Proceedings of the 7th International Conference on Applied Informatics
Eger, Hungary, January 28–31, 2007. Vol. 2. pp. 229–235.

Some algorithms concerning uniquely decipherable codes

János Falucskai

Department of Mathematic and Informatics, College of Nyíregyháza
e-mail: falu@nyf.hu

Abstract

The following problem plays an important role in code theory and its applications: Having a set of codewords we have to decide whether there are two or more sequences of codewords which form the same chain of characters of codewords. The problem can be approached in various ways, so the algorithms concerning uniquely decipherable codes use different devices for testing this property. The algorithm of Sardinas–Patterson is based on sequences of sets, other algorithms solve this problem by using finite automata. The purpose of this paper is to show the common root of different algorithms.

Keywords: Uniquely decipherable codes, automata, length-variable codes.

MSC: 94B35, 94A45, 68Q45.

1. The algorithm of Sardinas – Patterson

The algorithm of Sardinas – Patterson is based on the following: Let us compute all the remainders in all attempts at a double factorization. It can recognize a double factorization by the fact that the empty word is one of the remainders.

Let A be a set, which we call an *alphabet*. A *word* w on the alphabet A is a finite sequence of elements of A

$$w = (a_1, a_2, \ldots, a_n), \quad a_i \in A$$

The set of all words on the alphabet A is denoted by A^*. If we omit the empty word from A^* then we get A^+. Let X and Y be two subsets of A^+ and let $x \in X$ and $y \in Y$. Denote $X^{-1}Y$ the following set: w is an element of $X^{-1}Y$ if $xw = y$.

Let C be a subset of A^+, and let

$$\begin{aligned} U_1 &= C^{-1}C \setminus \{\varepsilon\} \\ U_2 &= C^{-1}U_1 \cup U_1^{-1}C \\ &\vdots \\ U_{n+1} &= C^{-1}U_n \cup U_n^{-1}C \end{aligned} \tag{1.1}$$

Thus we have:

Theorem 1.1 (See [1]). *Let $C \subset A^+$ and let $(U_n)_{n\geqslant 1}$ be defined as above. For all $n \geqslant 1$ and $k \in \{1, \ldots, n\}$, we have $\varepsilon \in U_n$ if and only if there exists a word $u \in U_k$ and integers $i, j \geqslant 0$ such that*

$$uC^i \cap C^j \neq 0, \quad i + j + k = n \tag{1.2}$$

Proof. (See [1].) We prove the statement for all n by descending induction on k. Assume, first $k = n$. If $\varepsilon \in U_n$, taking $u = 1, i = j = 0$, the equation above is satisfied. Conversely, if (1.2) holds, then $i = j = 0$. This implies $u = 1$ and consequently $\varepsilon \in U_n$. Now let $n > k$, and suppose that the equivalence holds for $n, n-1, \ldots, k+1$. If $\varepsilon \in U_n$, then by induction hypothesis, there exists $v \in U_{k+1}$ and two integers $i, j \geqslant 0$ such that

$$uC^i \cap C^j \neq 0, \qquad i + j + k + 1 = n$$

Thus there are words $x \in C^i; y \in C^j$ such that $vx = y$. Now $v \in U_{k+1}$, and there are two cases. Either there is a word $z \in C$ such that

$$zv = u \in U_k$$

or there exists $z \in C; u \in U_k$ such that

$$z = uv$$

In the first case, one has $ux = zy$, thus

$$uC^i \cap C^{j+1} \neq 0, u \in U_k$$

In the second case, one has $zx = uvx = uy$, thus

$$uC^j \cap C^{i+1} \neq 0, u \in C_k$$

In both cases, formula (1.2) is satisfied. Conversely, assume that there are $u \in U_k$ and $i, j \geqslant 0$ with

$$uC^i \cap C^j \neq 0, \qquad i + j + k + 1 = n$$

Then

$$ux_1x_2 \cdots x_i = y_1y_2 \cdots y_j$$

for some $x_r, y_s \in C$. If $j = 0$, then $i = 0$ and $k = n$. Thus $j \geqslant 1$. Once more, we distinguish two cases, according to the length of u compared to the length of y_1. If $u = y_1v$ for some $v \in A^+$, then $v \in C^{-1}U_k \subset U_{k+1}$ and further

$$vx_1x_2 \cdots x_i = y_2 \cdots y_j$$

Thus $vC^i \cap C^{j-1} \neq 0$ and by the induction hypothesis $\varepsilon \in U_n$. If $y_1 = uv$ for some $v \in A^+$, then $v \in U_k^{-1}C \subset U_{k+1}$ and

$$x_1x_2 \cdots x_i = vy_1y_2 \cdots y_j$$

showing that $C^i \cap vC^{j-1} \neq 0$. Thus again $\varepsilon \in U_n$ by the induction hypothesis. This concludes the proof. □

Theorem 1.2 (See [1]). *The set $C \subset A^+$ is a uniquely decipherable code if and only if none of the sets U_n defined above contains the empty word.*

Proof. (See [1].) The proof of theorem is based on the previous lemma. If X is not a code, then there is a relation

$$x_1x_2\cdots x_n = y_1y_2\cdots y_m; x_i, y_j \in C; x_1 \neq y_1$$

Assume $|y_1| < |x_1|$. Then $x_1 = y_1u$ for some $u \in A^+$. But then

$$u \in U_1 \text{ and } uC^{n-1} \cap C^{m-1} \neq 0$$

According to the lemma, $\varepsilon \in U_{n+m-1}$. Conversely, if $\varepsilon \in U_n$, take $k = 1$ in the lemma. There exists $u \in U_1$ and integers $i, j \geqslant 0$, such that $uC^i \cap C^j \neq 0$. Now $u \in U_1$ implies that $xu = y$ for some $x, y \in C$. Furthermore $x \neq y$ since $u \neq \varepsilon$. It follows from $xuC^i \cap xC^j \neq 0$ that $yC^i \cap xC^j \neq 0$, showing that C is not a code. This establishes the theorem. □

Example 1.3. $K = \{00, 01, 011, 100\}$

$$U_1 = K^{-1}K \setminus \{\varepsilon\} = \{1\}$$

$$U_2 = K^{-1}U_1 \cup U_1^{-1}K = \{00\}$$

$$U_3 = K^{-1}U_2 \cup U_2^{-1}K = \{\varepsilon\}$$

Since U_3 contains the empty word, the code K is not a uniquely decipherable code.

2. Kayoko Tsuji's algorithm

Let us construct an automaton A_K for set K: $L(A_K)$ is the set of all ambiguous words in K.

Theorem 2.1 (See [2]). *The set K is a uniquely decipherable code if and only if $L(A_K)$ is empty.*

The automaton is constructed by the following way:
$P(K) = \{p \in K \,|\, pq \in K, q \neq \varepsilon\}$
$!\,\varphi : P(K) \to \mathbb{N} : 1 \leqslant \varphi(p) \leqslant card(P(K))$

$S(K) = \{s \in A^+ \,|\, qs \in K, q, s \neq \varepsilon\}$
$!\,\psi : S(K) \to \mathbb{N} : card(P(K)) + 1 \leqslant \psi(p)$

S: initial state
$\varphi(P(K))$: inner states
$S \xrightarrow{p} \varphi(p)$: path

if $u = p^{-1}x$, then $\varphi(p) \xrightarrow{u} \psi(u)$:path and $\psi(u)$ inner state
if $uv = x_1 \cdots x_m$ and $\exists w : wv = x_m$, then $\psi(u) \xrightarrow{v} \psi(v)$:path and $\psi(v)$ inner state
$\psi(S(K) \cap K) \cap Q$: terminal states

Example 2.2. $K = \{01, 00, 011, 100\}$
$P(K) = \{01\}$
$S(K) = \{1, 0, 11, 00\}$
$\varphi(01) = 1,\ \psi(1) = 2, \psi(00) = 3, \psi(11) = 4, \psi(0) = 5$
$S \xrightarrow{01} \varphi(01) = 1, \quad 1 \xrightarrow{1} \psi(1) = 2, \quad 2 \xrightarrow{00} \psi(00) = 3$
States:$S, 1, 2, 3$
Terminal states: $\psi(S(K) \cap K) \cap Q =$
$= \psi(\{1, 0, 11, 00\} \cap \{01, 00, 011, 100\}) \cap \{S, 1, 2, 3\} =$
$= \psi(00) \cap \{S, 1, 2, 3\} = \{3\} \cap \{S, 1, 2, 3\} = \{3\}$

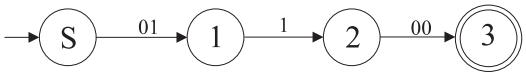

Figure 1: Kayako Tsuji's automaton for code $K = \{01, 00, 011, 100\}$

3. Our automaton to test codes

We construct an automaton for the code over A by union of automata of codewords. If $w = x_1x_2 \cdots x_n$ is a codeword then the automaton $\mathcal{A}(w)$ of codeword w is $\mathcal{A}(w) = (Q, q_i, Q_t, A, \delta)$ where q_i is the initial state of $\mathcal{A}(w)$ and Q_t is the set of terminal states. Q is the set of states and $Q_t = \{q_i\}; \quad q_i \in Q; \quad card(Q) = length(w)$ since the rules of automaton $\mathcal{A}(w)$ are the following:

$$\begin{array}{lcl} \delta(q_i, x_1) & = & q_{x_1} \\ \delta(q_{x_1}, x_2) & = & q_{x_1x_2} \\ & \vdots & \\ \delta(q_{x_1x_2\cdots x_{n-2}}, x_{n-1}) & = & q_{x_1x_2\cdots x_{n-2}x_{n-1}} \\ \delta(q_{x_1x_2\cdots x_{n-1}}, x_n) & = & q_i \end{array}$$

thus, $\mathcal{A}(w)$ can recognize w^*.

If we join the automata of codewords, then we get the automaton $\mathcal{A}(w_1, \ldots, w_n)$ of code $C = \{w_1, \ldots, w_n\}$. We can use notation $\mathcal{A}(C)$, too. So

$$\mathcal{A}(C) = (Q = Q^{w_1} \cup \cdots \cup Q^{w_n}, q_i, Q_t = \{q_i\}, A, \delta = \delta^{w_1} \cup \cdots \cup \delta^{w_n})$$

Obviously, $\mathcal{A}(C)$ accepts C^*. An automaton is non deterministic if there is more then one rule for the same pair of state and symbol.

If a string S decipherable on code C then $\mathcal{A}(C)$ accepts S, namely $\mathcal{A}(C)$ read it and stay in q_i state. If S is not uniquely decipherable then we can follow different paths during reading. We join these different paths by the equivalent deterministic automaton. Formally

$$\overbrace{\underbrace{x_1 \dots x_{|w_{i_1}|}}_{w_{i_1}} \dots x_{|w_{j_1}|}}^{w_{j_1}} \cdots \overbrace{\cdots\cdots\cdots\cdots\cdots\cdots\underbrace{\cdots x_{|S|}}_{w_{i_m}}}^{w_{j_n}}$$

$$\begin{aligned}
&\delta(q_i, x_1) &&= q_{x_1}\\
&\delta(q_{x_1}, x_2) &&= q_{x_1x_2}\\
&&&\vdots\\
&\delta(q_{x_1x_2\dots x_{|w_{i_1}|-1}}, x_{|w_{i_1}|}) &&= \{q_{x_1x_2\dots x_{|w_{i_1}|}}, q_i\}\\
&\delta(\{q_{x_1x_2\dots x_{|w_{i_1}|}}, q_i\}, x_{|w_{i_1}|+1}) &&= \{q_{x_1x_2\dots x_{|w_{i_1}|+1}}, q_{x_{|w_{i_1}|+1}}\}\\
&&&\vdots\\
&\delta(\{q_{x_1x_2\dots x_{|w_{j_1}|-1}}, q_{x_{|w_{i_1}|+1}\dots x_{|w_{j_1}|-1}}\}, x_{|w_{j_1}|}) &&= \{q_i, q_{x_{|w_{i_1}|+1}\dots x_{|w_{j_1}|}}\}\\
&&&\vdots\\
&\delta(\{q_{x_{|w_{j_n}|+1}\dots x_{|S|-1}}, q_{x_{|w_{i_m}|+1}\dots x_{|S|-1}}\}, x_{|S|}) &&= \{q_i, q_i\} = q_i
\end{aligned}$$

Thus two (or more) factorizations of a string will be ended by using two (or more) rules with right side q_i.

Theorem 3.1. *A code is uniquely decipherable if and only if at the most one state equal to q_i in right side of any rule of $\mathcal{A}_D(C)$, namely*

$$\forall\, \delta(\{q_{i_1}, \dots, q_{i_n}\}, x) = \{q_{j_1}, \dots, q_{j_m}\} \in \mathcal{A}_D(C) : \nexists \quad l, k : q_{j_l} = q_{j_k} = q_i.$$

Proof. The proof is indirect. If there exists

$$\delta(\{q_{i_1}, \dots, q_{i_n}\}, x) = \{q_{j_1}, \dots, q_{j_m}\} \in \mathcal{A}_D(C) : \exists \quad l, k : q_{j_l} = q_{j_k} = q_i$$

then there is a string with at least two different factorizations which is accepted by the automaton. Consequently the code is not uniquely decipherable. □

Example 3.2. Let $C = \{00, 01, 011, 100\}$. Thus we get $\mathcal{A}(00, 01, 011, 100)$ in Figure 2. ($q_i = S$). Let us construct $\mathcal{A}_D(00, 01, 011, 100)$. The result is given in Figure 3. We can see that

$$\delta(\{BC\}, 0) = \{S, S\} = \{S\}$$

so the code is not uniquely decipherable.

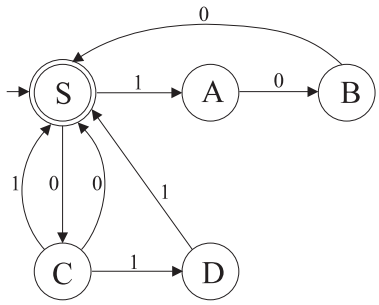

Figure 2: Automaton $\mathcal{A}(00, 01, 011, 100)$

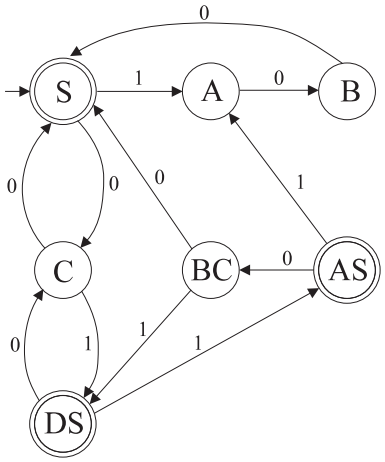

Figure 3: Automaton $\mathcal{A}_D(00, 01, 011, 100)$

4. Summary

We presented three algorithms in the article. Any algorithm of them can solve our problem concerning uniquely decipherable codes. Let us see the following relations among the algorithms.

Using the *Sardinas–Patterson* procedure (Section 1) we received the following

(Example 1.3):

$$\begin{aligned} \text{Common prefix of } 01, 011 = &\quad \{01\} \\ U_1 = &\quad K^{-1}K \setminus \{\varepsilon\} = \{1\} \\ U_2 = &\quad K^{-1}U_1 \cup U_1^{-1}K = \{00\} \\ U_3 = &\quad K^{-1}U_2 \cup U_2^{-1}K = \{\varepsilon\} \end{aligned}$$

So the "set-sequence" is:

$$\xrightarrow{01} \{U_1\} \xrightarrow{1} \{U_2\} \xrightarrow{00} \{U_3\}$$

Using the *Kajako Tsuji* procedure (Section 2) we received the following (Example 2.2):

$$S \xrightarrow{01} \{1\} \xrightarrow{1} \{2\} \xrightarrow{00} \{3\}$$

Using our procedure (Section 3) we obtained the following not uniquely decipherable path in automaton (3) (we consider only those states that contain S):

$$S \xrightarrow{01} \{DS\} \xrightarrow{1} \{AS\} \xrightarrow{00} \{S\}$$

It can be established that the result of each algorithm includes the following character sequence:

$$\xrightarrow{01} \quad \xrightarrow{1} \quad \xrightarrow{00}$$

References

[1] Berstel, J., Perrin, D., Theory of codes, *Pure and Applied Mathematics*, Vol. 117 (1985).

[2] Tsuji, Kayoko, An automaton for deciding whether a given set of words is a code, *RIMS Kokyuroku 1222*, (2001), 123–127.