Something about commutativity

Anonymous

January 17, 2024

We present a general framework for doing universal algebra and the construction of free algebra in Cubical Agda. We first identify the operations of an algebra by an ADT and their arities by the cardinalities of arbitary types, and then express the equations of the algebra by expressing the left and right hand sides of the equations as expression trees. We can show that a construction satisfies an algebraic structure by showing that an evaluated expression tree would indeed satisfy the system of equations, for example, $(\mathbb{N}, 0, +)$ satisfies the unit and associativity laws of a monoid. We can then formalize the notion of homomorphisms by defining it as functions which satisfies $\sigma(h(x), h(y), h(z), ...) = h(\sigma(x, y, z, ...))$ for a homomorphism h and any operation σ. Finally, we formalize the definition of a free algebra by its universal property, namely: for a free algebra F on A and an algebra $\mathfrak{B}$, a set function $g : A \to \mathfrak{B}$, there exists a unique homomorphism $f : F(A) \to \mathfrak{B}$ such that $g = f \circ i$, where $i : A \to F(A)$ is the canonical injection morphism. We define the $\# : (A \to \mathfrak{B}) \to (F(A) \to \mathfrak{B})$ operation, which lifts a set function $A \to \mathfrak{B}$ to a homomorphism $F(A) \to \mathfrak{B}$ using the universal property.

Using the framework we present different constructions of free monoids, such as list $List$, free monoid as a HIT $FreeMon$, and as an index function $\Sigma(n : \mathbb{N}).Fin\, n \to X$ which we call $Array$. We then extend our work by constructing free commutative monoids, such as swapped-list $SList$ and free monoid quotiented by a permutation relation $QFreeMon$, for example $List$ quotiented by permutation ($PList$) and $Array$ quotiented by isomorphism on index $\Sigma(\sigma : Fin\, n \simeq Fin\, m).\, v = w \circ \sigma$ (Bag). We prove these constructions are indeed free algebras by proving their universal property directly, and we can derive canonical maps between different constructions directly using the universal property.

Using the same framework, we hope to prove the following conjectures:

Conjecture 1. *A section to the canonical map from the free monoid to the free commutative monoid on set X implies a total order on set X.*

It is well known that a linear order on set X would imply a sort function on $List\, X$. It should follow that a sort function on set X would imply a linear order on X. We can formalize the notion of a sort function as a section to the canonical map from $List$ to $SList$, which can be thought of as a function which picks a canonical representation from an unordered list, thereby sorting the list in the process.

Definition 1. *Given a section $q : SList\, X \to List\, X$ to the canonical map $List\, X \to SList\, X$, we define a relation $\leq$: if x is the head of $q([x, y])$, $x \leq y$.*

Lemma 1. *Given a section $q : SList\, X \to List\, X$ to the canonical map $List\, X \to SList\, X$, $q(\{x, y\})$ must either be $[x, y]$ or $[y, x]$.*

We first concretely define the notion of membership for $List$ and $SList$: $x \in xs$ if x equals to one of the element in xs. We do so by noting propositions form a commutative monoid under $\vee$, which allow us to define membership proof for an element x using the $\#$ operation by lifting the function $\lambda y.\, x = y$ from $X \to Prop$ to $List\, X \to Prop$ and $SList\, X \to Prop$ respectively. We then note that the canonical map $f : List\, X \to SList\, X$ preserves length and preserves membership, $x \in xs$ iff $x \in f(xs)$. Since $f(q(\{x, y\})) = \{x, y\}$ by definition, $q(\{x, y\})$ must therefore have length 2, and $x, y \in q(\{x, y\})$. Let $q(\{x, y\})$ be $[u, v]$, we perform the following case analysis.

Case $x = u$, $y = v$: We obtain a proof $q(\{x, y\}) = [x, y]$, fulfilling the first case.

Case $x = v$, $y = u$: We obtain a proof $q(\{x, y\}) = [y, x]$, fulfilling the second case.

Case $x = u$, $y = u$ and $x = v$, $y = v$: We obtain a proof $x = y$. Since $q(\{x, x\}) = [u, v]$ by assumption, and $f([u, v]) = \{x, x\}$ by definition of q, $u, v \in \{x, x\}$. By the definition of membership u, v must either be x or x. Either case we obtain the proofs that $u = x$ and $v = x$. Since $u = v = x = y$, and $q(\{x, y\}) = [u, v]$, $q(\{x, y\}) = [x, y]$, fulfilling the first case.

With this lemma we can then prove $\leq$ does indeed satisfy all axioms of total order except transitivity.

Theorem 1. *$\leq$ is reflexive.*

By Lemma 1, $q(\{x, x\})$ must either be $[x, x]$ or $[x, x]$. Either case we obtain a proof that $q(\{x, x\}) = [x, x]$. Since x is the head of $[x, x]$, $x \leq x$.

Theorem 2. *$\leq$ is antisymmetric.*

Given $x \leq y$ and $y \leq x$, we want to show $x = y$. By Lemma 1, $q(\{x, y\})$ must either be $[x, y]$ or $[y, x]$. In the case $q(\{x, y\}) = [x, y]$, since $y \leq x$, y is the head of $[x, y]$, and we obtain a proof $y = x$. In the case $q(\{x, y\}) = [y, x]$, we invert x and y in the previous proof and obtain $x = y$. Either case we obtain $x = y$.

Theorem 3. *$\leq$ is total.*

We want to show for any x and y, either $x \leq y$ or $y \leq x$. By Lemma 1, $q(\{x, y\})$ must either be $[x, y]$ or $[y, x]$, therefore either x or y is the head of $q(\{x, y\})$. We obtain either $x \leq y$ or $y \leq x$.

Definition 2. *Given a section $q : SList\, X \to List\, X$ to the canonical map $List\, X \to SList\, X$, xs is said to be sorted if xs is in the image of q.*

It is unclear how transitivity of $\leq$ can be proven. We conjecture that the assumption the tail of a sorted list must be sorted is needed to prove transitivity. It should be noted that not all sections to the canonical map $List\, X \to SList\, X$ satisfy this assumption. Consider a function $s : SList\, \mathbb{N} \to List\, \mathbb{N}$ which sorts ascendingly given an odd-lengthed *SList* and descendingly given an even-lengthed *SList*. This function would indeed be a valid section, but violate the previous assumption.

If we assume X to have decidable equality, we can construct a linear order from the total order. With more constraint it should be possible to prove that the constructed linear order would be a well order, and therefore imply X is a choice set. We can then show that by assuming every set X is decidable and has a section to the canonical map from the free monoid to the free commutative monoid, we would be able to derive the axiom of choice.

Conjecture 2. *SList and Bag are free symmetric monoidal groupoid*

Previous works already established *SList* and *Bag* are free commutative monoids when 0-truncated, and it is folklore that when 1-truncated they should be free symmetric monoidal groupoid, although this has not yet been shown internally in HoTT.

In the case of *SList*, higher path constructors are needed. To establish the coherence of *swap* we need a higher *hexagon* path constructor. However to define the *hexagon* constructor it would involve compositions of *swap* which leads to a regularity problem. To avoid this we need to split the *hexagon* equations as below:

```
hexagon- : (a b c : A) (cs : SList A)
         -> a :: b :: c :: cs = c :: b :: a :: cs
hexagon↑ : (a b c : A) (cs : SList A)
         -> Square (\i -> b :: swap a c cs i) (hexagon- a b c cs)
                   (swap b a (c :: cs)) (swap b c (a :: cs))
hexagon↓ : (a b c : A) (cs : SList A)
         -> Square (hexagon- a b c cs) (swap a c (b :: cs))
                   (\i -> a :: swap b c cs i) (\i -> c :: swap b a cs i)
```

In the case of *Bag*, we first need to show that *Array* itself is a free monoidal groupoid. We then need to show a free monoidal groupoid quotiented by a permutation relation would be a free symmetric monoidal groupoid, and show that by quotienting *Array* with an isomorphism on the index, we would indeed get a free symmetric monoidal groupoid.