Schedae Informaticae Vol. 30 (2021): 9–15
doi: 10.4467/20838476SI.21.001.14382

On Some Cryptographic Protocol

Wit Foryś
Faculty of Applied Mathematics, AGH University of Science and Technology
al. Mickiewicza 30, 30-059 Kraków, Poland
e-mail: *wforys@agh.edu.pl*

Abstract. In this paper we present a cryptographic protocol for a seller – buyer problem and in particular we prove that non-emptiness of a semi-commutative set defined by mappings involved in the protocol is a decidable problem.

Keywords: cryptographic protocol, key-code, semi-commutativity set

1. Introduction

Let us consider the following problem. There are a seller S and a buyer B. The seller has some secrets and is offering them for sale and the buyer wants to buy one. It is assumed that B is not willing to disclose to S which secret he wants and that B after buying the secret should not learn more than this chosen secret only.
The problem, usually described as a cryptographic protocol, has generated a research on left inverses of mappings, mainly morphisms and semi-commutativity sets – see [5, 6].

In this paper, on the one hand we limited our considerations, assuming that key-code morphisms are used in the protocol, and on the other hand, we have generalized our considerations by introducing so called rational relations in place of left-inverses of morphisms.

2. Formalization of the Protocol

Let us formalize the mentioned above cryptographic protocol.

Let us denote secrets as words

$$w_1, ..., w_n.$$

The seller S has his own encryption function E_S and decryption funtion D_S such that for each word w under consideration

$$D_S(E_S(w)) = w.$$

Similarly, the buyer B has his encryption and decryption functions E_B, D_B with the above property. A protocol could be executed according to the following three steps.

1. S gives B the sequence
$$E_S(w_1), ..., E_S(w_n)$$
(descriptions and/or identifiers of secrets).
2. If B wants to buy the i-th secret he sents S the value $E_B(E_S(w_i))$.
3. After receiving a specified amount of money S sends B the value $D_S E_B E_S(w_i)$.

If the following equality holds

$$(*) \;\; D_B D_S E_B E_S(w_i) = w_i$$

B has learned the secret he chose. Notice that the additional cryptographic requirements are fulfilled. B should not be able to learn more than this chosen secret and S should not be able to determine which secret was chosen by B. It is worth noting that in order to ensure the latter requirement, encryption functions should not be made public. If not, the seller A could compute values $E_B(E_S(w_k))$ for $k = 1, ..., n$ and compare them with the obtained from B value $E_B(E_S(w_i))$ finally deducing the secret which S bought.

3. Preliminaries

We assume the reader is familiar with the basic notions and concepts from theories of formal languages and codes. For some properties of the introduced in this section notions see [1, 2, 4]

Let A^* denote a free monoid generated by a finite set A. The length of a word $w \in A^*$ is defined to be the number of letters occurring in w and denoted by $|w|$ (the length of the empty word 1 equals 0).

For a morphism $h : A^* \longrightarrow B^*$ we say that a mapping $h^{-1} : B^* \longrightarrow A^*$ is a left inverse of h, if and only if $h^{-1}h(w) = w$ for all $w \in A^*$. Notice, that

1. h^{-1} exists if h is injective, that is if h is a code,

2. $h^{-1} : B^* \longrightarrow A^*$ is, in general, determined ambiguously, according to the fact that usually $h(A^*)$ is a proper subset of B^*,

3. h^{-1} is, in general, a mapping, not necessarily a morphism.

Assuming that morphisms $f : A^* \longrightarrow A^*$ and $g : A^* \longrightarrow A^*$ are codes (injections) and denoting by f^{-1}, g^{-1} some fixed inverses on A^*, it is possible to define [1] a semi-commutativity set associated to the ordered quadruple f^{-1}, g^{-1}, f, g putting

$$SCOM(f^{-1}, g^{-1}, f, g) = \{w \in A^* : f^{-1}g^{-1}fg(w) = w\}.$$

A word $w \in A^*$ is called a key-word, if there is at least one letter in A that occurs exactly once in w. This letter is called a key of w. A set $C \subset A^*$ of key-words is called a key-code, if there exists an injection (called key-injection) $key : C \longrightarrow A$ such that for any $w \in C$,

1. $key(w)$ is a key of w,

2. $key(w)$ occurs in no word of C other than w itself.

For any key-word w in a key-code C and a fixed mapping key we use the notation $w = l(a)ar(a)$ where $a = key(w)$ is the key of w and $l(a), r(a)$ denote a suitable prefix and sufix of w. Given a key-code C and a fixed key-injection key the set of all keys of words in C is denoted by $key(C)$, simply an image of C by a mapping key. Of course $\#C = \#key(C)$. We extend key to words. For $w \in A^*$ $key(w)$ is a word composed of all keys occurring in w. If $key(w) \neq 1$ we say that w is a non-trivial word. A set $B \subset A^*$ is non-trivial if it contains a non-trivial word.

A key-injection could be considered as a bijection and inverted to

$$key^{-1} : key(C) \longrightarrow C.$$

Now it is possible to extend it to a morphism, say $t : A^* \longrightarrow A^*$ putting $t(a) = 1$ for all $a \in A \setminus key(C)$. The obtained in such a way morphism t is referred to as a key-code morphism. Obviously, in general a key-code morphism has not to be a code as a matter of the fact that usually it erases some letters from A.

Assuming that all morphisms used for a semi-commutativity sets and in fact for the introduced above protocol are key-code morphisms limits a bit the research. But we generalize the study considering inverses of key-code morphisms as rational relations. Namely, let us assume that t_S, t_B are key-code morphisms used as encryption operations by S and B, respectively. Now, we consider, in the place of equation $(*)$, a weak version of this equation, namely a relation

$$(**) \ \ w \in t_B^{-1} t_S^{-1} t_B t_S(w)$$

Even with this weak version, it is still possible to point out a strong connection with cryptographic protocols, for example using interpretative morphisms as in [7].

Choosing this direction of a research we focus on a decidability problem that we will present in the sequel.

4. Generalization of the problem

Let us assume that t_S, t_B are key-code morphisms used as encryption operations by S and B, respectively. Assume that $key(C_S) = key(C_B)$. Let t_B^{-1}, t_S^{-1} denote rational relations [3] defined by inverse images.
In the place of equation

$$(*)\ \ D_B D_S E_B E_S(w) = w$$

we consider

$$(**)\ \ w \in t_B^{-1} t_S^{-1} t_B t_S(w)$$

Hence, we consider a generalized semi-commutativity set associated to the ordered quadruple $t_B^{-1}, t_S^{-1}, t_B, t_S$

$$\overline{SCOM}(t_B^{-1}, t_S^{-1}, t_B, t_S) = \{w \in A^* : w \in t_B^{-1} t_S^{-1} t_B t_S(w)\}.$$

5. Result

Within this section we assume that there are given two key-code morphisms $t_S : A^* \longrightarrow A^*$ and $t_B : A^* \longrightarrow A^*$. A rational relation $t_S^{-1} \subset A^* \times A^*$ is defined as follows. For $u, w \in A^*$

$$u\, t_S^{-1}\, w \quad \text{iff} \quad u \in t_S^{-1}(w)$$

and of course exactly the same for t_B^{-1}.

Lemma 1. *Let $w \in A^{=}A^*$ 1) be a nontrivial word, that is $key(w) \neq 1$.*
$w \in t_B^{-1} t_S^{-1} t_B t_S(w)$ iff $C_S^+ \cap C_B^+$ is not empty.

Proof. If $w \in t_B^{-1} t_S^{-1} t_B t_S(w)$ then $t_B t_S(w) = t_S t_B(w) = z \in C_S^+ \cap C_B^+$.
If $w \in C_S^+ \cap C_B^+$, then $t_B t_S(w) = t_S t_B(w)$ and thus $w \in t_B^{-1} t_S^{-1} t_B t_S(w)$. $\square$

Theorem 1. *It is decidable if a non-empty and non-trivial word w is in $t_B^{-1} t_S^{-1} t_B t_S(w)$. Hence it is decidable if $\overline{SCOM}(t_B^{-1}, t_S^{-1}, t_B, t_S) = \{w \in A^* : w \in t_B^{-1} t_S^{-1} t_B t_S(w)\}$ is empty.*

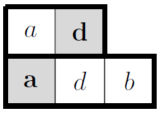
Proof. According to the Lemma 5.1 it is enough to prove that a problem of finding a non-empty word w in $C_B^* \cap C_S^*$ is decidable. We apply "dominoes technique" – see [4]. Let us visualise words of key-codes C_B, C_S as dominoes. Each domino is divided into squares and in each square there is a letter of a represented word. Thus the number of squares of a domino is exactly the same as the length of a word which it represents.

Now, having two sets of dominoes, say $[C]_B$ and $[C]_S$, which represent words of C_B and C_S respectively, we construct a set of double-dominoes according to the following procedure.

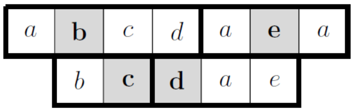
1. Fix a domino $[k_1]$ from $[C]_S$ where k_1 is its key.
2. Choose all dominoes from $[C]_B$ with keys occurring as letters in $[k_1]$.
3. Create a double domino by arranging two rows of dominoes, one on the top of the other. In the upper row, put the domino selected in step 1. from $[C]$ and in the lower one we put dominoes obtained in 2. in such a way that:
   (a) there are no gaps between them,
   (b) the letters in both rows are identical.
4. If the lower row is longer than the upper one, then consider a part of the most right domino in this row as a fixed in step 1. and repeat respectively steps 2. and 3. for $[C]_S$, putting this time dominoes in the upper row,
5. Continue this procedure for as long as possible.

As a result of applying this procedure to all dominoes from $[C]_S$ we obtain a finite set of double-dominoes.

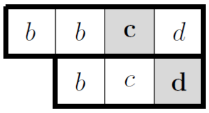
Hence we have the following possibilities:



A double domino of this type is referred to as an initial double domino.



A double domino of this type is referred to as a middle double domino.

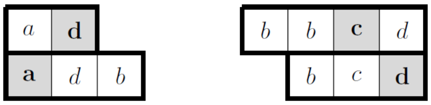


A double domino of this type is referred to as a final double domino.

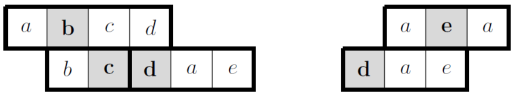
A node set consists of all double-dominoes (initial, final and middle).

It is essential that in what follows a well matched double dominoes means that for any two double dominoes it is possible to put them next to each other without gaps. It includes also the case in which an upper or lower domino from the second double domino covers (maybe partially) the first double domino. The accepted covering should be executed along full dominoes from C_S or C_B.

A well matched double dominoes:



A well matched double dominoes - covering case.



Finally we construct a tree of successful paths

1. choose an initial double domino as a root of a tree
2. choose all double dominoes well matched to the root and put an arc from the initial double domino to each of them.
3. having double dominoes – leaves of the constructed tree apply to each of them point 2 choosing new dominoes, that is in any path of the tree starting at the root there is no repeated double domino.
4. continue this procedure for as long as possible,

The obtained tree is finite. Find all paths from the root to the leafs which are final dominoes. These paths are covered by words - from C_B^+ and C_S^+ simultaneously. These words are generators of $C_B^* \cap C_S^*$.
Executing the above algorithm for all initial dominoes as roots we obtain D, a set of words - generators of $C_B^* \cap C_S^*$. Hence $C_B^* \cap C_S^* = D^*$ and the proof is finished. $\Box$

6. References

[1] J.A.Anderson, The intersection of retracts of A^*, Theoretical Computer Science, 2000, 237, pp. 312-326.

[2] J.A.Anderson, W.Forys, T.Head, Retracts and semiretracts of free monoids, AMS Meeting, San Francisco, 1991.

[3] S.Eilenberg, Automata, Languages, and Machines, Academic Press, 1974.

[4] W.Forys, K.Krawczyk, An Algorithmic Approach to the Problem of a Semiretract Base, Theoretical Computer Science, 2006, 369, pp. 314-322.

[5] G.Paun A.Salomaa, Semi-commutativity Sets a Cryptographically Grounded Topic, Bull. Math. Soc. Sci. Math. Roumaine, 1991, 35, pp. 255-270.

[6] L.Kari, G.Paun, A.Salomaa, Semi-Commutativity Sets of Morphisms over Finitely Generated Free Monoids, Bull. Math. Soc. Sci. Math. Roumaine, 1992, 36, pp. 293-307.

[7] A.Salomaa, Public-key Cryptography, Springer V. 1990.