Representations of a concrete category as objects in the functor category

Tibor Grün

July 15, 2020

Contents

1 Introduction

$$\mathbf{Quiv} \rightarrow^{CatClosure} \leftarrow_U \mathbf{Cats} \rightarrow^{k-Algebroid} \leftarrow_U \mathbf{k-Cats} \rightarrow^{AdditiveClosure} \leftarrow_U \mathbf{k-Cats}^{\oplus}$$

2 A short overview of the tools used

GAP, QPA / QPA2, Catreps, CAP, homalg_project

3 The categories Quiv, Cats, FinSets, k-Mat, CatReps and the Functor Category

In this section, we first want to define the category **Quiv** and how it is the prototype for the category **Cats**. We want to restrict ourselves to finite concrete categories, which brings us to the category **FinSets**. Our goal is to represent concrete categories, for this we need the source and target categories of our representations. The source category is **k-Algebroids** which we compute algorithmically from a concrete category. The target category of our category representations is **k-Mat**. The category where our category representations lie in is **CatReps** for which we show that it's a subcategory of the **Functor Category**.

3.1 The category Quiv

In order to describe the category **Quiv** of quivers, we first have to define what a category is and for this we need the definition of a quiver. Lateron we will revisit this definition as we can define quivers as the objects in the quiver category **Quiv**.

Definition 3.1.1. (Quiver)
A <u>directed graph</u> or <u>quiver</u> q consists of a class of <u>objects</u> (or <u>vertices</u>) $q_0 = \mathrm{Obj}\, q$ and a class of <u>morphisms</u> (or <u>arrows</u>) $q_1 = \mathrm{Mor}\, q$ together with two defining maps

$$s,t\colon q_1 \rightrightarrows q_0$$

s called <u>source</u> and t called <u>target</u>.

In the next definition we are giving a new characterization for q_1 by looking at all arrows between two fixed objects.

Definition 3.1.2. (Hom-set of a (locally) small quiver)

(1) Given two objects $M,N \in q_0$ we write $\mathrm{Hom}_q(M,N)$ or $q(M,N)$ for the fiber $(s,t)^{-1}(\{(M,N)\})$ of the product map $(s,t) : q_1 \longrightarrow q_0 \times q_0$ over the pair $(M,N) \in q_0 \times q_0$. This is the class of all morphisms with source $= M$ and target $= N$. We indicate this by writing $\varphi : M \longrightarrow N$ or $M \xrightarrow{\varphi} N$. Hence q_1 is the disjoint union $\dot{\bigcup}_{M,N \in q_0} \mathrm{Hom}_q(M,N) = q_1$. As usual we define $\mathrm{End}_q(M) := \mathrm{Hom}_q(M,M)$.

(2) If the class $\mathrm{Hom}_q(M,N)$ is a <u>set</u> for all pairs (M,N) then we call the quiver <u>locally small</u>. We therefore talk about <u>Hom-sets</u>. If additionally, q_0 is a set, then the quiver is called <u>small</u>.

Example 3.1.3. (Quiver with 2 objects and 3 morphisms)

$$\begin{tikzcd} 1 \arrow[r, "b"] \arrow[loop below, "a"] & 2 \arrow[loop below, "c"] \end{tikzcd}$$

The objects of this quiver q are $q_0 = \{1,2\}$, and the morphisms are $q_1 = \{a,b,c\}$ with
$s(a) = 1 = t(a)$, $s(c) = 2 = t(c)$ and $s(b) = 1, t(b) = 2$.
Thus $\mathrm{End}_q(1) = \{a\}, \mathrm{End}_q(2) = \{c\}$ and $\mathrm{Hom}_q(1,2) = \{b\}$ whereas $\mathrm{Hom}_q(2,1) = \varnothing$.

In `QPA` this quiver is encoded as `q(2)[a:1->1,b:1->2,c:2->2]` where the first `(2)` in parentheses stands for the total number of objects.

Definition 3.1.4. (Composable arrows; path in a quiver)
Since we already have the source and target maps, we say two arrows $a, b \in q_1$ are composable if $t(a) = s(b)$ or $t(b) = s(a)$. In this case we can write a sequence of composable arrows $p = a_1 a_2 \cdots a_n$ where $t(a_i) = s(a_{i+1})$ for $i = 1, \dots, n-1$. We call this sequence a path from $s(a_1)$ to $t(a_n)$ and the integer $n \in \mathbb{Z}_{\geq 0}$ the length $l(p)$ of the path p. Although it's not an arrow, we can define the source and target of a path $p = a_1 \cdots a_n$ as $s(p) := s(a_1)$ and $t(p) := t(a_n)$. A path $p = a_1 \cdots a_n$ with $s(a_1) = t(a_n)$, i.e. $s(p) = t(p)$, is called cyclic.
For an endomorphism $a \in \mathrm{End}_q(M)$ we write a^n for $aa \cdots a$ (n times). In the case of $n = 0$ an empty path whose source and target are the vertex $i \in q_0$ is called the trivial path at i and is denoted e_i. Note that the composition of paths $e_i e_i$ has length zero starting at i therefore $e_i^2 = e_i$.

Lemma 3.1.5. *Let Q be a quiver. If there is a path of length at least $|Q_0|$, then there are cyclic paths, and thus infinitely many paths.[2]*

Proof. Assume that there exists a path of length greater or equal to $|Q_0|$. Then there exists a path of length $n = |Q_0|$, say $\alpha_1 \cdots \alpha_n$. Consider the vertices $x_i = s(\alpha_i)$ for $1 \leq i \leq n$ and $x_{n+1} = t(\alpha_n)$. Then these are $n+1$ vertices, thus there has to exist $i < j$ with $x_i = x_j$. Let $\omega = \alpha_i \cdots \alpha_{j-1}$, this is a path with source and target $x_i = x_j$, thus a cyclic path. But then ω^m is a path for any natural number m. The path ω has length $j - i \geq 1$, thus ω^m has length $m(j-i)$. This shows that these paths are pairwise different. □

Example 3.1.6. (A quiver with no cycles)

$$\begin{tikzcd}
2 \arrow[r, "\psi"] \arrow[dr, "\psi\rho"] & 3 \arrow[d, "\rho"] \\
1 \arrow[u, "\varphi"] \arrow[ur, "\varphi\psi"] \arrow[r, "\varphi\psi\rho"'] & 4
\end{tikzcd}$$

The longest path $1 \to 2 \to 3 \to 4$ has length 3. If after the object 4 another arrow would go to either $1, 2, 3$ or 4 itself, we would have a cyclic path and thus infinitely many paths.

Definition 3.1.7. (Category)

A category $\mathcal{C}$ is a quiver with two further maps:

(id) The identity map $1_{()}$ mapping every object $X \in \mathcal{C}_0$ to its identity morphism 1_X:

$$\mathcal{C}_0 \xrightarrow{1} \mathcal{C}_1$$

(μ) And for any two composable morphisms φ and $\psi \in \mathcal{C}_1$, i.e. with $t(\varphi) = s(\psi)$, the composition map μ, which maps $\varphi, \psi \in \mathcal{C}_1 \times \mathcal{C}_1$ to $\mu(\varphi, \psi) \in \mathcal{C}_1$ which we also write as $\varphi\psi$.

$$\mathcal{C}_1 \times \mathcal{C}_1 \xrightarrow{\mu} \mathcal{C}_1$$

The defining properties for 1 and μ are:

(1) $s(1_M) = M = t(1_M)$, i.e.
$1_M \in \mathrm{End}_{\mathcal{C}} \, \forall M \in \mathcal{C}$.

(2) $s(\varphi\psi) = s(\varphi)$ and
$t(\varphi\psi) = t(\psi)$
for all composable morphisms $\varphi, \psi \in \mathcal{C}$.

$$\mu : \mathrm{Hom}_{\mathcal{C}}(M, L) \times \mathrm{Hom}_{\mathcal{C}}(L, N) \longrightarrow \mathrm{Hom}_{\mathcal{C}}(M, N)$$

(3) $(\varphi\psi)\rho = \varphi(\psi\rho)$ [associativity of composition]

(4) $1_{s(\varphi)}\varphi = \varphi = \varphi 1_{t(\varphi)}$ [unit property]
The identity is a left and right unit of the composition.

The concept of a functor is central in category theory. It is how the objects and morphisms of two categories relate to one another. The following

Definition 3.1.8. (Functor) In the category **Cat** which has categories as objects, functors are the morphisms between these objects. Let $\mathcal{C}, \mathcal{D} \in \mathrm{Obj}\,\mathbf{Cat}$ be categories. A functor $F : \mathcal{C} \longrightarrow \mathcal{D}$ between $\mathcal{C}$ and $\mathcal{D}$ consists of the following data:

(1) For every object $c \in \mathcal{C}_0$ there is an object $Fc \in \mathcal{D}$.

(2) For every morphism $f \in \mathcal{C}_1$, i.e. $c \xrightarrow{f} c'$ there is a morphism $Ff \in \mathcal{D}_1$ with $Fc \xrightarrow{Ff} Fc'$, i.e. $s(Ff) = F\,s(f)$ and $t(Ff) = F\,t(f)$

Functors are compatible with the identity map and the composition map:

(3) For every object $c \in \mathcal{C}_0$ we have $1_c \in \mathcal{C}_1$ and for the functor F we demand that $F\,1_c = 1_{Fc} \in \mathcal{D}_1$.

(4) For every pair of morphisms $f, g \in \mathcal{C}_1$ with $t(f) = s(g)$ we have $fg \in \mathcal{C}_1$ and we demand that $Fg\,Fg = F\,gf$.

With the definition of a category and the category of functors finished, we can come back and use them to define the category of quivers **Quiv**.

Definition 3.1.9. (The category **Quiv**)[3] Let the Kronecker category $\mathcal{K}$ be the category with two objects, 0 and 1, and two non-identity morphisms, s and t $1 \underset{t}{\overset{s}{\rightrightarrows}} 0$. Let **FinSets** be the category of finite sets with morphisms being maps between those sets. The category of quivers **Quiv** is the category of functors from $\mathcal{K}$ to **FinSets**. For a quiver $q \in \mathrm{Obj}\,\mathbf{Quiv}$ we write q_x for the image of $x \in \{0, 1\}$ under q. The images under q ofthe morphisms s and t are again denoted by s and t.

As we have seen, every category is a quiver, but in general, to become a category, a quiver is lacking identity morphisms and the composition of morphisms. To be more precise, there is a functor U from the category of categories CAT to the category of quivers Quiv, called the underlying quiver or forgetful functor.

$$\mathrm{CAT} \xrightarrow{U} \mathrm{Quiv}$$

mapping every object $M \in \mathcal{C}_0$ to the same objects in q_0, mapping every arrow $\varphi \in \mathcal{C}_1$ to an arrow $a \in q_1$, respecting source and target, but forgetting the special role of the identity morphisms and of the composition morphisms.

Example 3.1.10. (Underlying quiver)

$$\begin{tikzcd}
B \arrow[r, "\psi"] \arrow[rd, "\psi\rho" near start] \arrow[loop above, "1_B"] & C \arrow[d, "\rho"] \arrow[loop above, "1_C"] \\
A \arrow[u, "\varphi"] \arrow[ru, "\varphi\psi" near start] \arrow[r, bend left, "(\varphi\psi)\rho"] \arrow[r, bend right, "\varphi(\psi\rho)"'] \arrow[loop below, "1_A"] & D \arrow[loop below, "1_D"]
\end{tikzcd}
\xRightarrow{U}
\begin{tikzcd}
2 \arrow[r, "b"] \arrow[rd, "e" near start] \arrow[loop above, "h"] & 3 \arrow[d, "c"] \arrow[loop above, "i"] \\
1 \arrow[u, "a"] \arrow[ru, "d" near start] \arrow[r, bend left, "f"] \arrow[r, bend right, "f"'] \arrow[loop below, "g"] & 4 \arrow[loop below, "j"]
\end{tikzcd}$$

In the category on the left, associativity of composition guaranteed that $(\varphi\psi)\rho = \varphi(\psi\rho)$, so those two arrows were already the same, so they are mapped to the same arrow $f = U((\varphi\psi)\rho) = U(\varphi(\psi\rho))$ in the quiver on the right. We didn't have to draw both arrows for f, but since they are equal, there is still only one arrow in the hom-set $\mathrm{Hom}_q(1,4) = \{f, f\} = \{f\}$.
All the other identities are not preserved under the forgetful functor, e.g. d doesn't know what it has to do with a and b apart from $s(d) = s(a)$ and $t(d) = t(b)$. Especially the former identity arrows are now just endomorphisms with no defining property.
The paths $g^2 f, gf$ and fj^3 are all different, while in the category, they all simplify to $1_A 1_A(\varphi\psi)\rho = 1_A(\varphi\psi)\rho = (\varphi\psi)\rho 1_D 1_D 1_D = (\varphi\psi)\rho$ due to the unit property and associativity.

Definition 3.1.11. (Ab-category) An <u>Ab-category</u> is a category in which all homomorphism sets are abelian groups, and composition distributes over addition.
In other words, A category $\mathcal{C}$ is an <u>Ab-category</u> if for every pair of objects $M, N \in \mathcal{C}_0$, $(\mathrm{Hom}_{\mathcal{C}}(M,N), +)$ is an abelian group (with the neutral element called <u>zero morphism</u>), and for all morphisms $\gamma, \delta \in \mathrm{Hom}_{\mathcal{C}}(M,N), \alpha, \beta \in \mathrm{Hom}_{\mathcal{C}}(N,L)$

$$(\gamma + \delta)\alpha = \gamma\alpha + \delta\alpha \text{ and}$$
$$\gamma(\alpha + \beta) = \gamma\alpha + \gamma\beta.$$

Note that every hom-set has its own unique zero morphism. E.g. in $\mathrm{Mat}_{\mathbb{Q}}$ the 2-by-3 zero-matrix $0 \in \mathrm{Hom}(2,3)$ is different from the 4-by-4 zero-matrix $0 \in \mathrm{Hom}(4,4)$.

Definition 3.1.12. (Initial object, terminal object, zero object)

Example 3.1.13.

Definition 3.1.14. (Kernel of a morphism

Definition 3.1.15. (Abelian category)

Definition 3.1.16. (k-linear category)

Quiver -¿ CAT: U: forget 1, forget composition
search U^{-1}
Beispiel für Adjunktion
Path Algebra:

4 Limits and colimits

4.1 Monomorphisms and epimorphisms

4.2 Kernel and cokernel; image and coimage

4.3 Direct sum and direct product

5 Functors and natural transformations

5.1 Functors map one category to another

Example 5.1.1. (Identity Functor) bla

Example 5.1.2. (Forgetful functor) bla

Definition 5.1.3. (full functor; faithful functor)

5.2 Natural transformations are morphisms between functors

6 Adjunctions

6.1 Universal objects

6.2 Forgetting the forgetful functor: Free constructions

7 Yoneda's Lemma: Completion and cocompletion of a category

7.1 Embedding categories

Lemma 7.1.1. *(Yoneda's Lemma)*

Proof. □

8 Functors and natural transformations

8.1 Functors act on objects and morphisms of a category

8.2 Natural transformations are morphisms between functors

8.3 Representations are Functors into a matrix category

8.4 Finite concrete categories

Yonedas Einbettungs-Lemma: Fehlende Limiten bzw. Kolimiten exitieren nach der Einbettung.

Einbettung in Kategorien, die mehr Limiten haben als die Zielkategorie.

"(Ko-)Vervollständigung" der Kategorie (Completion / Cocompletion)

Quiver = unvollständige Struktur einer Kategorie Erzeugendensystem einer Kategorie.

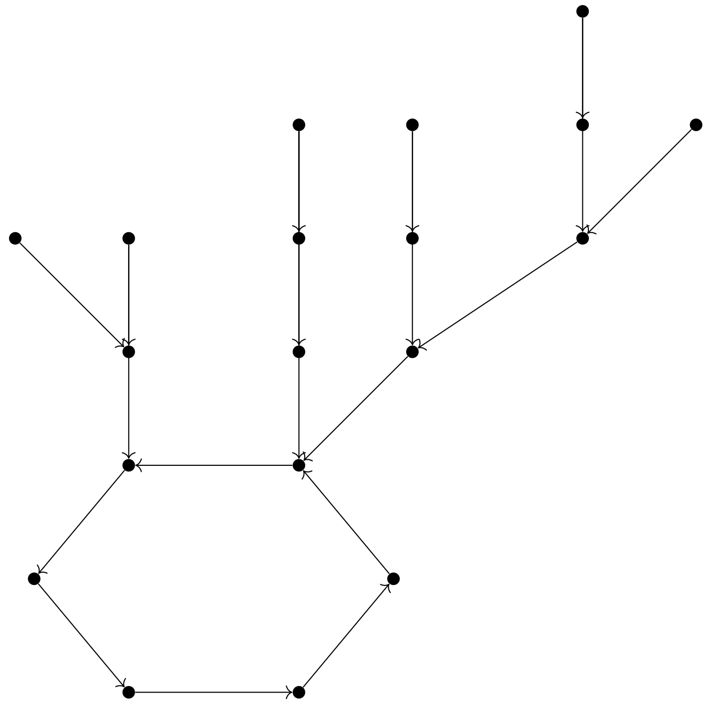
K-linearer Abschluss einer Kategorie

Pfadalgebra = Kategorien-Algebra path algebra = 1 Object, welches eine Algebra ist. Dabei verliert man wieder die Informationen über die mehreren Objekte.

So wie Menge ein Erz-system eines Monoid.

9 Relations of the Algebroid

9.1 Relations of endomorphisms



Lemma 9.1.1 (σ-Lemma). *For each endomorphism f in a finite concrete category $\mathcal{C}$ there exist $m, n \in \mathbb{N}$ such that $f^{(m+n)} = f^m$.*

Beschreibung der Algorithmen
WeakDirectSumDecomposition ¡– Tiefensuche. Objekte (Funktoren) in indecomposable Functors.

10 Category

Definition 10.0.1. (Quiver)
A quiver A consists of a class of objects (or vertices) $A_0 = \mathrm{Obj} A$ and a class of morphisms (or arrows) $A_1 = \mathrm{Mor} A$ together with two defining maps

$$s, t \colon A_1 \rightrightarrows A_0$$

s called source and t called target.[1]
We write $\mathrm{Hom}_A(M, N)$ (sometimes also $A(M, N)$) for the fiber $(s, t)^{-1}(\{(M, N)\})$ of the product map $(s, t) : A_1 \to A_0 \times A_0$ over the pair $(M, N) \in A_0 \times A_0$.
This is the class of all morphisms with source = M and target = N.
For a morphism $\varphi \in \mathrm{Hom}_A(M, N)$ we write

$$\varphi : M \longrightarrow N \text{ or } M \xrightarrow{\varphi} N$$

Clearly A_1 is the disjoint union $\dot{\bigcup}_{M,N \in A_0} \mathrm{Hom}_A(M, N) = A_1$. As usual we define $\mathrm{End}_A(M) := \mathrm{Hom}_A(M, M)$.

Definition 10.0.2. (Category)
A category $\mathfrak{A}$ is a quiver with two further defining maps

$$A_0 \xrightarrow{1} A_1 \xleftarrow[\mu]{} A_1 \times_{s, A_0, t} A_1$$

Example 10.0.3. (Representation of a concrete category)

$$\begin{tikzcd}
5 \arrow[r, "\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}"] \arrow[loop below, "\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}"] & 4 \arrow[loop below, "\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}"]
\end{tikzcd}$$

$$nine \Uparrow$$

$$\begin{tikzcd}
1 \arrow[r, "b"] \arrow[loop below, "a"] & 2 \arrow[loop below, "c"]
\end{tikzcd}$$

$$\begin{tikzcd}
\{1,2,3\} \arrow[r, "(4{,}5{,}6)"] \arrow[loop below, "(2{,}1{,}3)"] & \{4,5,6\} \arrow[loop below, "(5{,}6{,}4)"]
\end{tikzcd}$$

$$F(a)\eta_1 = \eta_1 G(a) F(b)\eta_2 = \eta_1 G(b)$$

11 $\mathbb{K}$-linear Category (Algebroid)

Group: Category with one object.

Groupoid: A small category in which every morphism is an isomorphism.

Algebroid

EmbeddingOfSumOfImages

What is an Algebroid? Bialgebroid?

12 Additive Category

13 Abelian Category

14 The Category of Categories

15 The Categories of Functors

16 The Representation of a Category

17 Representation

Grundidee von FunctorCategory
Standard-Monoidale Struktur von der Zielkategorie z.B. TensorUnit(C)

18 Algorithms

```
    AddInverse( C,
      function( alpha )
        return Inverse( UnderlyingCell( alpha ) ) / CapCategory( alpha );
    end );

    c := ConcreteCategory( L );

    C!.ConcreteCategoryRecord := c;

    objects := List( c.objects, FinSet );

    SetSetOfObjects( C, List( objects, o -> o / C ) );

    SetSetOfGeneratingMorphisms( C, List( c.generators, g -> ConvertToMapOfFinSets( obje

    Finalize( C );

    return C;

end );

##
InstallMethod( Algebroid,
        "for a homalg ring and a finite category",
        [ IsHomalgRing and IsCommutative, IsFiniteConcreteCategory ],

  function( k, C )
    local objects, gmorphisms, q, kq, relEndo, A, F, vertices, rel,
          func, st, s, t, homST, list, p, pos;

    objects := SetOfObjects( C );
    gmorphisms := SetOfGeneratingMorphisms( C );
    q := RightQuiverFromConcreteCategory( C );
    kq := PathAlgebra( k, q );
    relEndo := RelationsOfEndomorphisms( k, C );
    A := Algebroid( kq, relEndo );
    kq := UnderlyingQuiverAlgebra( A );
    F := CapFunctor( A, objects, gmorphisms, C );
```

```
    vertices := List( SetOfObjects(A), UnderlyingVertex );

    rel := [];
    func :=
      function( p, l )
        return ForAny( l, p1->
                        IsCongruentForMorphisms(
                                ApplyToQuiverAlgebraElement( F, p ),
                                ApplyToQuiverAlgebraElement( F, p1 ) )
                        );
    end;

    for st in Cartesian(vertices,vertices) do
        s := st[1];
        t := st[2];
        if s = t then
            continue;
        fi;
        homST := BasisPathsBetweenVertices( kq, s, t );
        homST := List( homST, p -> PathAsAlgebraElement( kq, p ) );

        list := [];

        for p in homST do
            pos := PositionProperty( list, l->func(p,l) );
            if IsInt(pos) then
                Add( list[pos], p );
            else
                Add( list, [p] );
            fi;
        od;
        list := List( list, l-> List( l, p -> p!.representative ) );
        Append( rel, list );
    od;

    rel := Filtered( rel, l -> Length(l)>1 );
    rel := List( rel, l -> List( l{[ 2 .. Length(l) ]}, p -> l[1]-p ) );
    rel := Flat( rel );
    rel := Concatenation( relEndo, rel );

    kq := PathAlgebra( kq ) / rel;

    kq := PathAlgebra( kq ) / GroebnerBasis( IdealOfQuotient( kq ) );
```

We want the endomorphism relations so that the path algebra is finite-dimensional and we get a finite Gröbner basis.

Proof that algorithm is correct Proof that it terminates.

Wir haben BasisOfExternalHom benutzt um Decompose in CAP umzusetzen um EmbeddingOfSubRepresentation umzusetzen um WeakDirectSumDecomposition umzusetzen.

Notes

[1]Some authors use maps t, h for *tail* and *head* instead of source and target, defining the arrows to go from the tail to the head. This use of t as the starting point instead of the end target as in our definition can lead to some confusion.

Algorithm 1: `RightQuiverFromConcreteCategory`

Input : a finite concrete category C with n objects
Output : the right quiver $q(n)$

let Obj be the set of objects of C;
let $n := Length(Obj)$;
let $gMor$ be the set of generating morphisms of C;
let A be the empty set and let $i := 1$;
foreach *morphism mor in gMor* **do**
  let $A_{i,1}$ be the position of $Source(mor)$ in Obj;
  let $A_{i,2}$ be the position of $Range(mor)$ in Obj;
  let $i := i + 1$;
end
let q be the right quiver with vertices $\{1, \ldots, n\}$ and arrows A.
return q;

Algorithm 2: `RelationsOfEndomorphisms`

Input: a commutative ring k and a finite concrete category C
Output: the endomorphism relations of the category C

let $q :=$ `RightQuiverFromConcreteCategory`(C);
let kq be the path algebra generated by k and q;
let $gMor$ be the set of generating morphisms of C;
let $A := Arrows(q)$;
let $relsEndo$ be the empty set;
foreach $i = 1, \ldots, Length(gMor)$ **do**
  let $mor := gMor_i$ **if** *mor is not an endomorphism* **then**
    continue;
  end
  let $m := 0$ and let $powers$ be the empty set;
  let $foundEqual$ be false;
  while $mor^m \notin powers$ **do**
    let $n := 1$;
    while $\neg foundEqual$ **do**
      if $mor^{(m+n)} = mor^m$ **then**
        Add the relation $kq.(A_i)^{(m+n)} - kq.(A_i)^m$ to relsEndo;
        foundEqual := true;
      end
      n := n+1;
    end
    Add mor^m to powers;
    m := m+1;
  end
end
return *relsEndo*;

References

[1] https://web.northeastern.edu/martsinkovsky/p/Parnu2019/slides-facchini.pdf

[2] https://www.math.uni-bielefeld.de/ sek/kau/leit4.pdf

[3] Jan Geuenich. https://hss.ulb.uni-bonn.de/2017/4681/4681.pdf