ON MULTI-VALUED HOMOMORPHISMS

Dimiter Skordev
Sofia University, Department
of Mathematics and Mechanics
1126 Sofia, Bulgaria

Usually, one calls homomorphisms some single-valued mappings having certain additional properties. However, Capelli has introduced a notion of multi-valued homomorphism between groups more than hundred years ago (cf. [1, p. 61]). In the present paper, we shall study multi-valued homomorphisms in a class of systems containing the many-sorted algebras. We shall make an attempt to demonstrate the usefulness of such homomorphisms in computation theory. We shall also note some possibilities for using them in model theory.

We shall consider many-sorted algebras with primitive operations which are possibly partial-multi-valued, and we shall allow some of the primitive operations to give results which are finite sequences of objects (for each operation, the sequences to which the operation is applicable and the sequences which are obtainable as results of the application have a fixed length and their members are objects of some fixed sorts).

The precise definition of the notion of such algebra is based on the notion of signature. A signature is an ordered triple (S, Ω, τ), where S is a set (of sort names), Ω is a set (of operation symbols), and τ is a function from Ω to $S^* \times S^*$ (where S^* is the set of all finite sequences of elements of S, including the empty sequence Λ). If $\Sigma = (S, \Omega, \tau)$ is a signature and $\omega \in \Omega$ then ω is called an operation symbol of type $\tau(\omega)$ in Σ. Instead of writing

$$\tau(\omega) = ((s_1, \ldots, s_m), (s_{m+1}, \ldots, s_{m+n}))$$

we shall write

$$\tau(\omega) = (s_1, \ldots, s_m \rightarrow s_{m+1}, \ldots, s_{m+n}).$$

A generalized many-sorted algebra $\mathfrak{A}$ (called an algebra, for short) with the signature $\Sigma = (S, \Omega, \tau)$ is a pair consisting of a family $(A_s \mid s \in S)$ of non-empty sets (the carriers of $\mathfrak{A}$) and of a family $(F_\omega \mid \omega \in \Omega)$, where

$F_\omega \subseteq (A_{s_1} \times \ldots \times A_{s_m}) \times (A_{s_{m+1}} \times \ldots \times A_{s_{m+n}})$
whenever

$\tau(\omega) = (s_1, \ldots, s_m \rightarrow s_{m+1}, \ldots, s_{m+n})$

(if $((a_1, \ldots, a_m), (a_{m+1}, \ldots, a_{m+n})) \in F_\omega$ then $(a_{m+1}, \ldots, a_{m+n})$ is considered as a value of the expression $F_\omega(a_1, \ldots, a_m)$).

Let $\mathfrak{A} = ((A_s \mid s \in S), (F_\omega \mid \omega \in \Omega))$ and $\mathfrak{B} = ((B_s \mid s \in S), (G_\omega \mid \omega \in \Omega))$ be two algebras with the signature $\Sigma = (S, \Omega, \tau)$. A <u>multi-valued homomorphism</u> from $\mathfrak{A}$ to $\mathfrak{B}$ is, by definition, a family of relations $(H_s \mid s \in S)$ where $H_s \subseteq A_s \times B_s$ for all s in S and the following condition is satisfied for all ω in Ω: whenever

$\tau(\omega) = (s_1, \ldots, s_m \rightarrow s_{m+1}, \ldots, s_{m+n})$,

$((a_1, \ldots, a_m), (a_{m+1}, \ldots, a_{m+n})) \in F_\omega$,

$(a_1, b_1) \in H_{s_1}, \ \ldots, \ (a_m, b_m) \in H_{s_m}$,

there are $b_{m+1}, \ldots, b_{m+n}$ such that

$((b_1, \ldots, b_m), (b_{m+1}, \ldots, b_{m+n})) \in G_\omega$,

$(a_{m+1}, b_{m+1}) \in H_{s_{m+1}}, \ \ldots, \ (a_{m+n}, b_{m+n}) \in H_{s_{m+n}}$.

<u>Example 1</u>. This will be an example of multi-valued homomorphism from a partial-multi-valued two-sorted algebra $\mathfrak{A}$ to a partial algebra $\mathfrak{B}$ with the same signature. Consider the signature

$\Sigma = (\{\text{real, Boolean}\}, \{\text{sum, mult, opp, inv, exp, pos}\}, \tau)$,

where

$\tau(\text{sum}) = \tau(\text{mult}) = (\text{real, real} \rightarrow \text{real})$,

$\tau(\text{opp}) = \tau(\text{inv}) = \tau(\text{exp}) = (\text{real} \rightarrow \text{real})$,

$\tau(\text{pos}) = (\text{real} \rightarrow \text{Boolean})$.

Let B_{real} be the set of the real numbers, A_{real} be the set of all pairs (a, ε), where a is a rational number and ε is a non-negative rational number. Let $A_{\text{Boolean}} = B_{\text{Boolean}} = \{\text{true, false}\}$,

$H_{\text{real}} = \{((a, \varepsilon), x) \mid |x - a| \leqq \varepsilon\}$,

$H_{\text{Boolean}} = \{(\text{true, true}), (\text{false, false})\}$.

Let

$G_{\text{sum}}(x,y) = x+y$, $G_{\text{mult}}(x,y) = x.y$, $G_{\text{opp}}(x) = -x$, $G_{\text{exp}}(x) = e^x$

for all real numbers x and y,

$G_{\text{inv}}(x) = 1/x$ for $x \neq 0$,

$G_{\text{pos}}(x) = \text{true}$ for $x > 0$, $G_{\text{pos}}(x) = \text{false}$ for $x < 0$

($G_{\text{inv}}(x)$ and $G_{\text{pos}}(x)$ are not defined for x=0). For all (a, ε) and (b, η) in A_{real}, let

$F_{\text{sum}}((a, \varepsilon), (b, \eta)) = (a+b, \varepsilon+\eta)$,

$F_{\text{mult}}((a, \varepsilon), (b, \eta)) = (a.b, |a|.\eta + |b|.\varepsilon + \varepsilon.\eta)$,

$F_{\text{opp}}((a, \varepsilon)) = (-a, \varepsilon)$,

$F_{\text{inv}}((a, \varepsilon)) = \text{if } |a| > \varepsilon \text{ then } (1/a, \varepsilon/(a^2 - \varepsilon.|a|)) \text{ else not defined}$,

$F_{\text{pos}}((a, \varepsilon)) = \text{if } a > \varepsilon \text{ then true else if } a < -\varepsilon \text{ then false else not defined.}$

For all (a,ε) in A_{real}, let the values of the expression $F_{exp}((a,\varepsilon))$ be all pairs of the form

$$\left(\sum_{k=0}^{m}\frac{a^k}{k!},\ \frac{(|a|+\varepsilon)^{m+1}}{m!(m+1-|a|-\varepsilon)}+\varepsilon\cdot\sum_{k=0}^{m-1}\frac{(|a|+\varepsilon)^k}{k!}\right),$$

where $m+1>|a|+\varepsilon$. Then the system of the two relations H_{real}, $H_{Boolean}$ is a multi-valued homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$.

The above example has some connection with the problem of error estimation at numerical calculations. In the present paper, however, we are more interested in examples where the multi-valued homomorphisms turn out to be useful for the study of programs. Let us give such an example.

Example 2. Consider the following two PASCAL programs:

```
program f (input, output);
   var n, x, y, z: integer;
   begin
      x:=1; y:=1; read(n);
      while n>1 do
         begin
            z:=x+y; x:=y; y:=z; n:=n-1
         end;
      write(y)
   end.
```

```
program g (input, output);
   var n, x, y: integer;
   begin
      x:=1; y:=1; read(n);
      while n>1 do
         begin
            if x>y then y:=x+y else x:=x+y;
            n:=n-1
         end;
      if x>y then write(x) else write(y)
   end.
```

We shall prove that these two programs are equivalent, i.e. they have one and the same input-output relation (when the input is a positive integer then the output is the corresponding Fibonacci number, but it is not our task to prove this in the example).

We first note that the input-output relation F_{result} of program f can be represented as the composition of three functions F_{start}, F_{cycle} and F_{final}, the first one acting from the set Z of the integers to the set Z^4 of the ordered quadruples of integers (each such quadruple re-

presents a valuation of the variables n, x, y, z), the second one acting from $\underline{Z}^4$ to $\underline{Z}^4$, and the third one acting from $\underline{Z}^4$ to $\underline{Z}$. Namely, F_{start} is the multi-valued function transforming each integer i into an arbitrary quadruple (i,1,1,1), where $1 \in \underline{Z}$, F_{cycle} describes the transformation of an arbitrary valuation of n, x, y, z by the operator

```
while n>1 do
   begin
      z:=x+y; x:=y; y:=z; n:=n-1
   end
```

and $F_{final}(i,j,k,l)=k$ for all integers i, j, k, l (The choice of F_{start} corresponds to the assumption that the initial value of z in the execution of the program may be an arbitrary integer. In the case of such a PASCAL compiler that this value is necessarily 0, it would be more convenient to set $F_{start}(i)=(i,1,1,0)$ for all integers i). On its part, the function F_{cycle} is representable as the while-iteration of the function F_{step} controled by the predicate F_{cond}, where

$F_{step}(i,j,k,l)=(i-1,k,j+k,j+k)$,

$F_{cond}(i,j,k,l)=$if $i>1$ then true else false

for all integers i, j, k, l.

Consider the signature $\Sigma_0=(S, \Omega_0, \tau_0)$, where

$S=\{$integer, memory, Boolean$\}$,

$\Omega_0=\{$start, step, cond, final$\}$,

τ_0(start)=(integer$\longrightarrow$memory),

τ_0(step)=(memory$\longrightarrow$memory),

τ_0(cond)=(memory$\longrightarrow$Boolean),

τ_0(final)=(memory$\longrightarrow$integer),

as well as the algebra $\mathfrak{A}_0=((A_s \mid s \in S), (F_\omega \mid \omega \in \Omega_0))$ with the signature Σ_0, where

$A_{integer}=\underline{Z}$, $A_{memory}=\underline{Z}^4$, $A_{Boolean}=\{$true, false$\}$.

Consider also the consecutive enrichments $\Sigma_\nu=(S, \Omega_\nu, \tau_\nu)$, $\nu=1,2,3$, of Σ_0 and the corresponding algebras $\mathfrak{A}_\nu=((A_s \mid s \in S), (F_\omega \mid \omega \in \Omega_\nu))$, where

$\Omega_1=\Omega_0 \cup \{$cycle$\}$, $\Omega_2=\Omega_1 \cup \{$start$\circ$cycle$\}$, $\Omega_3=\Omega_2 \cup \{$result$\}$,

τ_1(cycle)=(memory$\longrightarrow$memory), $\tau_1 \restriction \Omega_0=\tau_0$,

τ_2(start$\circ$cycle)=(integer$\longrightarrow$memory), $\tau_2 \restriction \Omega_1=\tau_1$,

τ_3(result)=(integer$\longrightarrow$integer), $\tau_3 \restriction \Omega_2=\tau_2$,

$F_{start\circ cycle}=F_{start} \circ F_{cycle}$.

A similar inspection of program g leads to the same signatures and the corresponding algebras $\mathfrak{B}_\nu=((B_s \mid s \in S), (G_\omega \mid \omega \in \Omega_\nu))$, $\nu=0,1,2,3$, where

$B_{integer}=\underline{Z}$, $B_{memory}=\underline{Z}^3$, $B_{Boolean}=\{$true, false$\}$,

$G_{start}(i)=(i,1,1)$, $G_{final}(i,j,k)=\max\{j,k\}$,

$G_{step}(i,j,k)$=if $j>k$ then $(i-1,j,j+k)$ else $(i-1,j+k,k)$,

$G_{cond}(i,j,k)$=if $i>1$ then true else false,

G_{cycle} is the while-iteration of G_{step} controled by G_{cond},

$G_{start\circ cycle}=G_{start}\circ G_{cycle}$,

$G_{result}=G_{start}\circ G_{cycle}\circ G_{final}$.

Consider now the system of binary relations $(H_s \mid s\in S)$, where

$H_{integer}=\{(i,i) \mid i\in \underline{Z}\}$,

$H_{Boolean}=\{(\text{true}, \text{true}), (\text{false}, \text{false})\}$,

$H_{memory}=\{((i, \min\{j,k\}, \max\{j,k\}, l), (i,j,k)) \mid i,j,k,l\in \underline{Z},\ j\geq 0,\ k\geq 0\}$.

Then it is easy to check that $(H_s \mid s\in S)$ is a multi-valued homomorphism from $\mathcal{A}_0$ to $\mathcal{B}_0$ and $(H_s^{-1} \mid s\in S)$ is a multi-valued homomorphism from $\mathcal{B}_0$ to $\mathcal{A}_0$. But enrichment by composition and while-iteration preserves multi-valued homomorphisms (we omit the precise formulations*/ and the proofs which are straightforward). This leads to the consecutive conclusion that $(H_s \mid s\in S)$ is a multi-valued homomorphism from $\mathcal{A}_\nu$ to $\mathcal{B}_\nu$ and $(H_s^{-1} \mid s\in S)$ is a multi-valued homomorphism from $\mathcal{B}_\nu$ to $\mathcal{A}_\nu$ ($\nu=1$, 2,3). Using this fact for $\nu=3$ and applying the definition of the notion of multi-valued homomorphism for ω=result, we reach the desired conclusion that

$F_{result}=G_{result}$.

So far, we did not consider examples where some of the operation symbols has type $(s_1,\ldots,s_m \longrightarrow s_{m+1},\ldots,s_{m+n})$ with $n\neq 1$. Without giving such examples with $n>1$, let us note that in the multi-valued case the corresponding operations cannot, in general, be expressed as Cartesian combination of multi-valued operations of type $(s_1,\ldots,s_m \longrightarrow s_{m+r})$, $r=1,\ldots,n$. The case $n=0$, although seeming to be not interesting because of being degenerate, can be useful for covering the general notion of algebraic system having not only primitive operations, but also primitive relations. Of course, a primitive relation between objects of sorts $s_1,\ldots,s_m$ can be treated as an operation of type $(s_1,\ldots,s_m \longrightarrow$ Boolean). Unfortunately, the corresponding notion of homomorphism is not in accordance with the usual notion of homomorphism for general algebraic systems. A more successful way of treatment of a primitive relation of the kind mentioned above is the following one: to consider the partial operation of type $(s_1,\ldots,s_m \longrightarrow \Lambda)$ whose domain coincides with the given relation (considered as a set of m-tuples). Using such an approach, we have a correct representation of the usual notion of homomorphism.

---

*/ Precise formulations of a number of results of such kind using a slightly different terminology can be found in [2] (let us note that the notion of homomorphism of the present paper corresponds to the notion of semi-homomorphism of [2]).

As to the generalization of the notion of isomorphism in the spirit of the proposed generalization of the notion of homomorphism, it turns out that the following definition is an appropriate one: if $\mathfrak{A} = ((A_s | s \in S), (F_\omega | \omega \in \Omega))$ and $\mathfrak{B} = ((B_s | s \in S), (G_\omega | \omega \in \Omega))$ are two algebras having one and the same signature then we call a multi-valued isomorphism from $\mathfrak{A}$ to $\mathfrak{B}$ each multi-valued homomorphism $(H_s | s \in S)$ from $\mathfrak{A}$ to $\mathfrak{B}$ such that $(H_s^{-1} | s \in S)$ is a multi-valued homomorphism from $\mathfrak{B}$ to $\mathfrak{A}$ and $pr_1 H_s = A_s$, $pr_2 H_s = B_s$ for all s in S. The well-known fact that isomorphic algebraic systems have one and the same elementary theory remains valid if we adopt the above notion of isomorphism and treat the primitive relations as indicated above. As an example of a generalized isomorphism of the considered kind, we can mention the natural many-one correspondance between the elements of an algebraic system and the elements of its factor-system with respect to a congruence relation.

Acknowledgment. Thanks are due to Prof. Yu. L. Ershov for calling the author's attention to the book [1].

References

1. O. Schmidt. Izbrannye trudy. Matematika. Izdatel'stvo AN SSSR, Moscow, 1959.

2. D. Skordev. On multivalued homomorphisms in the theory of computability. In: 7th International Congress of Logic, Methodology and Philosophy of Science (Salzburg, Austria, July 11-16, 1983), vol. 1, Abstracts of Sections 1, 2, 3, 4 and 7, pp. 152-155.