Solving a Class of Higher-Order Equations over a Group Structure

Ştefan ANDREI and Wei-Ngan CHIN
National University of Singapore, CS Programme,
Singapore-MIT Alliance, 3 Science Drive 2, Singapore, 117543.
E-mails: {andrei, chinwn}@comp.nus.edu.sg

Abstract

In recent years, symbolic and constraint-solving techniques have been making major advances and are continually being deployed in new business and engineering applications. A major push behind this trend has been the development and deployment of sophisticated methods that are able to comprehend and evaluate important sub-classes of symbolic problems (such as those in polynomial, linear inequality and finite domains). However, relatively little has been explored in higher-order domains, such as equations with unknown functions.

This paper proposes a new symbolic method for solving a class of higher-order equations with an unknown function over the complex domain. Our method exploits the closure property of group structure (for functions) in order to allow an equivalent system of equations to be expressed and solved in the first-order setting.

Our work is an initial step towards the relatively unexplored realm of higher-order constraint-solving, in general; and higher-order equational solving, in particular. We shall provide some theoretical background for the proposed method, and also prototype an implementation under Mathematica. We hope that our foray will help open up more sophisticated applications, as well as encourage work towards new methods for solving higher-order constraints.

Keywords: higher-order equation, symbolic computation, *Mathematica*

1 Introduction

Looking in [Wolfram(1999)], the *(higher order) functional equations* are those in which a function is sought which is to satisfy certain relations among its values at all points. For example, we may look for functions satisfying $f(x*y) = f(x) + f(y)$ and enquire whether the logarithm function $f(x) = log(x)$ is the only solution (it's not!). A special case involves difference equations, that is, equations comparing $f(x) - f(x-1)$, for example, with some expression involving x and $f(x)$. In some ways these are discrete analogues of differential equations.

In [Kuczma et al(1990)], a functional equation is defined as *an equation of the form $f(x, y, ...) = 0$, where f contains a finite number of independent variables, known functions, and unknown functions which are to be solved for.* Many properties of functions can be determined by studying the types of functional equations they satisfy. For example, the gamma function $\Gamma(z)$ satisfies the functional equations $\Gamma(1+z) = z\Gamma(z)$ and $\Gamma(1-z) = -z\Gamma(-z)$. When the focus of a functional equation is on continuity of functions and a domain is specified, this becomes a question of topology. Functions of one variable which satisfy a difference equation will tend to follow patterns set by ordinary differential equations; naturally functions of two or more variables behave more like solutions of partial differential equations.

We refer now to the notion known as *constraint solving*. Difficult problems can offer too many choices, many of which are incompatible, few of which are optimal. *Constraints* arise in design and configuration, planning and scheduling, diagnosis and testing, and in many other contexts. Constraint programming can solve problems in telecommunications, internet commerce, electronics, bioinformatics, transportation, network management, supply chain management, and many other fields. Functional equations and constraints are the object of many recent research.

An example of research effort is *Distributed Constraint Solving for Functional Logic Programming* ([Marin et al(2001)]). They realized a distributed software system consisting of a functional logic language interpreter on one machine and a number of constraint solving engines running on other machines. The interpreter is based on an existing (sequential) implementation of a functional logic language on the computer system Mathematica ([Wolfram(1999)]) extended in two directions: the possibility to specify (non-linear) constraints over real numbers and, secondly, the possibility to specify OR-parallelism among different clauses of a predicate.

In [Flajolet et al(2001)], the authors dealt with classes of generating functions implicitly defined by differential relations, globally referred to as *functional equations*. Functional equations arise in well defined combina-

torial contexts and they can lead systematically to well-defined classes of functions. The Algolib software (`http://algo.inria.fr/libraries/software.html`) allows to specify, generate, enumerate combinatorial structures, manipulate the associated generating functions, functional equations or recurrences and studying their asymptotic behaviour.

Our paper deals with relatively unexplored higher-order (functional) equations. We use mainly the composition operation, which means the replacing of variables to other functions. This represents one of the characteristics of symbolic computation. For instance, the paper [Hong(1998)] refers to the the operation of replacing the variables with polynomials. In fact, we solve the constraints by the following form, i.e. determine the function $f : \mathbf{C} \to \mathbf{C}$ from the functional equation:

$h_1(x) \cdot (f \circ f_0)(x) + h_2(x) \cdot (f \circ f_1)(x) + ... + h_n(x) \cdot (f \circ f_{n-1})(x) = g(x)$, where $n \geq 2$, $h_i : \mathbf{C} \to \mathbf{C}$, $\forall\ i = \overline{1,n}$, $g : \mathbf{C} \to \mathbf{C}$ with the additional hypothesis that $f_i : \mathbf{C} \to \mathbf{C}$, $\forall\ i = \overline{1,n}$ are functions which form a group structure.

Generally speaking, the previous functional equation can be viewed as a linear (higher-order) equation (in f) of the form $\sum_{i=1}^{n} E_i(f,x) = g(x)$, where $E_i(f,x) = h_i(x) \cdot (f \circ f_{i-1})(x)$, $\forall\ i = \overline{1,n}$.

We assume the reader is familiar with the notions of algebraic structures, but for the sake of the presentation, we introduce some notations and definitions. We denote by $\mathbf{C}$, $\mathbf{R}$, $\mathbf{Q}$ and $\mathbf{Z}$, the set of all complex, real, rational and integer numbers. $A - B$ denotes the set $\{x \in A \mid x \notin B\}$.

The pair $(G, \circ)$ is called a *group structure* iff the following hold: (*closure*) $\forall\ x,\ y \in G,\ x \circ y \in G$; (*associativity*) $\forall\ x,\ y,\ z \in G$, $(x \circ y) \circ z = x \circ (y \circ z)$; (*identity property*) there exists $e \in G$ (called *identity element*) so that $x \circ e = e \circ x = x$, $\forall\ x \in G$; (*inverse property*) $\forall\ x \in G$, there exists an $x' \in G$ (the inverse of x) so that $x \circ x' = x' \circ x = e$. An algebraic structure (for instance, a group) $(G, \circ)$ is called *commutative* (or *Abelian*) iff $\forall\ x, y \in G$, $x \circ y = y \circ x$. A group $(G, \circ)$ is called *finite* iff it has a finite number of elements, which is also called the *order* of G. In contrast, the group $(G, \circ)$ is *infinite* iff it has an infinite number of elements. For finite groups, the so called *composition table* can be attached. In fact, denoting by $G = \{x_1, x_2, ..., x_n\}$, this table is a square matrix $A_n = (a_{ij})_{1 \leq i,j \leq n}$, such that $\forall\ 1 \leq i, j \leq n,\ a_{ij} = x_k$ iff $x_i \circ x_j = x_k$

Let $f : A \to B$ be an arbitrary function. Then f is *injective* iff $\forall\ x_1 \neq x_2$ we get $f(x_1) \neq f(x_2)$; f is *surjective* iff $\forall\ y \in B$, $\exists\ x \in A$ so that $f(x) = y$; f is *bijective* (*one-to-one correspondence*) iff f is injective and surjective.

Two groups $(G_1, \circ)$ and $(G_2, *)$ are said to be *isomorphic* (denoted by $G_1 \simeq G_2$) iff there exists a one-to-one correspondence $f : G_1 \to G_2$ such that $f(x \circ y) = f(x) * f(y)$, $\forall\ x,\ y \in G_1$. A one-to-one correspondence between the set $\{1, 2, ..., n\}$ and itself is called a *permutation* of order n, denoted by σ. Using the representation of a finite group G with composition table, it is easy to remark that G is group iff on each raw and each column of the matrix A_n, there exists exactly one occurrence of an element from G. In other words, the condition means that every raw (column) is a permutation of any other raw (column) ([Aschbacher(2000), Suzuki(1986)]).

A basic example of a finite group is the *symmetric group* denoted by Σ_n, which is the *group of permutations* (or "under permutation") of n objects. One very common type of group is the *group of integers modulo n*, denoted by $(\mathbf{Z}_n, \oplus)$ and is defined for every integer $n > 1$. Considering the notation $\hat{q} = \{m' \mid \exists\ p \in \mathbf{Z}, m' = n \cdot p + q\}$, then the elements of $\mathbf{Z}_n$ are usually denoted by $\hat{0}$, $\hat{1}$, ..., $\widehat{n-1}$, and $x \oplus y = (x + y)\ mod\ n$.

Section 2 presents our method which exploits the closure property of group structure (for functions) in order to allow an equivalent system of equations to be expressed and solved in the first-order setting. The main result is Theorem 2.1, which actually solves the previous higher-order equation. Section 3 shows a basis for building a library of groups (of functions) to support our method, of which an important sub-class is the so-called *homographic functions*. Section 4 illustrates the pragmatics of our method with an implementation under *Mathematica* ([Wolfram(1999)]). Some general, and a particular example, higher-order equation is solved using *Mathematica*. Finally, some conclusions are pointed out.

2 Our proposed method

In this section, we present the main result of the paper, i.e. the solution of the functional equation proposed in the first section. After that, some particular cases are pointed out.

Theorem 2.1 *Let $G = (\{f_0, f_1, ..., f_{n-1}\})$ be a group, where $f_0 = 1_{\mathbf{C}}$, $f_i : \mathbf{C} \to \mathbf{C}$, $i = \overline{1,n}$, $n \geq 2$ and "$\circ$" denotes function composition. Consider the functions $h_i : \mathbf{C} \to \mathbf{C}$, $i = \overline{1,n}$ and $g : \mathbf{C} \to \mathbf{C}$. Then the following equation:*

$$h_1 \cdot (f \circ f_0) + h_2 \cdot (f \circ f_1) + ... + h_n \cdot (f \circ f_{n-1}) = g$$

can be solved.

Proof The above functional equation is a short-hand for (0):

$h_1(x) \cdot (f \circ f_0)(x) + ... + h_n(x) \cdot (f \circ f_{n-1})(x) = g(x)$, where $x \in \mathbf{C}$. We will make the following $n-1$ substitutions in (0): $x \to f_1(x)$, $x \to f_2(x)$, ..., $x \to f_{n-1}(x)$.

The obtained functional equations will be denoted by (1), (2), ..., $(n-1)$.

Let us discuss in detail the substitution $x \to f_1(x)$ (the others being quite similar). The obtained functional equation will be:

(1′) $h_1(f_1(x)) \cdot (f \circ f_0 \circ f_1)(x) + h_2(f_1(x)) \cdot (f \circ f_1 \circ$
$\circ f_1)(x) + ... + h_n(f_1(x)) \cdot (f \circ f_{n-1} \circ f_1)(x) = g(f_1(x))$

Because G is a finite group, it follows that the elements $f_0 \circ f_1$, $f_1 \circ f_1$, ..., $f_{n-1} \circ f_1$ are different from each other and are a permutation of the initial group. So, we will denote by $\sigma_1 : G \to G$ the permutation such that $\sigma_1(i) = j$ if and only if $f_i \circ f_1 = f_j$.

Therefore, the previous functional equation can be rewritten in the following equivalent way:

(1) $h_1(f_1(x)) \cdot (f \circ f_{\sigma_1(0)})(x) + h_2(f_1(x)) \cdot (f \circ f_{\sigma_1(1)})(x) + \\ + ... + h_n(f_1(x)) \cdot (f \circ f_{\sigma_1(n-1)})(x) = g(f_1(x))$

In the same manner, we denote by σ_2, ..., σ_{n-1} the corresponding permutations for the equations $(2')$, ..., $(n-1')$. The obtained functional equations are:

(2) $h_1(f_2(x)) \cdot (f \circ f_{\sigma_2(0)})(x) + h_2(f_2(x)) \cdot (f \circ f_{\sigma_2(1)})(x) + \\ + ... + h_n(f_2(x)) \cdot (f \circ f_{\sigma_2(n-1)})(x) = g(f_2(x))$

. . .

$(n-1)$ $h_1(f_{n-1}(x)) \cdot (f \circ f_{\sigma_{n-1}(0)})(x) + h_2(f_{n-1}(x)) \cdot \\ \cdot (f \circ f_{\sigma_{n-1}(1)})(x) + ... + h_n(f_{n-1}(x)) \cdot (f \circ f_{\sigma_{n-1}(n-1)})(x) = \\ = g(f_{n-1}(x))$

Because σ_1, σ_2, ..., σ_{n-1} are permutations over the set $\{0, 1, ..., n-1\}$, we can interchange the order of the terms in the above sums. In this way, we get an $n \times n$ functional equations system in the variables $f \circ f_0$, $f \circ f_1$, ..., $f \circ f_{n-1}$. However this is a system with n variables, we actually are interested only in $f \circ f_0$ which equals f. We remind that h_i, $i = \overline{1,n}$ and g are known functions.

Using the fact that σ is permutation, we reorder the obtained system to have identical elements in the columns. So, we get the equivalent system (S):

$$(S): \begin{cases} h_1(x) \cdot (f \circ f_0)(x) + h_2(x) \cdot (f \circ f_1)(x) + ... \\ \quad + h_n(x) \cdot (f \circ f_{n-1})(x) = g(x) \\ h_{\sigma_1^{-1}(0)+1}(f_1(x)) \cdot (f \circ f_0)(x) + ... \\ \quad + h_{\sigma_1^{-1}(n-1)+1}(f_1(x)) \cdot (f \circ f_{n-1})(x) = g(f_1(x)) \\ . \ . \ . \\ h_{\sigma_{n-1}^{-1}(0)+1}(f_{n-1}(x)) \cdot (f \circ f_0)(x) + ... \\ \quad + h_{\sigma_{n-1}^{-1}(n-1)+1}(f_{n-1}(x)) \cdot (f \circ f_{n-1})(x) = \\ \quad = g(f_{n-1}(x)) \end{cases}$$

Before applying the Cramer's Theorem ([Shilov(1971), Poole(2003)]), we consider the determinant Δ_x of the previous system (S):

$$\begin{vmatrix} h_1(x) & . \ . & h_n(x) \\ h_{\sigma_1^{-1}(0)+1}(f_1(x)) & . \ . & h_{\sigma_1^{-1}(n-1)+1}(f_1(x)) \\ . & . \ . & . \\ h_{\sigma_{n-1}^{-1}(0)+1}(f_{n-1}(x)) & . \ . & h_{\sigma_{n-1}^{-1}(n-1)+1}(f_{n-1}(x)) \end{vmatrix}$$

If $\Delta_x \neq 0$ (0 means the null function, i.e. $f(x) = 0$, $\forall\ x \in \mathbf{C}$), then the system (S) is compatible and has unique solution. According to Cramer's Rule, we obtain $(f \circ f_0)(x) = f(x) = \frac{\Delta f_0}{\Delta_x}$, where Δf_0 is obtained from Δ by replacing the first column with the column of right hand side of the system (S).

On the contrary, if Δ_x equals null function, then according to Kronecker-Capelli's Theorem (known also as Rank Theorem, [Shilov(1971), Poole(2003)]), a nonzero principal determinant of rank p $(p < n)$ (denoted by Δ_p) will be found (otherwise, the system is trivial!). If every characteristic determinant (formed from Δ_p by adding the right hand side of (S)) is zero, then (S) is compatible and has p principal variables, the others $n-p$ variables being secondary and putted in the right-hand side of (S). In this case, the set of solutions for f is infinite (i.e. it will depend on $n-p$ independent parameters). Otherwise (if there exists at least one nonzero characteristic determinant), then (S) is incompatible, so there is no solution for this system.

Thus, the proposed functional equation has been solved.

■

Example 2.1 *Here are some examples of important groups for functional equations:*

a) $G_1 = (\{f_0, f_1\}, \circ)$, *where* $f_i : \mathbf{C} \to \mathbf{C}$, $\forall\ i = \overline{0,1}$, $f_0(x) = x$, $f_1(x) = -x$.

b) $G_2 = (\{f_0, f_1\}, \circ)$, *where* $f_0 : \mathbf{C} \to \mathbf{C}$, $f_0(x) = x$, $f_1 : (\mathbf{C} - \{0\}) \to \mathbf{C}$, $f_1(x) = \frac{1}{x}$.

c) $G_3 = (\{f_0, f_1, f_2\}, \circ)$, *where* $f_0 : \mathbf{C} \to \mathbf{C}$, $f_0(x) = x$, $f_1 : (\mathbf{C} - \{0\}) \to \mathbf{C}$ $f_1(x) = \frac{x-1}{x}$, $f_2 : (\mathbf{C} - \{1\}) \to \mathbf{C}$, $f_2(x) = \frac{1}{1-x}$.

d) $G_4 = (\{f_0, f_1, f_2\}, \circ)$, *where* $f_0 : \mathbf{C} \to \mathbf{C}$, $f_0(x) = x$, $f_1 : (\mathbf{C} - \{-\frac{\sqrt{3}}{3}\}) \to \mathbf{C}$, $f_1(x) = \frac{x+\sqrt{3}}{1-x\sqrt{3}}$, $f_2 : (\mathbf{C} - \{\frac{\sqrt{3}}{3}\}) \to \mathbf{C}$, $f_2(x) = \frac{x-\sqrt{3}}{1+x\sqrt{3}}$.

e) $G_5 = (\{f_0, f_1, f_2, f_3\}, \circ)$, *where* $f_0(x) = x$, $f_1(x) = \frac{1}{x}$, $f_2(x) = -x$, $f_3(x) = -\frac{1}{x}$. *The functions* f_i *are defined as* $f_0, f_2 : \mathbf{C} \to \mathbf{C}$ *and* $f_1, f_3 : (\mathbf{C} - \{0\}) \to \mathbf{C}$.

f) $G_6 = (\{f_0, f_1, f_2, f_3, f_4, f_5\}, \circ)$, *where* $f_0(x) = x$, $f_1(x) = \frac{1}{1-x}$, $f_2(x) = \frac{x-1}{x}$, $f_3(x) = \frac{1}{x}$, $f_4(x) = 1 - x$, $f_5(x) = \frac{x}{x-1}$. *The functions* f_i *are defined as* $f_0, f_4 : \mathbf{C} \to \mathbf{C}$, $f_1, f_5 : (\mathbf{C} - \{1\}) \to \mathbf{C}$ *and* $f_2, f_3 : (\mathbf{C} - \{0\}) \to \mathbf{C}$.

g) $G_7 = (\{f_0, f_1, f_2, ..., f_{n-1}\}, \circ)$, *where* $f_i(x) = \varepsilon^i \cdot x$, $\forall\ i = \overline{0, n-1}$, *where* $\varepsilon^n = 1$ *(*$\varepsilon = \cos\frac{2\pi}{n} + i\sin\frac{2\pi}{n}$ *being called the* **complex unity of order** n*). The functions are define as:* $f_i : \mathbf{C} \to \mathbf{C}$, $\forall\ i = \overline{0, n-1}$.

For 2 and 3 elements, there exists only one group! This is denoted by $\mathbf{Z}_2$, and $\mathbf{Z}_3$ respectively ([Aschbacher(2000), Suzuki(1986)]).

Remark 2.1 *a) The groups* G_1, ..., G_5 *are commutative. The composition table for* G_1 *and* G_2 *is* $(G_1 \simeq G_2)$*:*

$\circ$	f_0	f_1
f_0	f_0	f_1
f_1	f_1	f_0

and for G_3 *and* G_4 *is* $(G_3 \simeq G_4)$*:*

$\circ$	f_0	f_1	f_2
f_0	f_0	f_1	f_2
f_1	f_1	f_2	f_0
f_2	f_2	f_0	f_1

b) The group G_6 *is noncommutative, and has the composition table:*

$\circ$	f_0	f_1	f_2	f_3	f_4	f_5
f_0	f_0	f_1	f_2	f_3	f_4	f_5
f_1	f_1	f_0	f_3	f_2	f_5	f_4
f_2	f_2	f_4	f_5	f_1	f_3	f_0
f_3	f_3	f_5	f_4	f_0	f_2	f_1
f_4	f_4	f_2	f_1	f_5	f_0	f_3
f_5	f_5	f_3	f_0	f_4	f_1	f_2

The group G_6 is isomorphic to Σ_3 (i.e. the group of permutations of order 3).

c) G_7 is an example of a commutative group with arbitrary number of elements. Its operation can be analytically defined as $f_i \circ f_j = f_{(i+j) \bmod n}$, where $x \bmod y$ means the positive integer remainder of the division remainder of x to y. Actually, G_7 is isomorphic with the well-known integer group modulo n $(\mathbf{Z}_n, \oplus)$. Both groups satisfies the equation $x^n = 1$, $\forall$ x element in the given groups (1 denotes the identity element).

d) Depending on the requirements of the functional equation, the domain/image definitions of functions can be restricted as: for G_1 we can consider $f_i : \mathbf{Z} \to \mathbf{Z}$, $i = \overline{0,1}$; for G_2 we can consider $f_i : \mathbf{Z} - \{0\} \to \mathbf{Z}$, $i = \overline{0,1}$; for G_3, $f_i : \mathbf{Q} - \{0,1\} \to \mathbf{Q}$, $i = \overline{0,2}$; for G_4, $f_i : \mathbf{R} - \{\pm\frac{\sqrt{3}}{3}\} \to \mathbf{R}$, $i = \overline{0,2}$; for G_5, $f_i : \mathbf{Q} - \{0\} \to \mathbf{Q}$, $i = \overline{0,3}$; for G_6, $f_i : \mathbf{Q} - \{0,1\} \to \mathbf{Q}$, $i = \overline{0,5}$.

We now show a general method for construction new groups from smaller existing groups. This problem is a converse of ([Eick et al(2002)]), where the subgroups are computed from finite solvable groups.

Definition 2.1 *Given $(G_1, \circ)$ and $(G_2, \circ)$ two groups whose elements are functions, we denote:*

$G_1 \odot G_2 = \{f \mid \exists f_1 \in G_1,\ \exists\ f_2 \in G_2$ such that $f = f_1 \circ f_2$ or $f = f_2 \circ f_1\}$

Given $(G_1, \circ)$ and $(G_2, \circ)$ two groups whose elements are functions, let us consider the following condition:

(C) $\forall f_1 \in G_1 \cup G_2$, $\forall\ f_2 \in G_1 \odot G_2$ then $f_1 \circ f_2 \in G_1 \odot G_2$

Theorem 2.2 *Let $(G_1, \circ)$ and $(G_2, \circ)$ be two groups whose elements are functions. Then $(G_1 \odot G_2, \circ)$ is a group if and only if the condition (C) holds.*

Proof (**if**) Only the closure property and the inverse property for $(G_1 \odot G_2, \circ)$ need to be proved (i.e. the other properties stand from the hypothesis).

If $g \in G_1 \odot G_2$, then there exist $f_1 \in G_1$ and $f_2 \in G_2$ such that $g = f_1 \circ f_2$ (or $g = f_2 \circ f_1$). Without loss of generality, we take into consideration only the first case. Let $g' = f_2^{-1} \circ f_1^{-1}$, where f_1^{-1} and f_2^{-1} are the inverse functions in G_1, and G_2, respectively. According to Definition 2.1, it follows that $g' \in G_1 \odot G_2$. It can be easily seen, using associativity, that $g \circ g' = (f_1 \circ (f_2 \circ f_2^{-1}) \circ f_1^{-1}) = 1$ and $g' \circ g = (f_2^{-1} \circ (f_1^{-1} \circ f_1) \circ f_2) = 1$, where 1 denotes the identity element.

We come back to the closure property for $(G_1 \odot G_2, \circ)$. Let f_{12} and f_{34} be two arbitrary elements belonging to $G_1 \odot G_2$. Then there exist f_1, $f_3 \in G_1$, f_2, $f_4 \in G_2$ so that $f_{12} = f_1 \circ f_2$ (or $f_{12} = f_2 \circ f_1$) and $f_{34} = f_3 \circ f_4$ (or $f_{34} = f_4 \circ f_3$). Without loss of generality, we take into consideration only the first case. So, we get $f_{12} \circ f_{34} = f_1 \circ f_2 \circ f_3 \circ f_4$. According to Definition 2.1, $f_3 \circ f_4 \in G_1 \odot G_2$. Because (C) holds, then $f_2 \circ (f_3 \circ f_4) \in G_1 \odot G_2$, and, as a consequence, $f_1 \circ (f_2 \circ (f_3 \circ f_4)) \in G_1 \odot G_2$. So, it results that $(G_1 \odot G_2, \circ)$ is a group.

(**only if**) The proof is simply based on the inclusion $G_1 \cup G_2 \subseteq G_1 \odot G_2$ and the closure property of $(G_1 \odot G_2, \circ)$. ∎

In the following, we come with some discussion about the special case when the "target" group $(G_1 \odot G_2, \circ)$ is commutative.

Theorem 2.3 *Let $(G_1, \circ)$ and $(G_2, \circ)$ be two groups whose elements are functions. If $(G_1 \odot G_2, \circ)$ is a commutative algebraic structure, then $(G_1 \odot G_2, \circ)$ is a group.*

Proof For simplicity, we prove only the closure property for $(G_1 \odot G_2, \circ)$, the other properties can be done like in the proof of Theorem 2.2.

Let f_{12} and f_{34} be two arbitrary elements belonging to $G_1 \odot G_2$. Then there exist f_1, $f_3 \in G_1$, f_2, $f_4 \in G_2$ so that $f_{12} = f_1 \circ f_2$ (or $f_{12} = f_2 \circ f_1$) and $f_{34} = f_3 \circ f_4$ (or $f_{34} = f_4 \circ f_3$). Without loss of generality, we take into consideration only the first case. We get $f_{12} \circ f_{34} = f_1 \circ f_2 \circ f_3 \circ f_4$, and applying the commutativity and associativity, we obtain $f_{12} \circ f_{34} = (f_1 \circ f_3) \circ (f_2 \circ f_4)$. Because of the closure properties of G_1 and G_2, it follows $f_{12} \circ f_{34} \in G_1 \odot G_2$. So, $(G_1 \odot G_2, \circ)$ is a group. ∎

As a notation, we introduce $G_{n_1} \odot G_{n_2} \odot ... \odot G_{n_k}$ as $(...((G_{n_1} \odot G_{n_2}) \odot G_{n_3}) \odot ... \odot G_{n_k})$, where $k > 2$ is a nonnegative integer number.

Corollary 2.1 *Let $G_{n_i} = (\{f_0^i, f_1^i, f_2^i, ..., f_{n_i-1}^i\}, \circ)$ where $f_j^i(x) = \varepsilon_i^j \cdot x$, $\forall\ j = \overline{0, n_i - 1}$, $\varepsilon_i^{n_i} = 1$, $i = \overline{1,k}$, $k > 1$ ($\varepsilon_i = \cos\frac{2\pi}{n_i} + i \sin\frac{2\pi}{n_i}$ being called the* **complex unity of order** *n_i). Then $G_{n_1} \odot G_{n_2} \odot ... \odot G_{n_k}$ is a group.*

Proof The proof can be done by induction on $k \geq 2$. For simplicity, we shall do only the base of induction, i.e. the step $k = 2$.

We have $(f_{i_1}^1 \circ f_{i_2}^2)(x) = \varepsilon_1^{i_1} \cdot f_{i_2}^2(x) = \varepsilon_1^{i_1} \cdot \varepsilon_2^{i_2} \cdot x = \varepsilon_2^{i_2} \cdot \varepsilon_1^{i_1} \cdot x = \varepsilon_2^{i_2} \cdot f_{i_1}^1(x) = (f_{i_2}^2 \circ f_{i_1}^1)(x)$, so $f_{i_1}^1 \circ f_{i_2}^2 = f_{i_2}^2 \circ f_{i_1}^1$. According to Theorem 2.2, it follows that $G_{n_1} \odot G_{n_2}$ is a group. ∎

Remark 2.2 *According to the notations from Example 2.1 and Corollary 2.1, we have:*

1. $G_5 = G_1 \odot G_2$ and $G_6 = G_2 \odot G_3$. The pair (G_2, G_3) satisfy the Theorem 2.2 and the pair (G_1, G_2) satisfy also the condition of Theorem 2.3;

2. In general, even if two groups H_1 and H_2 are commutative, it is not necessary that $H_1 \odot H_2$ be a commutative group. As a (counter)example in this sense, the groups G_2 and G_3 are commutative, but $G_2 \odot G_3 = G_6$ isn't !

3. According to the notations from Corollary 2.1, it follows that $G_{n_1} \odot G_{n_2} \odot ... \odot G_{n_k}$ is a commutative group.

3 Case Study: Groups of Order 3

At the first part of Section 3, we refer to general functions for the groups of order 3. A necessary theorem for determining the minimum relation needed for determining if functions f_1 and f_2 are elements of a group with order 3 can be reviewed:

Theorem 3.1 *Let $f_1, f_2 : \mathbf{C} \to \mathbf{C}$ so that $f_1 \circ f_2 = 1_{\mathbf{C}}$ and $f_1 \circ f_1 = f_2$. Then $f_2 \circ f_2 = f_1$, $f_2 \circ f_1 = 1_{\mathbf{C}}$ and $G = (\{1_{\mathbf{C}}, f_1, f_2\}, \circ)$ is a group.*

Proof It can be easily proof that the composition table for $(\{1_{\mathbf{C}}, f_1, f_2\}, \circ)$ is:

$\circ$	$1_{\mathbf{C}}$	f_1	f_2
$1_{\mathbf{C}}$	$1_{\mathbf{C}}$	f_1	f_2
f_1	f_1	f_2	$1_{\mathbf{C}}$
f_2	f_2	$1_{\mathbf{C}}$	f_1

Because on each row and column, each element of the set $\{1_{\mathbf{C}}, f_1, f_2\}$ occurs exactly once, it follows that $(\{1_{\mathbf{C}}, f_1, f_2\}, \circ)$ is a group. ∎

As a side remark, the converse of Theorem 3.1 is an obvious result: *Let $G = (\{f_0, f_1, f_2\}, \circ)$ be a group where $f_i : \mathbf{C} \to \mathbf{C}$, $\forall\ i = \overline{1,3}$. Then one of the functions is identity, (e.g. $f_0 = 1_{\mathbf{C}}$), $f_1 \circ f_2 = 1_{\mathbf{C}}$ and $f_1 \circ f_1 = f_2$.*

We continue by determining all the groups with 3 elements, for which each element is a homographic function. We choose this particular subclass of functions because it is closed under composition, i.e. the composition of two homographic functions is a homographic function, too. This property is useful for the closure property of the group G (in particular, of order 3).

Definition 3.1 *A function is called* **homographic** *iff there exists complex (real) numbers a, b, c, d so that $f : \mathbf{C} - \{-\frac{d}{c}\} \to \mathbf{C}$ and $f(x) = \frac{ax+b}{cx+d}$*

Actually, we will find the general form of the homographic functions f_1 and f_2 (f_0 being *identity function*). In fact, this will be a generalisation of the groups G_2 and G_3.

The next result is restricted to the set of real numbers.

Theorem 3.2 *Let f_1 and f_2 be two distinct homographic functions such that $f_1 \circ f_2 = 1_{\mathbf{C}}$ and $f_1 \circ f_1 = f_2$. Then, we distinguish only two situations:*

a) there exists $b \in \mathbf{R} - \{0\}$ for which:

$f_1(x) = \frac{bx+b^2}{-x}$ *and* $f_2(x) = \frac{-b^2}{x+b}$ *(of course, $x \notin \{0, -b\}$);*

b) there exist $a,\ b \in \mathbf{R}$ and $c \in \mathbf{R} - \{0\}$ such that $c^2 + c + ab = -1$ (and $ab \leq -\frac{3}{4}$) for which $f_1(x) = \frac{x+a}{bx+c}$ and $f_2(x) = \frac{cx-a}{-bx+1}$.

Proof Without loss of generality, we suppose that the only possible homographic functions are:

a) $f_1(x) = \frac{x+b_1}{c_1x+d_1}$, $f_2(x) = \frac{b_2}{c_2x+d_2}$ and b) $f_1(x) = \frac{x+b_1}{c_1x+d_1}$, $f_2(x) = \frac{x+b_2}{c_2x+d_2}$

Now, we put the hypothesis conditions and we get for the two cases:

a) From the first condition, we obtain that if $c_2 = 0$, then $f_1 = f_2 = -b_1$ which is not good because f_1 and f_2 are distinct functions. If $c_2 \neq 0$ then $d_1 = 0, b_1c_2 = b_2c_1$ and $b_2 = -b_1d_2$. Checking also the second condition, we get $b_1c_1 = -1$, therefore (by denoting b_1 with b) we get the conclusion of point a).

b) From the first condition, we get $b_2 = -b_1d_2$, $c_1 = -c_2d_1$ and $b_1c_2 = -1$. Using also the second condition, we get $1 + d_1 \neq 0$ (otherwise, we get a wrong result) and $b_1c_2 = b_2c_1$. therefore $d_1d_2 = 1$. Finally, $b_2 = -\frac{b_1}{d_1}$, $c_2 = -\frac{c_1}{d_1}$, $d_2 = \frac{1}{d_1}$ and $d_1^2 + d_1 + b_1c_1 + 1 = 0$. The discriminant of the previous quadratic equation must be positive, so $\Delta = -4b_1c_1 - 3 \geq 0$ implies $b_1c_1 \leq -\frac{3}{4}$. By renaming b_1, c_1, d_1 with a, b, c we got the conclusion of point b). Of course, we have the relations $c^2 + c + ab = -1$ and $ab \leq -\frac{3}{4}$. ∎

Remark 3.1 *1. The functions determined in Theorem 3.2 are the most general ones related to homographic functions;*

2. Taking $b = -1$ at the point a), we get the group G_3;

3. Taking $a = \sqrt{3}$, $b = -\sqrt{3}$ and $c = 1$ at the point b), we get the group G_4.

4 Implementation in *Mathematica*

Mathematica ([Wolfram(1999)]) is one of the most popular fully integrated environment for technical and symbolic computing. First released in 1988, it has had a profound effect on the way computers are used in many technical and other fields. *Mathematica* software is able to solve equations (linear, polynomial, differential, etc), to do image processing, to create graphics, for financial mathematics, a.s.o.

This powerful software can be used in our paper for obtaining the final solution. Regarding the main result of this paper (Theorem 2.1), we denote by `X1`, `X2`, ..., `XN`, the variables of the obtained linear system of equations (i.e. $(f \circ f_0)(x)$, $(f \circ f_1)(x)$, ..., $(f \circ f_{n-1})(x)$) and by `b1`, `b2`, ..., `bN` the right-hand side terms $g(x)$, $g(f_1(x))$, ..., $g(f_{n-1}(x))$. Moreover, we shall consider a matrix $A = (a_{ij})$, $1 \leq i, j \leq n$ such as $a_{ij} = h_{\sigma_{i-1}^{-1}(j-1)+1}(f_i(x))$. Now, we write a *Mathematica* program in order to solve the above system of equations:

```
eq1 = a11 X1 + a12 X2 + ... + a1N xN == b1
```

```
eq2 = a21 X1 + a22 X2 + ... + a2N xN == b2
 . . .
eqN = aN1 X1 + aN2 X2 + ... + aNN xN == bN
TheSystem = [eq1, eq2, ..., eqN]
TheVariables = [X1, X2, ..., XN]
Solve[TheSystem, TheVariables]
```

Of course, in the above program, some equivalent notations were made, e.g. `a11` stands for a_{11}, `a1N` stands for a_{1n}, the dots `...` will be replaced in a concrete example. The built-in function `Solve` attempts to solve an equation or set of equations for the mentioned (unknown) variables. Equations are given in the form `lhs == rhs`, where `lhs`, `rhs` denote left hand-side and right hand-side, respectively. The space between two identifiers is interpreted as a multiplication. Simultaneous equations are combined in a list denoted `TheSystem`. The list of variables is specified in `TheVariables`. We show how *Mathematica* can be useful for solving our kind of functional equations. The *Mathematica* evaluation for `X1` will corresponds to $(f \circ f_0)(x)$, which is the required variable.

As mentioned in the first part of our paper, the proposed functional equation is higher-order because the function f is required, not the x-values. *Mathematica* cannot directly solve our kind of functional equation. However, *Mathematica* can solve higher-order equation only for differential equations (i.e. of some classical form).

Example 4.1 *Unsuccessful run of Mathematica for the functional equation:*

"Find $f : \mathbf{R} - \{0, 1\} \to \mathbf{R}$ *so that* $x^2 \cdot f(x) + 2 \cdot x \cdot f(1 - \frac{1}{x}) = \frac{1}{x} - x + 1$*."*

If we model directly with Mathematica and request for it to be solved:

```
In:=Solve[x^2*f[x]+2*x*f[1-1/x]==1/x-x+1,f]
```

we obtain `Out = {{}}`*. Alternatively, adding* x *as a second parameter in the following request:*

```
In:=Solve[x^2*f[x]+2*x*f[1-1/x]==1/x-x+1,f,x]
```

we get the message "`The equations appear to involve the variables to be solved for in an essentially non-algebraic way.`*".*

As shown above, *Mathematica* cannot be applied directly. Let us now see how our method can be applied to solve this class of higher-order equation.

Taking into consideration the group G_3 from Example 2.1, we obtain $h_1(x) \cdot (f \circ f_0)(x) + h_2(x) \cdot (f \circ f_1)(x) = g(x)$, where $h_1(x) = x^2$, $h_2(x) = 2 \cdot x$, and $g(x) = 1/x - x + 1$. According to the proof of Theorem 2.1, we can apply the substitutions $x \to f_1(x)$ and $x \to f_2(x)$, so as to get the following system of equations:

$$\begin{cases} h_1(f_1(x)) \cdot (f \circ f_1)(x) + h_2(f_1(x)) \cdot (f \circ f_2)(x) = g(f_1(x)) \\ h_1(f_2(x)) \cdot (f \circ f_2)(x) + h_2(f_2(x)) \cdot (f \circ f_0)(x) = g(f_2(x)) \end{cases}$$

Using the above notations (`X1` for $(f \circ f_0)(x)$, a.s.o.), we can have a *Mathematica* program in the following:

Example 4.2 *Successful run of Mathematica for the same functional equation is:*

```
In:=
 g[x_] = (1/x) - x + 1
 h1[x_] = x^2
 h2[x_] = 2*x
 f0[x_] = x
 f1[x_] = 1 - 1/x
 f2[x_] = 1/(1 - x)
 eq1=h1[f0[x]]*f[f0[x]]+h2[f0[x]]*f[f1[x]]==
     g[f0[x]]
 eq2 = eq1 /. x -> f1[x]
 eq3 = eq1 /. x -> f2[x]
 TheSystem=
   {Simplify[eq1],Simplify[eq2],
    Simplify[eq3]}/.
   {f[x]->X1,f[Simplify[f1[x]]]->X2,
    f[Simplify[f2[x]]]->X3}
 TheVariables = {X1, X2, X3}
 Solve[Eliminate[TheSystem,Rest[TheVariables]],
       X1]
```

whose output yields:

```
Out:={{X1->1/7(4-2/(1-x)^3+6/(1-x)^2-2/(1-x)-
          1/x^3-1/x^2+1/x-12x+4x^2)}}
```

Some explanations are necessary for the previous *Mathematica* program. In *Mathematica*, a *transformation rule* (i.e. *substitution*) of the form `x -> e` means that `x` is replaced with `e` in a purely symbolic fashion. Now, to apply a substitution (`rule`) to a particular *Mathematica* expression (`expr`), we have to type `expr /. rule`, where `/.` is called the replacement operator. For instance, in our previous example, `eq2 = eq1 /. x -> f1[x]` means the replacement of each occurrence of `x` from expression `eq1` with `f1[x]`.

Next, `Simplify[expr]` find the simplest form of `expr` by applying various standard algebraic transformations. This helps to normalize our program and we avoided the need of a composition table for the given group of functions.

`Rest[list]` returns `list` with the first element dropped, therefore the expression `Rest[TheVariables]` equals `{X2,X3}`.

The built-in function `Eliminate[eqns,vars]` eliminates variables between a set of simultaneous equations. So, the last command of the previous *Mathematica* program provides in the output only the value of `X1`, which is the required result.

With this, we conclude that the function disscussed in Examples 4.1 and 4.2 is $f(x) = \frac{1}{7} \cdot \left(4 - \frac{2}{(1-x)^3} + \frac{6}{(1-x)^2} - \frac{2}{(1-x)} - \frac{1}{x^3} - \frac{1}{x^2} + \frac{1}{x} - 12x + 4x^2\right)$

In the following, we summarize the general algorithm denoted by (HOSolve) (i.e. from Higher-Order Equational Solving) for solving our class of higher-order equation.

Algorithm HOSolve:

The Input: The equation $\sum_{i=1}^{n} E_i(f,x) = g(x)$, where $E_i(f,x) = h_i(x) \cdot (f \circ f_{i-1})(x)$, $\forall\ i = \overline{1,n}$ and a library of finite groups with functions;
The Output: The function f which satisfies the above equation.

We shall present the main steps of the proposed algorithm:

1. read a higher-order equation with unknown f
2. parse it in order to obtain h_i $(i = \overline{1,n})$, f_i $(i = \overline{0,n-1})$, and g
3. check if $G = \{f_0, ..., f_{n-1}\}$ belongs to an existing group (syntactically by matching with the library of finite functional groups)
4. build `TheSystem` and `TheVariables`
5. pass to *Mathematica* the request associated to the expression `Solve[Eliminate[TheSystem, Rest[TheVariables]], X1]`
6. display the solution to f, as $f(x) = ...$

5 Conclusions

We have presented a symbolic method for solving a class of higher-order equations with an unknown function over the complex domain (or some sub-domain). Our method (Theorem 2.1) exploits the closure property of group structure (for functions) in order to allow an equivalent system of equations to be expressed and solved in the first-order setting.

In order to support the reasoning of groups, we propose Theorem 2.2 which contains a sufficient and necessary condition for the construction of new groups out of existing (sub)groups. This condition forms a basis for building a library of groups (of functions) to support our method, of which an important sub-class is the so-called *homographic functions*. Theorem 3.2 points out all the groups with 3 elements, for which each element is a homographic function over the set of real numbers.

Our work is an initial step towards the relatively unexplored realm of higher-order constraint-solving, in general; and higher-order equational solving, in particular. We have provided some theoretical background for the proposed method, and has also illustrated the pragmatics of our method with an implementation under *Mathematica*. The power of constraint-solving and symbolic computation has been found to be extremely useful for supporting many applications ([Wolfram(1999), Marin et al(2001)]). We hope that our foray will help open up more sophisticated applications, as well as encourage work towards new methods for solving higher-order constraints.

Some possible future work in this area includes higher-order inequality and higher-order constraints of non-complex domain (such as algebraic data structures).

Acknowledgment: Special thanks to Professor Ioan Tomescu for insightful comments that lead to a much improved paper.

References

[Aschbacher(2000)] M. Aschbacher. *Finite Group Theory*. Second Edition. Cambridge University Press, New York, 2000

[Flajolet et al(2001)] P. Flajolet, and R. Sedgewick. *Analytic Combinatorics: Functional Equations, Rational and Algebraic Functions*. INRIA, Rapport de recherche, France, 2001

[Eick et al(2002)] B. Eick, and C. R. B. Wright. Computing Subgroups by Exhibition in Finite Solvable Groups. *Journal of Symbolic Computation*. Vol. 33, 2002, pp. 129 - 143

[Hong(1998)] H. Hong. Groebner Basis Under Composition I. *Journal of Symbolic Computation*. Vol. 25, 1998, pp. 643-663

[Kuczma et al(1990)] M. Kuczma, B. Choczewski and R. Ger. *Iterative Functional Equations*. Cambridge University Press, England, 1990

[Marin et al(2001)] M. Marin, T. Ida and W. Schreiner. CFLP: a Mathematica Implementation of a Distributed Constraint Solving System. *The Mathematica Journal*. Vol. 8 (2), 2001, pp. 287-300

[Poole(2003)] D. Poole. *Linear Algebra. A Modern Introduction* Brooks/Cole, USA, 2003

[Sedgewick et al(1996)] R. Sedgewick, and P. Flajolet. *An Introduction to the Analysis of Algorithms* Addison Wesley Publishing Company, New York, USA, 1996

[Shilov(1971)] G.E. Shilov. *Linear Algebra*. Prentice Hall Inc., USA, 1971

[Suzuki(1986)] M. Suzuki. *Group Theory*. Springer Verlag. Berlin, Germany, 1986

[Wolfram(1999)] S. Wolfram. *The Mathematica Book*. Forth Edition. The Wolfram Medic Inc. Champaign, The Wolfram Research Company (http://www.wolfram.com/) and Cambridge University Press, Illinois, USA, 1999