Classification of finite semigroups and categories using computational methods

Najwa Ghannoum
W. Fussner, T. Jakl, and C. Simpson

Université Côte d'Azur
LJAD

AITP 2020
September 16, 2020

Content

Part 1 - Background

Motivations

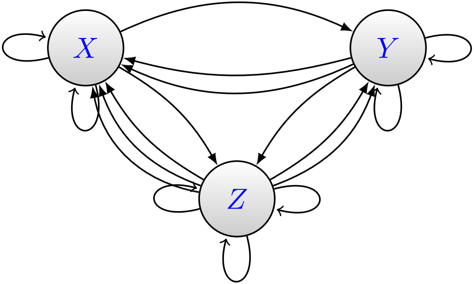
1. Associative structures:
   - *Combinatorial results in groups, monoids,...*
   - *Study finite categories.*
   - *Understand deeply the enumeration and classification problems in associative algebra.*

Motivations

1. Associative structures:
    - *Combinatorial results in groups, monoids,...*
    - *Study finite categories.*
    - *Understand deeply the enumeration and classification problems in associative algebra.*
2. Previous work:
    - A. Distler, T. Kelsey (2009): *The monoids of orders eight, nine and ten.*
    - S. Allouch, C. Simpson (2017): *Classification of categories with matrices of coefficient 2 and order n.*

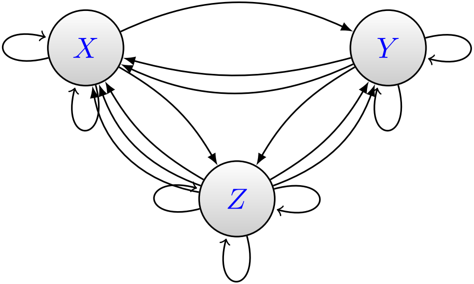
Category associated to a positive square matrix

Every finite category is associated to a square matrix. The entries of the matrix are the number of morphisms between each two objects.



Category associated to a positive square matrix

Every finite category is associated to a square matrix. The entries of the matrix are the number of morphisms between each two objects.



$$\begin{array}{c} \\ X \\ Y \\ Z \end{array} \begin{array}{c} \begin{array}{ccc} X & Y & Z \end{array} \\ \begin{pmatrix} 2 & 1 & 1 \\ 2 & 2 & 1 \\ 3 & 2 & 3 \end{pmatrix} \end{array}$$

In generality: let $\mathcal{M} \in \mathcal{M}_n(\mathbb{N})$ defined by:

$$\mathcal{M} = \begin{pmatrix} m_{11} & m_{12} & \cdots & m_{1n} \\ m_{21} & m_{22} & \cdots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nn} \end{pmatrix}$$

In generality: let $\mathcal{M} \in \mathcal{M}_n(\mathbb{N})$ defined by:

$$\mathcal{M} = \begin{pmatrix} m_{11} & m_{12} & \cdots & m_{1n} \\ m_{21} & m_{22} & \cdots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nn} \end{pmatrix}$$

Definition

Let A be a finite ordered category of order n whose objects are $\{x_1, ..., x_n\}$; we say that A is a category associated to $\mathcal{M}$ if: $|(x_i, x_j)| = m_{ij}, \forall i, j \in \{1, ..., n\}$.

In generality: let $\mathcal{M} \in \mathcal{M}_n(\mathbb{N})$ defined by:

$$\mathcal{M} = \begin{pmatrix} m_{11} & m_{12} & \cdots & m_{1n} \\ m_{21} & m_{22} & \cdots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nn} \end{pmatrix}$$

Definition

Let A be a finite ordered category of order n whose objects are $\{x_1, ..., x_n\}$; we say that A is a category associated to $\mathcal{M}$ if: $|(x_i, x_j)| = m_{ij}, \forall i, j \in \{1, ..., n\}$.

Going from a category towards a matrix can be done in a unique way. Inversely, it can be done in several ways.

Relationship between finite categories and matrices

The previously demonstrated association between finite categories and matrices can be also represented as a function:

$$\begin{array}{rccc} F: & \mathsf{FinCat} & \rightarrow & \mathsf{Mat} \\ & \mathcal{C} & \mapsto & M \end{array}$$

Relationship between finite categories and matrices

The previously demonstrated association between finite categories and matrices can be also represented as a function:

$$\begin{array}{rccc} F: & \mathsf{FinCat} & \to & \mathsf{Mat} \\ & \mathcal{C} & \mapsto & M \end{array}$$

- F is not injective.

Relationship between finite categories and matrices

The previously demonstrated association between finite categories and matrices can be also represented as a function:

$$\begin{array}{rccc} F: & \mathsf{FinCat} & \to & \mathsf{Mat} \\ & \mathcal{C} & \mapsto & M \end{array}$$

- F is not injective.
  - **Monoids:** they have different pre-images. ((2) admits 2 monoids).

Relationship between finite categories and matrices

The previously demonstrated association between finite categories and matrices can be also represented as a function:

$$\begin{array}{rccc} F: & \mathsf{FinCat} & \to & \mathsf{Mat} \\ & \mathcal{C} & \mapsto & M \end{array}$$

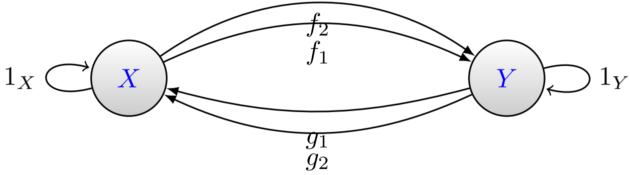
- F is not injective.
  - **Monoids:** they have different pre-images. ((2) admits 2 monoids).
- F is not surjective.

Relationship between finite categories and matrices

The previously demonstrated association between finite categories and matrices can be also represented as a function:

$$\begin{array}{rccc} F: & \mathsf{FinCat} & \to & \mathsf{Mat} \\ & \mathcal{C} & \mapsto & M \end{array}$$

- F is not injective.
  - **Monoids:** they have different pre-images. ((2) admits 2 monoids).
- F is not surjective.
  - **Example:** $\begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$ has no pre-image by F. If not, then:



$g_1 = g_1 \circ (f_1 \circ g_2) = (g_1 \circ f_1) \circ g_2 = g_2$. **Contradiction!**

Brief review of the literature

1. In 2008, Berger and Leinster proved that every square matrix of positive integers whose diagonal entries are all at least 2 admits a category.
2. In 2014, Allouch and Simpson have shown that a category is associated to a square matrix under certain conditions on the coefficients of the matrices in terms of their determinants.
   - For example, if we take a matrix M of size 2 and if one of the diagonals is 1, then the determinant $det(M) \geq 1$.
   - As in general if we have 1 on the diagonals and it's unique, we do the determinant of every submatrix of order 2.
   - If there is more than one "1", then no category.

Matrices whose all coefficients are 2

Now that we have the necessary and sufficient conditions on the matrix coefficients required to obtain at least one finite category, we ask the question: how many are there?
In a previous work for Allouch and Simpson, they were able to count the number of finite categories associated to matrices whose all coefficients are 2:

- For size 2: there is one category.
- For size 3: there are 5 categories.
- As for size n, they weren't able to obtain the exact number of associated categories, however they were able to bound this number between $\frac{2^{[\frac{n}{3}]^3}}{n!}$ and $18^{C_n^3}$ (where C_n^3 is the 3-combination of n).

Part 2 - Obtaining the data

- We want to push the counting problem further and we start by counting finite categories associated to matrices whose all coefficients are 3 instead of 2.
- For that reason, we rewrite finite categories as certain expanded semigroups, then use Mace4 for listing all the models of the categories, and this is how we obtain the data.
- We feed Mace4 equations that are generated by a Python script that we optimized for the purpose.
- We need to pay attention of the way of writing equations, because it may lead to keep the program running with no end.

Representation in terms of semigroups

A *finite category* $\mathcal{C} = (O, \mathfrak{m}, d, r, 1_{(-)}, \circ)$ consists of the following data:

1. A finite set of objects O.
2. A finite set of morphisms $\mathfrak{m}$.
3. Functions $d, r : \mathfrak{m} \to O$ and $1_{(-)} : O \to \mathfrak{m}$.
4. A partial function $\circ : \mathfrak{m} \times \mathfrak{m} \rightharpoonup \mathfrak{m}$ with $domain = \{(f, g) \in \mathfrak{m} \times \mathfrak{m} \mid r(f) = d(g)\}$.

A *compositional semigroup* $(S, \circ, 0, \mathfrak{u})$ consists of the following:

1. $0 \notin \mathfrak{u}$.
2. For all $e \in \mathfrak{u}, e \circ e = e$.
3. For all $x \in S$, there exist unique $e, e' \in \mathfrak{u}$ such that $e \circ x \neq 0$ and $x \circ e' \neq 0$.
4. There exists $e \in \mathfrak{u}$ such that $f \circ e \neq 0$ and $e \circ g \neq 0$ if and only if $f \circ g \neq 0$.
5. $e_1 \circ f \circ e_2 \neq 0$ and $e_2 \circ g \circ e_3 \neq 0$ if and only if $e_1 \circ (f \circ g) \circ e_3 \neq 0$.
6. $d(f) = e$ implies that $e \circ f = f$ and $r(f) = e$ implies that $f \circ e = f$.

Definition

The category of finite categories denoted by FinCat consists of the following:

1. Objects: finite categories.
2. Morphisms: functors.

Definition

The category of finite categories denoted by FinCat consists of the following:

1. Objects: finite categories.
2. Morphisms: functors.

Definition

The category of compositional semigroups denoted by CSem consists of the following data:

1. Objects: compositional semigroups.
2. Morphisms: $h : (S_1, \circ_1, 0_1, \mathfrak{u}_1) \to (S_2, \circ_2, 0_2, \mathfrak{u}_2)$, where:
   - $h(x \circ_1 y) = h(x_1) \circ_2 h(y)$.
   - $h[\mathfrak{u}_1] \subseteq \mathfrak{u}_2$.
   - $h(x) = 0 \iff x = 0$.

Theorem

CSem and FinCat are equivalent.

Lemma

Two objects in CSem are isomorphic iff they are isomorphic in the signature $(\circ, 0, \mathfrak{u})$.

Python script

```
testing = False

# input = tuple( [ tuple([3]) ])  # this is how you write matrix 1x1

input = ( (3, 3)
        , (3, 3) )

# input = ( (3, 3, 3)
#         , (3, 3, 3)
#         , (3, 3, 3) )
```

Python script

```
# Units, from the right
for j in obj:
    for i in obj:
        for h in hom_values(j,i):
            print (p(j, i, h) + " * " + p(i, i, unit) + " = " + p(j
                , i, h) + ".")

print


if forbidCompositionsEqualToUnits:
    # The law that forbids certain compositions to be equal to the
        units, from the left
    for i in obj:
        for j in obj:
            if i != j:
                for h in hom_values(i,j):
                    for g in hom_values(j,i):
                        print (p(i,j,h) + " * " + p(j,i,g) + " != "
                            + p(i,i,unit) + ".")

    print
```

Python script

```
# Forcing composition to have the right type
for i in obj:
    for j in obj:
        for k in obj:
            for h in hom_values(i,j):
                for g in hom_values(j,k):
                    if (i == j and h == unit) or (j == k and g ==
                        unit):
                        continue

                    options = [ p(i,j,h) + " * " + p(j,k,g) + " = "
                        + p(i,k,l) for l in hom_values(i,k)]
                    print (" | ".join(options) + ".")

            print
```

Python script

```
# If the composition doesn't typecheck, make it fail

for i in obj:
    for j in obj:
        for k in obj:
            if j == k: continue

            for l in obj:
                for h in hom_values(i,j):
                    for g in hom_values(k,l):
                        print (p(i,j,h) + " * " + p(k,l,g) + " = 0
                            .")
```

Equations in Mace4

```
% hash[AA1] = 1
% hash[AA2] = 2
% hash[AA3] = 3
% hash[AB1] = 4
% hash[AB2] = 5
% hash[AB3] = 6
% hash[BA1] = 7
% hash[BA2] = 8
% hash[BA3] = 9
% hash[BB1] = 10
% hash[BB2] = 11
% hash[BB3] = 12

x * (y * z) = (x * y) * z.
0 * x = 0.
x * 0 = 0.

1 * 1 = 1.
1 * 2 = 2.
1 * 3 = 3.
1 * 4 = 4.
1 * 5 = 5.
1 * 6 = 6.
10 * 7 = 7.
10 * 8 = 8.
10 * 9 = 9.
10 * 10 = 10.
10 * 11 = 11.
10 * 12 = 12.

1 * 1 = 1.
2 * 1 = 2.
3 * 1 = 3.
4 * 10 = 4.
5 * 10 = 5.
6 * 10 = 6.
7 * 1 = 7.
8 * 1 = 8.
9 * 1 = 9.
10 * 10 = 10.
11 * 10 = 11.
12 * 10 = 12.
```

Output of Mace4

```
interpretation( 13, [number = 1,seconds = 0], [
    function(*(_,_), [
          0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
          0, 1, 2, 3, 4, 5, 6, 0, 0, 0, 0, 0, 0,
          0, 2, 2, 2, 4, 4, 4, 0, 0, 0, 0, 0, 0,
          0, 3, 2, 1, 4, 5, 6, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 4, 4, 4,
          0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 5, 4, 4,
          0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 6, 4, 4,
          0, 7, 7, 7,11,11,11, 0, 0, 0, 0, 0, 0,
          0, 8, 7, 8,11,11,11, 0, 0, 0, 0, 0, 0,
          0, 9, 7, 9,11,11,11, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 0, 0, 0, 7, 8, 9,10,11,12,
          0, 0, 0, 0, 0, 0, 0, 7, 7, 7,11,11,11,
          0, 0, 0, 0, 0, 0, 0, 7, 7, 7,12,11,11])]).
interpretation( 13, [number = 2,seconds = 0], [
    function(*(_,_), [
          0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
          0, 1, 2, 3, 4, 5, 6, 0, 0, 0, 0, 0, 0,
          0, 2, 2, 2, 4, 4, 4, 0, 0, 0, 0, 0, 0,
          0, 3, 2, 1, 4, 5, 6, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 4, 4, 4,
          0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 5, 4, 4,
          0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 6, 4, 4,
          0, 7, 7, 7,11,11,11, 0, 0, 0, 0, 0, 0,
          0, 8, 7, 8,11,11,11, 0, 0, 0, 0, 0, 0,
          0, 9, 7, 9,11,11,11, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 0, 0, 0, 7, 8, 9,10,11,12,
          0, 0, 0, 0, 0, 0, 0, 7, 7, 7,11,11,11,
          0, 0, 0, 0, 0, 0, 0, 7, 7, 8,12,11,11])]).
```

Theorem

The number of finite categories associated to the matrix $\begin{pmatrix} 3 & 3 \\ 3 & 3 \end{pmatrix}$ is 362.

Part 3 - Analyzing the data

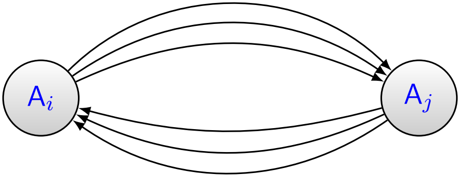
Computing using Mace4

We can identify objects in a category with their endomorphism monoids, and try to classify them from this point of view.
There are 7 monoids of size 3:

comp	A_1	A_2	A_3	A_4	A_5	A_6	A_7
$2 \circ 2 =$	1	2	2	2	2	2	3
$3 \circ 3 =$	3	2	3	3	2	3	2
$2 \circ 3 =$	3	2	2	3	3	2	1
$3 \circ 2 =$	3	2	3	2	3	2	1

These are the possible multiplications of the elements $\{1, 2, 3\}$.
Each column is a monoid denoted by A_i.

Computing using Mace4



We combine monoids of 3 elements together in one category, two monoids $\mathsf{A}_i, \mathsf{A}_j$ are called connected if there exists a category where the monoids A_i and A_j are the endomorphism monoids of the two objects. Viewing them as objects, each object is one of the monoid of endomorphisms listed before, and this graph of a category is associated to the matrix:

$$\begin{pmatrix} 3 & 3 \\ 3 & 3 \end{pmatrix}$$

- Each entry in the table represents the number of categories obtained when A_i and A_j are connected.

	A_1	A_2	A_3	A_4	A_5	A_6
A_1	10	21	2	2	0	36
A_2	–	62	4	4	0	76
A_3	–	–	2	0	0	8
A_4	–	–	–	2	0	8
A_5	–	–	–	–	2	0
A_6	–	–	–	–	–	123

- The highest numbers of categories are obtained when we have zero semigroups and semilattices as objects.
- The lowest numbers of categories are obtained when we have rectangular bands as objects.
- A_3 and A_4 are not connected.
- A_5 is only connected to itself.

The patterns in the data above inspired us to make conjectures and helped us prove many of them. We present a proof that A_5 is only connected to A_5.

Theorem

Let $M = \begin{pmatrix} 3 & b \\ c & 3 \end{pmatrix}$ and A be the category associated to M whose objects are X and Y.
$\mathsf{A}(X, X) = \mathsf{A}_5$ iff $\mathsf{A}(Y, Y) = \mathsf{A}_5$.

Proof (Sketch).

The idea is to construct a subcategory. Take the "group part" (i.e. removing the identity since $\mathsf{A}_5 \setminus \{1\} = \mathbb{Z}_2$) of the monoid A_5 and choose the sets of morphisms from X to Y and from Y to X in a way where the identity of the group acts like an identity on them too. The matrix becomes $\begin{pmatrix} 3 & b' \\ c' & 2 \end{pmatrix}$. Then under some conditions that are imposed on the coefficients, the second object is forced to be A_5.

The lemmas below are results used in proving the previous theorem, and were also inspired by the data in the table.

Lemma

Let $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and A be a category associated to M whose objects are X and Y where $G = \mathsf{A}(Y, Y)$ is a group of order d. G acts freely on the sets $\mathsf{A}(X, Y)$ and $\mathsf{A}(Y, X)$.

Lemma

Let $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and A be the category associated to M whose objects are X and Y where $\mathsf{A}(Y, Y)$ is a group of order d. Then:

1. b, c are multiples of d.
2. $a \geq \frac{bc}{d} + 1$.

Same for

Theorem

Let $M = \begin{pmatrix} 3 & a_{12} & \dots & a_{1n} \\ a_{21} & 3 & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & \dots & \dots & 3 \end{pmatrix}$ and let A be a reduced category associated to M With $Ob(\mathsf{A}) = \{1, ..., n\}$.
If there exists $i \in Ob(\mathsf{A})$ such that $\mathsf{A}(i,i) = \mathsf{A}_5$ then $\mathsf{A}(j,j) = \mathsf{A}_5 \quad \forall j \in Ob(\mathsf{A})$

Future Perspectives

1. The data suggests lots of other structural properties of finite categories, and ongoing work to enumerate categories associated with other matrices will lead to more conjectures.
2. Current work in our group seeks to apply techniques from neural networks to obtain approximate counts on the number of semigroups of given cardinality.
3. The most difficult combinatorial questions in the present work seem to be related to semigroups, we expect that similar techniques from AI further the enumeration and the classification here.

Thank you for your attention!