Automorphisms of Types, Cayley Graphs and Representation of Finite Groups *

Sergei Soloviev

IRIT, University of Toulouse-3,
118, route de Narbonne, 31062, Toulouse, France,
soloviev@irit.fr

Isomorphisms of types are represented by λ-terms $t : A \to B$ such that there exists $t^{-1} : B \to A$ and $t^{-1} \circ t \equiv id_A, t \circ t^{-1} \equiv id_B$. (For detailed definitions and history see, *e.g.*, [2].)

An isomorphism is called an automorphism if $A = B$. For example, for $A = a \to (a \to a)$ there are two automorphisms: the identity, and the permutation of premises. Obviously, the automorphisms $t : A \to A$ form a group w.r.t. composition.

The automorphisms and automorphism groups of types were not studied in [2], but the methods developed there permit to obtain a straightforward proof of the following theorem:

Theorem 1 A finite group G is isomorphic to the group of automorphisms of some type A of simply typed λ-calculus with surjective pairing and terminal object iff it is isomorphic to the group of automorphisms of a finite rooted tree.

It turns out that using either the types of system F (see, e.g., [2]) or dependent product kinds (of, say, logical framework LF, see [4] or [6]) it is possible to represent *any* finite group. Let G be a finite group with the set of generators $S \subseteq G$. Let $C_S(G)$ denote its Cayley colored graph [8].

Proposition. (See [8], theorem 4-8.) The subgroup of color-preserving automorphisms of $Aut(C_S(G))$ is isomorphic to G.

Theorem 2. It is possible to construct a type $T_S(G)$ of system F (a kind $K_S(G)$ of LF) such that:

(i)The group of automorphisms of the type $T_S(G)$ is isomorphic to the group of color-preserving automorphisms of $C_S(G)$ and thus isomorphic to G.

(ii) The group of automorphisms of the type $K_S(G)$ is isomorphic to the group of color-preserving automorphisms of $C_S(G)$ and thus isomorphic to G.

The proof uses an explicit construction of $T_S(G)$ and $K_S(G)$. An important difference with the simply typed case is exploited: the fact that the equality of types in system F (or LF) is non-trivial, for example, it includes the α-conversion.

Example. The following types of system F are α-equal:

$$\forall X.\forall Y.\forall Z.((X \to Y) \to (Y \to Z) \to (Z \to X) \to \forall U.U)$$

$$\forall Z.\forall X.\forall Y.((Z \to X) \to (X \to Y) \to (Y \to Z) \to \forall U.U)$$

$$\forall Y.\forall Z.\forall X.((Y \to Z) \to (Z \to X) \to (X \to Y) \to \forall U.U).$$

The non-trivial automorphisms are obtained by *simultaneous* application of the same permutation to the quantifiers and the premises of implication. The group of automorphisms is the cyclic group C_3. (In general the construction is more complex and uses an encoding of colours of the Cayley graph.)

Applications. The types isomorphic to a given type and their isomorphisms form a groupoid (in cases of simple types, types of system F and kinds of LF it is finite). The

*This work was partially supported by the Climt project, ANR-11-BS02-016-02.

study of automorphism groups of types is part of the study of this groupoid, and may be of interest for foundational research, for example, the Homotopy Type Theory [1]. Another, more practical application is suggested by the observation that since automorphisms do not change data types, but change their elements, they may be of use in cryptography.

Future research. It would be of interest to study automorphisms in other type systems, for example the simply typed calculi with empty and sum types [3]. Consideration of retractions (cf. [7]) may be interesting if one wants to go beyond groups and groupoids as algebraic structures to be represented.

Acknowledgements. A paper that contains the proofs of these results is actually submitted to MSCS [5]. I would like to thank Ralph Matthes for useful remarks concerning the early version of this paper, and anonymous referees for remarks concerning this abstract.

References

[1] *Homotopy Type Theory: Univalent Foundations of Mathematics*. IAS, Princeton, 2013.

[2] R. Di Cosmo. *Isomorphisms of types: from lambda-calculus to information retrieval and language design*. Birkhauser, 1995.

[3] M. Fiore. Isomorphisms of generic recursive polynomial types. In *POPL '04*, pages 77–88, New York, USA, 2004. ACM.

[4] Z. Luo. *Computation and Reasoning: A Type Theory for Computer Science*. International Series of Monographs on Computer Science. Oxford University Press, 1994.

[5] S. Soloviev. Automorphisms of types in certain type theories and representation of finite groups. *Mathematical Structures in Computer Science*, submitted, December 2015.

[6] S. Soloviev. On isomorphism of dependent products in a typed logical framework. In *TYPES 2014*, LIPICS, pages 275–288, Schloss Dagstuhl, 2015. Leibniz-Zentrum für Informatik.

[7] C. Stirling. Proof systems for retracts in simply typed lambda calculus. In *Proceedings of ICALP*, volume 2, pages 398–409, 2013.

[8] A. T. White. *Graphs, Groups and Surfaces*. North-Holland, 1984.