(G)Arrows: Mathematical overview

Alexander Kuklev
Göttingen University

1 Monoids and algebraic generalizations

It has been noted by many researchers [1][2][3], that algebra mostly studies monoids internal to various monoidal categories, most ubiqtuous examples being (associative unital) rings or semirings such as $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}, \mathbb{H}, \mathbb{M}_n(R)$ which are monoids in the category of abelian groups Ab and abelian monoids AbMon respectively.

A monoid A in a given category is said to be an algebra over another monoid B whenever there is an action of A on B ("scalar multiplication by elements of B") satisfying the usual axioms of action and associative with product in both A and B, i.e. $(a_1 a_2)b = a_1(a_2 b)$, $a(b_1 b_2) = (ab_1)b_2$ for all $a_* \in A$, $b_* \in B$ where juxtaposition denotes product or scalar multiplication depending on context. B is canonically embedded into A by $b \mapsto b\,\mathbf{1}_A$. Category of all algebras over monoid M internal to category $\mathcal{C}$ can be defined as an under category $M \downarrow \mathcal{C}$.

Being an algebra over M is the coherent notion of extending M [4]:
Integers $\mathbb{Z}$ are an algebra over $\mathbb{N}$ (generated by -1), reals $\mathbb{R}$ are a non-finitely generated algebra over rationals $\mathbb{Q}$, both are non-finitely generated algebras over integers $\mathbb{Z}$, quaternions $\mathbb{H}$ are an algebra over $\mathbb{R}$ generated by $\{i, j\}$, matrices $\mathbb{M}_2(R)$ (2-by-2 matrices with R-valued entries) are an algebra over R generated by $\left\{\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}, \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\right\}$.

Other monoids (internal to various categories) studied in algebra include various structures native to homotopical algebra and differential topology, but also monads (monoids internal to endofuctor categories), operads (monoids internal to $\mathsf{Psh}(\mathbb{P})$, presheaf category of finite-cardinalities gruppoid) and of course monodical categories themsevles being monoids internal to Cat.

2 (G)Arrows

In the continued tradition of imperative and semi-imperative programming languages from FORTRAN and ALGOL to C, Java and Scala, procedures are composed of smaller procedures using only two primitives: composition and sequencing. For two given functions f: `A => B` and g: `B => C`

- Composition produces $g \circ f$: `A => C`, meaning "do f, then do g applied to its result".
- Sequencing produces $f; g$: `(A, B) => (C, D)`, meaning "do f and then g (their arguments and results being independent)".

Loops can be always unwound to recursion, variables can be eliminated in a Forth-esque[1] fashion, with only composition and sequencing remaining. This two operations are correctly internalized by garrows[2] which are constitute a notable special case of monoidal categories and also can be characterized as monoids in bifunctor categories. This categories provide the most general known framework for effectful computations and correspond (via Curry-Howard) to deductive systems with possibly non-serializable reasoning including linear logics, quantum logics, non-monotone logics etc. Morphisms of gar rows are called procedures and tensor product is called sequencing denoted ";" to reassemble C-style notation.

```
class Category g => GArrow g (;) u where
  --id        :: g x x
  --comp      :: g x y -> g y z -> g x z
  ga_first    :: g x y -> g (x ; z) (y ; z)
  ga_second   :: g x y -> g (z ; x) (z ; y)
  ga_cancell  :: g (u ; x) x
  ga_cancelr  :: g (x ; u) x
  ga_uncancell:: g x     (u; x)
  ga_uncancelr:: g x     (x ; u)
  ga_assoc    :: g ((x ; y) ; z) (x ; (y ; z))
  ga_unassoc  :: g (x ; (y ; z)) ((x ; y) ; z)
```

Such categories can be considered a vast generalization of rings with composition instead of addition and sequencing instead of product. Arrows (introduced by Hughes in 2000) are precisely the garrows that form an algebra over a simply-typed lambda calculus (i.e. cartesian closed category).

[1] The concatenative languages' juxtaposition is an overloaded operator reducing to composition and sequencing depending on how the types of operands match.

[2] Megacz, A. Multi-Level Languages are Generalized Arrows

3 Center of a (G)Arrow

Each imperative programming language contains a central subgarrow (cf. center of a ring) of procedures doing nothing more than computing results from incoming arguments — the category Func of pure functions for this particular language. It's a mere category because the operation of sequencing for pure functions is trivial: it's implemented by cartesian product. The whole language as a garrow G is an algebra over it's center, so whenever the center is cartesian closed (i.e. the language provides first class function types an lambda abstractions), G is an arrow.

4 Applicatives

Depending on the nature of present side effects sequencing may be or may be not commutative.

- Non-commutative arrow categories over Func can be quite complex, but
- if sequential execution is essentially (i.e. up to conjugation with an central isomorphism) commutative, the arrow category is completely defined by an applicative functor from Func.