. We have a contradiction and the theorem is proved. $\square$

References

[1] J. Falucskai, On a test for codes, *Acta Math. Acad. Paedagog. Nyíregyháziensis* **22** (2006), 121–125.

[2] A. A. Sardinas and C. W. Patterson, A necessary and sufficient condition for the unique decomposition of coded messages, *IRE Internat. Conv. Rec.* **8** (1953), 104–108.

[3] Al. A. Markov, On alphabet coding, *Soviet. Phys. Dokl.* **6** (1962), 553–554.

[4] G. Bandyopadhyay, A simple proof of the decipherability criterion of Sardinas and Patterson, *Infor. and Control* **6** (1963), 331–336.

[5] V. I. Levenšteĭn, Some properties of coding and self-adjusting automata for decoding messages, *Problemy Kibernet.* **11** (1964), 63–121.

[6] J. A. Riley, The Sardinas-Patterson and Levenshtein theorems, *Inform. and Control* **10** (1967), 120–136.

[7] A. de Luca, A note on variable length codes, *Inform. and Control* **32** (1976), 263–271.

[8] J. C. Spehner, Quelques problémes d'extension, de conjugaison et de présentation des sous-monoïndes d'un monoïde libre, PhD thesis, *Université Paris*, *Paris*, 1976.

[9] M. Rodeh, A fast test for unique decipherability based on suffix trees, *IEEE Trans. Inform. Theory* **IT-28** (1982), 648–651.

[10] A. Apostolico and R. Giancarlo, Pattern matching machine implementation of a fast test for unique decipherability, *Inform. Process. Lett.* **18** (1984), 155–158.

[11] R. McCloskey, An $O(n^2)$ time algorithm for deciding whether a regular language is a code, Vol. 2, Proceedings of the 8th International Conference of Computing and Information, ICCI'96 (Waterloo, ON), *J. Comput. Inf.*, 1996, 79–89.

[12] C. M. Hoffmann, A note on unique decipherability, Vol. 176, Mathematical foundations of computer science, 1984 (Prague, 1984), Lecture Notes in Comput. Sci., *Springer*, *Berlin*, 1984, 50–63.

[13] Z. Galil, Open problems in stringology, Vol. 12, Combinatorial algorithms on words (Maratea, 1984), NATO Adv. Sci. Inst. Ser. F Comput. Systems Sci., *Springer*, *Berlin*, 1985, 1–8.

[14] R. König, Lectures on codes, Internal Reports of the IMMD I, *Friedrich Alexander University of Erlangen-Nürnberg*, 1994.

[15] H. Jürgensen and S. Konstantinidis, Codes, Handbook of Formal Languages, Vol. I, NATO Adv. Sci. Inst. Ser. F Comput. Systems Sci., (G. Rozenberg and A. Salomaa, eds.), *Springer*, *Berlin*, 1997, 511–607.

[16] J. Devolder and E. Timmerman, Finitary codes for bi-infinite words, *RAIRO Inform. Théor Appl.* **26** (1992), 363–386.

[17] J. Devolder, Precircular codes and periodic biinfinite words, *Inform and Comput.* **107** (1993), 185–201.

[18] J. Devolder, M. Latteux, I. Litovski and L. Staiger, Codes and infinite words, *Acta Cybernet.* **11** (1994), 241–256.

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

[20] J. A. Brzozowski, Roots of star events, *J. Assoc. Comput. Mach.* **14** (1967), 466–477.

[21] J. E. Hopcroft and J. D. Ullman, Formal languages and their relation to automata, *Addison-Wesley*, *London*, 1969.

[22] A. Salomaa, Formal languages, *Academic Press Inc.*, *New York*, 1973.

JÁNOS FALUCSKAI
INSTITUTE OF MATHEMATICS AND INFORMATICS
COLLEGE OF NYÍREGYHÁZA
SÓSTÓI U. 31/B
H-4400 NYÍREGYHÁZA
HUNGARY

E-mail: falu@nyf.hu

(Received September 9, 2009; revised January 18, 2011)