ON COMPUTABLE AUTOMORPHISMS IN FORMAL CONCEPT ANALYSIS

A. S. Morozov

Abstract: Under study are the automorphism groups of computable formal contexts. We give a general method to transform results on the automorphisms of computable structures into results on the automorphisms of formal contexts. Using this method, we prove that the computable formal contexts and computable structures actually have the same automorphism groups and groups of computable automorphisms. We construct some examples of formal contexts and concept lattices that have nontrivial automorphisms but none of them could be hyperarithmetical in any hyperarithmetical presentation of these structures. We also show that it could be happen that two formal concepts are automorphic but they are not hyperarithmetically automorphic in any hyperarithmetical presentation.

Keywords: formal concept analysis, computable formal context, automorphism

Formal concept analysis (FCA) is an approach to the analysis and visualization of the data given by a table “object-property.” It is related to universal algebra and artificial intellect as well as to some other areas of pure and applied mathematics. We can find the basic ideas and results on FCA in [1].

In [2, 3], we started the study of the effective (computable) content of FCA trying to answer questions like “How much effectivity do we need to define the basic objects that occur in FCA?” or “How close are they to computable structures?”, “Which problems of FCA could be solved by algorithms?”, etc. The present paper continues this study and is devoted to the computable aspects of symmetries (automorphisms) of formal contexts. Since in FCA we use the binary predicate “$x$ satisfies the property $y$,” and most of the properties of computable structures could be interpreted in a single computable binary predicate, we prove that computable formal contexts have the same automorphisms and the same difficult problems on them as do computable structures. Actually, we give a general method to transform the results on the automorphisms of computable structures into results on the automorphisms of formal contexts. It is important to note that here we do not deal with the space or time complexities of algorithms for FCA. Our main subject here is the study of the existence or nonexistence of algorithms and computable presentations in the classical or generalized sense.

We assume that the reader who wants to understand all details is familiar with the basics of FCA [1] and some ideas of generalized computability [4]. When working with hyperarithmetic in the proof of Theorem 2, we find it convenient to use the admissible set $\text{HYP}_\omega$, although these results could be obtained without it but most probably in a more complicated way. The research can find the basic facts of the theory of admissible sets and, in particular, the basic properties of $\text{HYP}_\omega$ in [5].

Recall the basic definitions and introduce some notation.

Any triple $\mathcal{F} = (G, M, \models)$, where $\models \subseteq G \times M$, is called a formal context. The sets $G$ (Gegenstände) and $M$ (Merkmale) are meant to be the set of objects and the set of their properties correspondingly; the relation $g \models m$ means that $g$ has property $m$. The relation $\models$ is also called incidence relation. The symbol $\models$ will be also used to denote the usual satisfiability relation between structures, formulas, and tuples of elements; we hope that this will lead to no confusion.

Each formal context $(G, M, \models)$ could be also presented as a classical model-theoretic structure. Here we use a standard trick. Namely, without loss of generality we can assume that $G \cap M = \emptyset$, and view

---

The author was supported by the Russian Foundation for Basic Research (Grant 09–01–12140–ofi–m) and the International DFG–RFBR grant (05–01–04003-NNIOa, COMO, 436 RUS 113/829/0–1).

this formal context as a structure \( \langle G \cup M; W, \models \rangle \), where \( W \) is an extra unary predicate distinguishing \( G \).

With each family of objects \( A \subseteq G \), we can associate its *theory* \( \text{Th}(A) \) defined as

\[
\text{Th}(A) = \{ y \in M \mid \forall x \in A \,(x \models y) \},
\]

and with each family of properties \( B \subseteq M \) we can associate the class of its models

\[
\text{Mod}(B) = \{ x \in G \mid \forall y \in B \,(x \models y) \}.
\]

If \( A \subseteq G \) and \( B \subseteq M \) then we say that the pair \( \langle A, B \rangle \) is a formal concept if \( A = \text{Mod}(B) \) and \( B = \text{Th}(A) \).

If \( \alpha = \langle A, B \rangle \) is a formal concept then \( A \) is called the *extent* of \( \alpha \) and \( B \) is called the *intent* of \( \alpha \) while we use the following notation: \( A = \text{ext}(\alpha) \) and \( B = \text{int}(\alpha) \). Note that each formal concept is uniquely defined by its extend or intent. The set of all formal concepts is partially ordered by the following relation:

\[
\langle X_0, Y_0 \rangle \leq \langle X_1, Y_1 \rangle \iff X_0 \subseteq X_1
\]

which could be easily shown to be equivalent to \( Y_0 \supseteq Y_1 \).

It is known that, for each formal context \( \mathfrak{F} \), the family of all its formal concepts forms a complete lattice, which is called the concept lattice of \( \mathfrak{F} \) and denoted by \( \mathcal{L}(\mathfrak{F}) \), and that each complete lattice \( \langle L; \leq \rangle \) is isomorphic to the concept lattice of the formal context \( \mathfrak{F}_L = \langle L, L, \leq \rangle \). Moreover, all formal concepts in \( \mathcal{L}(\mathfrak{F}_L) \) have the form \( \langle \hat{a}, \hat{a} \rangle \), for appropriate \( a \in L \), where \( \hat{a} = \{ x \in L \mid x \leq a \} \) and \( \hat{a} = \{ x \in L \mid a \leq x \} \).

These statements are known as the Basic Theorem of FCA (see [1]).

We denote the set of all natural numbers by \( \omega \).

A triple \( \mathfrak{F} = \langle G, M, \models \rangle \) is called a computable (arithmetical or hyperarithmetical) formal context if the sets \( G \) and \( M \) are computable (arithmetical or hyperarithmetical) subsets of \( \omega \) and the relation \( \models \subseteq G \times M \) is computable (arithmetical or hyperarithmetical).

A formal concept \( \alpha = \langle A, B \rangle \) of \( \mathfrak{F} \) is called computable (arithmetical or hyperarithmetical) provided that both sets \( A \) and \( B \) (i.e., its intent and extent) are computable (arithmetical or hyperarithmetical).

The reader can find some discussions on these definitions in [2, 3].

Given an arbitrary pair \( p = \langle a, b \rangle \), we denote its coordinates by \( (p)_0 = a \) and \( (p)_1 = b \).

Let \( \mathfrak{F} = \langle G, M, \models \rangle \) be a formal context. The pair of bijections \( f = \langle f_G, f_M \rangle \), \( f_G : G \to G \), \( f_M : M \to M \), is called an automorphism of \( \mathfrak{F} \) if for all \( g \in G \) and \( m \in M \) we have

\[
g \models m \iff f_G(g) \models f_M(m)
\]

(see [1]). If \( \mathfrak{F} \) is computable (arithmetical or hyperarithmetical), we say that the automorphism \( f \) is computable (arithmetical or hyperarithmetical) provided that both \( f_0 \) and \( f_1 \) are computable (arithmetical or hyperarithmetical). The Turing degree of an automorphism \( f \) is defined as the Turing degree of the mapping \( f_0 \oplus f_1 \).

Each automorphism \( f = \langle f_G, f_M \rangle \) of a formal context \( \mathfrak{F} = \langle G, M, \models \rangle \) naturally induces an automorphism \( \bar{f} \) of the lattice \( \mathcal{L}(\mathfrak{F}) \) as follows:

\[
\bar{f}(\alpha) = \langle f_G(\text{ext}(\alpha)), f_M(\text{int}(\alpha)) \rangle.
\]

Given a structure \( \mathfrak{M} \) and a formal context \( \mathfrak{F} \), we denote their automorphism groups by \( \text{Aut} \mathfrak{M} \) and \( \text{Aut} \mathfrak{F} \) respectively.

Recall the notion of a computable structure. First, define effective signatures. A signature

\[
\sigma = \langle F^{m_0}_0, F^{m_1}_1, \ldots; P^{n_0}_0, P^{n_1}_1, \ldots; c_0, c_1, \ldots \rangle
\]

is called effective if there is an effective procedure that given a natural number \( k \) computes the number of arguments \( m_k \) for the operation symbol \( F^{m_k}_k \) and the number of arguments \( n_k \) for the operation symbol \( P^{n_k}_k \). Sometimes we omit the upper indices \( m_i \) and \( n_i \) showing the numbers of arguments of the related symbols. A computable structure \( \mathfrak{M} \) is a structure of effective signature whose basic set is