computation

MDPI

Article

Cayley Hash Values of Brauer Messages and Some of Their Applications in the Solutions of Systems of Differential Equations

María Alejandra Osorio Angarita [1], Agustín Moreno Cañadas [2], Cristian Camilo Fúneme [2], Odette M. Mendez [2] and Robinson-Julian Serna [3,*]

[1] Escuela de Ingeniería de Sistemas, Facultad de Ingeniería, Universidad Pedagógica y Tecnológica de Colombia, Avenida Central del Norte 39-115, Tunja 150003, Colombia
[2] Departamento de Matemáticas, Universidad Nacional de Colombia, Edificio Yu Takeuchi 404, Kra 30 No 45-03, Bogotá 11001000, Colombia
[3] Escuela de Matemáticas y Estadística, Universidad Pedagógica y Tecnológica de Colombia, Avenida Central del Norte 39-115, Tunja 150003, Colombia
* Correspondence: robinson.serna@uptc.edu.co

Abstract: Cayley hash values are defined by paths of some oriented graphs (quivers) called Cayley graphs, whose vertices and arrows are given by elements of a group $\mathfrak{H}$. On the other hand, Brauer messages are obtained by concatenating words associated with multisets constituting some configurations called Brauer configurations. These configurations define some oriented graphs named Brauer quivers which induce a particular class of bound quiver algebras named Brauer configuration algebras. Elements of multisets in Brauer configurations can be seen as letters of the Brauer messages. This paper proves that each point $(x,y) \in \mathcal{V} = \mathbb{R}\backslash\{0\} \times \mathbb{R}\backslash\{0\}$ has an associated Brauer configuration algebra $\Lambda_{\mathfrak{B}^{(x,y)}}$ induced by a Brauer configuration $\mathfrak{B}^{(x,y)}$. Additionally, the Brauer configuration algebras associated with points in a subset of the form $(\lfloor(x)\rfloor, \lceil(x)\rceil] \times (\lfloor(y)\rfloor, \lceil(y)\rceil] \subset \mathcal{V}$ have the same dimension. We give an analysis of Cayley hash values associated with Brauer messages $\mathfrak{M}(\mathfrak{B}^{(x,y)})$ defined by a semigroup generated by some appropriated matrices $A_0, A_1, A_2 \in \mathrm{GL}(2,\mathcal{R})$ over a commutative ring $\mathcal{R}$. As an application, we use Brauer messages $\mathfrak{M}(\mathfrak{B}^{(x,y)})$ to construct explicit solutions for systems of linear and nonlinear differential equations of the form $X''(t) + MX(t) = 0$ and $X'(t) - X^2(t)N(t) = N(t)$ for some suitable square matrices, M and $N(t)$. Python routines to compute Cayley hash values of Brauer messages are also included.

Keywords: Brauer configuration algebra; Brauer message; Cayley graph; Cayley hash; path algebra; quiver representation

Citation: Osorio Angarita, M.A.; Cañadas, A.M.; Fúneme, C.C.; Mendez, O.M.; Serna, R.-J. Cayley Hash Values of Brauer Messages and Some of Their Applications in the Solutions of Systems of Differential Equations. *Computation* **2022**, *1*, 164. https://doi.org/10.3390/computation10090164

Academic Editors: Akbar Ali, Guojun Li, Mingchu Li, Rao Li, Colton Magnant and Madhumangal Pal

Received: 19 August 2022
Accepted: 14 September 2022
Published: 17 September 2022

1. Introduction

Hash functions are compression functions that are easy to compute. They are helpful tools in several cryptographic tasks. Ensuring data integrity and password authentication systems are some of the primary roles of hash functions. Data integrity provided by hash functions prevents undesirable data modification [1].

To date, it is believed that well-posed hash functions such as SHA256 are unbreakable by several attacks, including those conducted by a quantum computer.

The *Cayley graph* $\mathcal{C}_{\mathfrak{H},\mathfrak{s}} = (V, E)$ associated with a subset $\mathfrak{s}$ of a multiplicative (semi)group $\mathfrak{H}$ is a k-regular graph, whose set of vertices V are in bijective correspondence with the set $\mathfrak{H}$. In this case, if $v_{g_1}, v_{g_2} \in V$ are vertices corresponding to the elements $g_1, g_2 \in \mathfrak{H}$, then there is an arrow connecting v_{g_1} and v_{g_2} provided that there is an element $s \in \mathfrak{s}$, such that $g_1^{-1} g_2 s^{-1} = 1$, where 1 is the identity element of the group $\mathfrak{H}$. The idea behind the use of Cayley graphs in cryptography is to produce hash functions that are collision resistant.

The input to these functions gives directions for walking around the graph, and the output is the ending vertex of the walk [2].

On the other hand, Brauer messages were introduced by Espinosa et al. [3] in their research regarding some bound quiver algebras called Brauer configuration algebras by Green and Schroll [4]. This paper provides Cayley hash values to some of these Brauer messages.

Brauer messages have applications in different science fields. For instance, they were used by Cañadas et al. in the graph energy theory to compute the trace norm of some matrices and in the war games theory to declare winners and losers in war games based on the behavior of a missile defense system (MDS) [5]. Furthermore, mutations of Brauer messages can be used to give an algebraic interpretation of the Advanced Encryption Standard (AES) key schedule [6].

1.1. Motivations

This paper uses an interaction between cryptography, algebraic geometry, combinatorics, and the theory of representation of associative algebras to obtain applications of the theory of Brauer configuration algebras in cryptography and differential equations.

Relationships between algebraic geometry and Brauer configuration algebras are motivated by a recent paper published by Green and Schroll [7], who proved that each point in an affine variety has an associated suitable associative algebra and that all these algebras have the same dimension. As an interpretation of this result, we prove that, associated with each point in $\mathbb{R}^* \times \mathbb{R}^*$, there is a suitable Brauer configuration algebra and that Brauer configuration algebras associated with points in a set of the form $(m, m+1] \times (n, n+1], m, n \in \mathbb{Z}$ have the same dimension. The messages arising from these Brauer configurations (called Brauer messages) are used to provide solutions for linear and nonlinear systems of differential equations. We compute Cayley hash values of Brauer messages establishing for which integer numbers such hash values are collision-resistant.

Henceforth, we outline the main results presented in this paper and how previous works are used to obtain them.

1.2. Contributions

We prove that each point $(x, y) \in \mathcal{V} = \mathbb{R}\backslash\{0\} \times \mathbb{R}\backslash\{0\}$ is associated with a suitable Brauer configuration algebra. Properties of these algebras are also established. In particular, it is proved that their dimensions can be computed by enumerating suitable lattice paths. Similar conditions have their corresponding centers.

As an application, we interpret message specializations of these Brauer configurations as appropriated matrices with applications in Catalan combinatorics and differential equations. It is also proved that some of these matrices are solutions to some linear and nonlinear systems of differential equations of the form $X''(t) + 2^{\gamma-1}X(t) = 0$ and $X'(t) - X^2(t)\mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta)t = \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta)t$, where $\mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta)$ is a matrix whose entries are given by appropriated Brauer messages. It is worth noting that finding explicit solutions to nonlinear differential equations is in general a cumbersome problem.

Cayley hash functions based on matrix semigroups are applied to some Brauer messages to prove that these hash families are collision resistant.

Python routines to compute Cayley hash values of Brauer messages associated with points in the plane are also introduced.

Figure 1 shows how previous works, mutations, and Brauer configuration algebras (BCAs) can be related to obtain the main results (targets of red arrows) presented in this paper.

In Section 2.1.1, we recall the main results regarding BCAs; Proposition 1 and Theorem 2 give formulas for the dimensions of these algebras and their centers.

Theorem 3 provides the properties of BCAs associated with points in the plane. Their dimensions and combinatorial data regarding indecomposable projective modules over these algebras are given.

Theorem 4 describes arithmetic properties of specialized matrices given by mutations of Brauer configurations associated with the plane.

Corollary 1 provides explicit solutions of linear and nonlinear systems of differential equations.

Section 4 is devoted to Cayley hash functions. Theorem 5 gives Cayley hash values for appropriated matrix semigroups. These results allow to give Cayley hash values of Brauer messages. Corollary 2 proves that Sosnovski hash functions are collision-resistant when they are applied to Brauer messages.

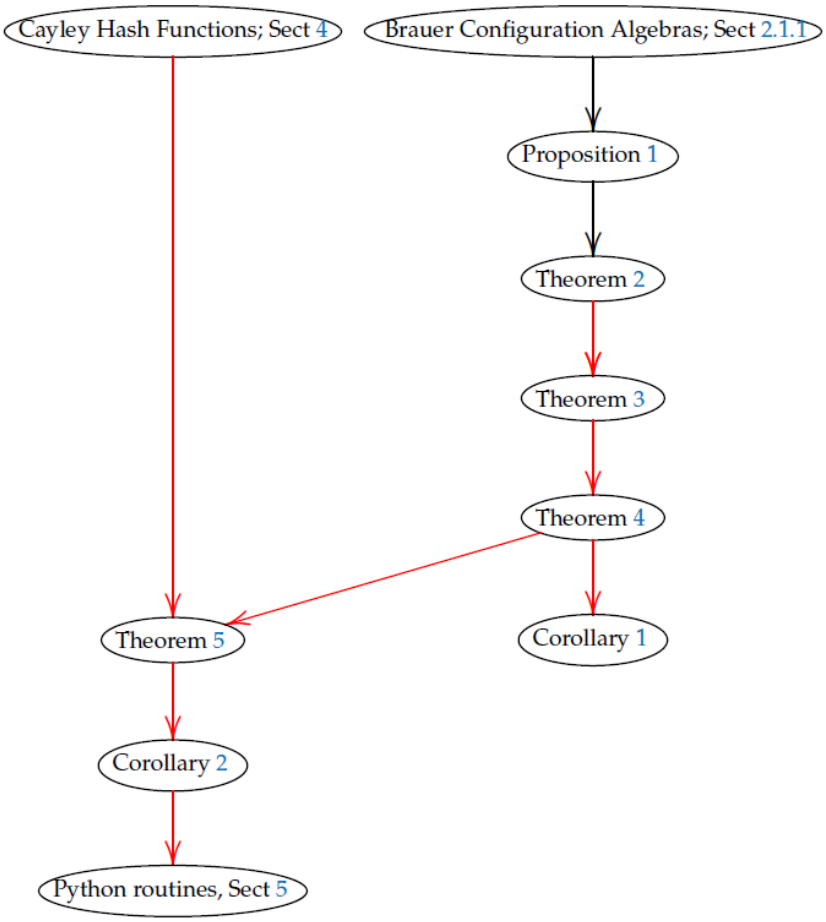

Figure 1. This graph shows how the background relates to the main results presented in this paper.

This paper is distributed as follows: Section 2 is devoted to recalling definitions and notation used throughout the document. In particular, we recall the notion of Brauer configuration algebra. In Section 3, we give our main results. We associate to each point $(x, y) \in \mathcal{V}$ an apropriated Brauer configuration algebra determining which of them have the same dimension. Mutations are defined for the variables associated with these algebras. Such mutations give rise to new classes of matrices with applications in combinatorics and differential equations. Section 4 is devoted to the values of Cayley hash functions associated with Brauer messages. Experimental data are given in Section 5. Concluding remarks and possible future works are described in Section 6. Appendix A gives Python routines to compute Brauer messsages, their Sosnovski hash values, and polygons of Brauer configurations associated with the plane.

2. Background and Related Work

In this section, we introduce some definitions and notation to be used throughout the paper. In particular, a brief overview regarding hash functions and Brauer configuration algebras is given [4].

Henceforth, the symbol $\mathbb{F}$ ($\mathbb{C}$) denotes a field (the complex numbers field), and $\lceil(x)\rceil$ ($\lfloor(x)\rfloor$) denotes the smallest (greatest) integer greater (less) than or equal to a real number x. In path algebras $\mathbb{F}Q$, it is assumed that $\mathbb{F}$ is an algebraic closed field.

2.1. Related Work on Cayley Hash Functions

This section gives a background on hash functions, in particular, Cayley hash functions.

Hash functions are easy-to-compute compression functions, as described before. Such functions are used in several contexts. For instance, they are helpful in password management systems. Servers that authenticate user passwords save a one-way hash associated with a unique password so that if an attacker steals the database, it may be unfeasible for the attacker to recover the original password as plaintext [1,8].

A hash family is a four-tuple $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$, where

- $\mathcal{X}$ is a set of possible messages, which could be finite or infinite.
- $\mathcal{Y}$ is a finite set of possible message digests or authentication tags.
- $\mathcal{K}$ is the set of keys.
- For each $k \in \mathcal{K}$, there is a hash function $h_K : \mathcal{X} \to \mathcal{Y} \in \mathcal{H}$. If $|\mathcal{X}|$ and $|\mathcal{Y}|$ denote the cardinals of $\mathcal{X}$ and $\mathcal{Y}$, and $2|\mathcal{Y}| \leq |\mathcal{X}|$, then h_k is said to be a compression function. If $\mathcal{X} = \mathcal{Y}$, and the hash function h is the identity, then h is said to be an unkeyed hash function [1,2].

A hash function is said to be secure if the following three problems are difficult to solve:

Preimage

Instance: A hash function $h : \mathcal{X} \to \mathcal{Y}$ and an element $y \in \mathcal{Y}$.
Find: $x \in \mathcal{X}$ such that $h(x) = y$.

Second Preimage

Instance: A hash function $h : \mathcal{X} \to \mathcal{Y}$ and an element $x \in \mathcal{X}$.
Find: $x' \in \mathcal{X}$ such that $x' \neq x$ and $h(x') = h(x)$.

Collision

Instance: A hash function $h : \mathcal{X} \to \mathcal{Y}$ and an element $x \in \mathcal{X}$.
Find: $x, x' \in \mathcal{X}$ such that $x' \neq x$ and $h(x') = h(x)$.

Hash functions are used to construct a short fingerprint or digest the message of some data. If an attacker alters the data, then the fingerprint will no longer be valid. One of the most used methods to construct iterated hash functions is the Merkle–Damgård scheme, which builds hash functions from a compression function. Rivest introduced, in 1990, the first scheme of this type named MD4. Soon afterwads, Rivest himself proposed an improved version of MD4 called MD5.

Collisions in the compression function of MD4 and MD5 were discovered in the 1990s.

The family of secured hash algorithms (SHAs) was proposed as a standard by NIST in 1993. SHA-0 was adopted as FIPS 180. Each of these hash algorithms was an improvement of the earlier versions to prevent previously found attacks.

It was shown in 1998 that SHA-0 allows collisions in approximately 2^{61} steps, whereas the first collision for SHA-1 was found in 2017. SHA-2 includes the four hash functions known as SHA-224, SHA-256, SHA-384, and SHA-512, according to the sizes of the corresponding fingerprints. It is worth pointing out that currently SHA-256, which outputs 256 bits fingerprints, is the most used hash function. It is the basis of many password authentication systems.

According to the National Academy of Sciences, Engineering, and Medicine [8] (NAE), although, nowadays it is believed to be essentially impossible to break a hash function such

as SHA-256, password hashing is at higher risk due to the size of all 10-character passwords being only about 2^{66} passwords, and thus prone to an attack based on a quantum computer.

Possible attacks on the currently used hash functions have encouraged the use of provably secure hash functions, whose security is based on the difficulty of solving a known hard problem.

Cayley hash functions based on the Cayley graph of certain (semi)groups are examples of these types of schemes, whose security follows from the hardness of the expander graph problem associated with a (semi)group.

In 1991, Zémor [9] introduced hash function schemes based on matrix products in the special linear group $\mathrm{Sl}(\mathbb{F}_p)$, where p is a fixed prime number. Zémor himself and Tillich [10] broke such schemes. Furthermore, they introduced the group $\mathrm{Sl}(\mathbb{F}_{2^n})$ to increase the security of the original hash functions [11]. In this setting, $\mathbb{F}_{2^n}$ is a field.

Due to the popularity of the hash functions introduced by Zémor and Tillich, several proposals in the same line were proposed by Petit and Lubotzki et al., who introduced Cayley hash functions based on Ramanujan graphs, in particular LPS hash functions [12–15].

The Tillich–Zémor hash function hashes each bit of a given message individually. In this case, the matrices have the form $A = \begin{bmatrix} \alpha & 1 \\ 1 & 0 \end{bmatrix}$ and $B = \begin{bmatrix} \alpha & \alpha+1 \\ 1 & 1 \end{bmatrix}$, where $\mathbb{F}_{2^n} = F_2[x]/(p(x))$, F_2 is the two-element field, $(p(x))$ is the ideal generated by an irreducible polynomial of degree n, and α is a root of $p(x)$. For instance, the message $M = 110011$ is hashed to the matrix $B^2A^2B^2$.

It is worth noting that the Tillich–Zémor hash function was successfully attacked by Grass et al. [16], who obtained collisions using the Euclidean algorithm for polynomials. Afterwards, Petit and Quisquater [17] introduced an extended form of Grass et al.'s algorithm to provide a second preimage algorithm. Grassl et al. also ran Grover's algorithm on a quantum computer to study the strength of the cryptographic system AES [18].

Mullan and Tsaban [19] introduced a general attack for the Tillich–Zémor scheme. It runs with polynomial time $o(\sqrt{q})$ to find collisions for an arbitrary q. It does not work for bit strings of length $n > 100$.

Other pairs of matrices such as in the Tillich–Zémor scheme have been proposed by Bromber et al. and Sosnovki, who introduced a semigroup platform corresponding to the hash functions $f(x) = 2x + 1$ and $g(x) = 3x + 1$ modulo a prime $p > 3$ (the corresponding associated matrices have the form $A = \begin{bmatrix} 2 & 1 \\ 0 & 1 \end{bmatrix}$ and $B = \begin{bmatrix} 3 & 1 \\ 0 & 1 \end{bmatrix}$). In this case, the input string hash function can have an arbitrary length, and the output has the size $2log(p)$.

In this paper, we applied Sosnovski hash functions to Brauer messages to investigate their collision-resistant property.

2.1.1. Background and Related Work on Brauer Configuration Algebras

Brauer configuration algebras (BCAs) were introduced by Green and Schroll [4] to generalize research on tame algebras. Soon afterwards, Cañadas et al. used these algebras and their associated messages to obtain applications in cryptography, cibersecurity, and the graph energy theory [3,5,6,20,21].

It is worth pointing out that Espinosa [3] introduced in his doctoral dissertation the notion of the message of a Brauer configuration as the element of a word algebra. He used Brauer messages to give formulas for the number of perfect matchings of a snake graph and the number of homological ideals associated with a Nakayama algebra. On the other hand, Cañadas et al. introduced mutations of Brauer configurations to give an algebraic description of the cryptosystem AES [6].

In this paper, we associate Brauer configuration algebras with points in the plane, establishing which points have associated Brauer configuration algebras with the same dimension.

2.1.2. Path Algebras

In this section, we give a brief discussion on quivers, path algebras, and their ideals based on the work of Assem et al. [22].

A *quiver* or *directed graph* $Q = (Q_0, Q_1)$ is a quadruple consisting of two sets: Q_0 (whose elements are called *points* or *vertices*) and Q_1 (whose elements are called *arrows*) and two maps $s, t : Q_1 \to Q_0$ which associate to each arrow $\alpha \in Q_1$, its source $s(\alpha) \in Q_0$, and its target $t(\alpha) \in Q_0$, respectively. If $\mathbb{F}$ is an algebraically closed field, then we let $\mathbb{F}Q$ denote the path algebra associated with the quiver Q, whose underlying $\mathbb{F}$-vector space has as its basis the set of all paths of length $l \geq 0$ in Q, such that the product of two basis vectors is given by the usual concatenation of paths.

The following Figure 2 shows a quiver Q with four vertices a_1, a_2, a_3, and a_4 and three arrows α_1, α_2, and α_3. Note that, $Q_1 = \{\alpha_1, \alpha_2, \alpha_3\}$ is the set of paths of length 1, whereas $Q_2 = \{\alpha_1\alpha_2, \alpha_1\alpha_3\}$ is the set of paths of length 2 in Q.

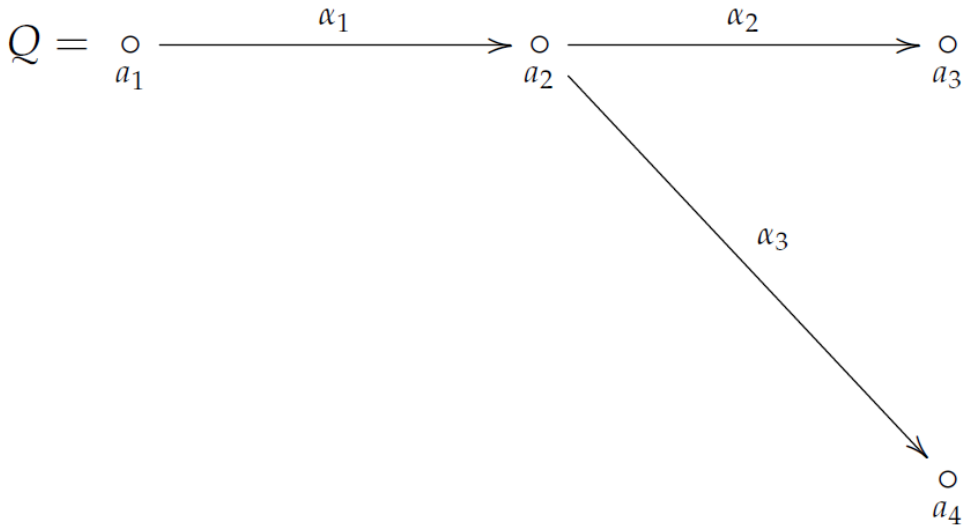

Figure 2. A quiver Q with four vertices and three arrows α_1, α_2, and α_3, $s(\alpha_2) = s(\alpha_3) = a_2 = t(\alpha_1)$, $s(\alpha_1) = a_1$, $t(\alpha_2) = a_3$, and $t(\alpha_3) = a_4$.

The basis $\mathcal{B}$ of the algebra $\mathbb{F}Q$ associated with the quiver Q shown in Figure 2 is $\mathcal{B} = \{e_{a_1}, e_{a_2}, e_{a_3}, e_{a_4}, \alpha_1, \alpha_2, \alpha_3, \alpha_1\alpha_2, \alpha_1\alpha_3\}$, where $\{e_{a_1}, e_{a_2}, e_{a_3}, e_{a_4}\}$ is the set of primitive idempotents, with $e_{a_i}^2 = e_{a_i}$ and $e_{a_i}e_{a_j} = 0$ if $i \neq j$.

Let Q be a finite and connected quiver. The *two-sided ideal* R_Q of the path algebra $\mathbb{F}Q$ generated (as an ideal) by the arrows of Q is called the arrow ideal of $\mathbb{F}Q$. A two-sided ideal I of $\mathbb{F}Q$ is said to be *admissible* if there exists $m \geq 2$ such that $R_Q^m \subseteq I \subseteq R_Q^2$.

If I is an admissible ideal of $\mathbb{F}Q$, the pair (Q, I) is said to be a *bound quiver*. The quotient algebra $\mathbb{F}Q/I$ is said to be the algebra of the bound quiver (Q, I) or, simply, a *bound quiver algebra*.

Let Q be a quiver. A *relation* in Q with coefficients in $\mathbb{F}$ is an $\mathbb{F}$-linear combination of paths of length, with at least two having the same source and target.

If $(\rho_j)_{j\in J}$ is a set of relations for a quiver Q such that the ideal they generate $\langle \rho_j \mid j \in J \rangle$ is admissible, we say that the quiver Q is bound by the relation $(\rho_j)_{j\in J}$ or by the relations $\rho_j = 0$ [22].

Henceforth, we let rad Λ denote the radical of a path algebra $\Lambda = \mathbb{F}Q$, which is the intersection of all maximal ideals. Actually, if I is an admissible ideal of Λ, it holds that $\text{rad}(\mathbb{F}Q/I) = R_Q/I$.

If $\prec$ is an *admissible well-ordering* on the set of paths, i.e., $\prec$ is a well-ordering such that

1. If $a, b, u, v \in Q$, where uav and ubv are both nonzero $a \prec uav$ or $a = uav$.
2. If $a \prec b$, then $uav \prec ubv$.

Then, the *tip* $Tip(x) = w$ of an element $x \in \mathbb{F}Q$ is the maximal path w with respect to $\prec$ such that w has a nonzero coefficient in x when it is written as a linear combination of the elements of a fixed basis of $\mathbb{F}Q$. $Tip(X) = \{Tip(x) \mid x \in X\}$ is the set of tips of elements in X [7].

Let I be an ideal in a path algebra $\mathbb{F}Q$ and let $\mathcal{G} \subset \mathcal{I}$. If $\langle Tip(\mathcal{G})\rangle = \langle Tip(I)\rangle$, then $\mathcal{G}$ is a *Gröbner basis* for I with respect to $\prec$.

2.1.3. Brauer Configuration Algebras

In this section, we briefly discuss the main results regarding Brauer configuration algebras [4].

A Brauer configuration algebra $\Lambda_{\mathfrak{B}}$ (or simply Λ if no confusion arises) is induced by a Brauer configuration $\mathfrak{B} = (\mathfrak{B}_0, \mathfrak{B}_1, \mu, \mathcal{O})$, consisting of a pair of finite sets $\mathfrak{B}_0$ and $\mathfrak{B}_1$, a function $\mu : \mathfrak{B}_0 \to \mathbb{N}^+$ ($\mathbb{N}^+$ denote the set of positive integers), and an orientation $\mathcal{O}$.

- Elements of $\mathfrak{B}_0$ ($\mathfrak{B}_1$) are called *vertices* (*polygons*). Polygons are labeled multisets consisting of vertices.
- If $V \in \mathfrak{B}_1$, then $|V| > 1$ (i.e., each polygon contains more than one vertex).
- $\mathcal{O}$ is a choice for each vertex $\delta \in \mathfrak{B}_0$ of a cyclic ordering of the polygons in which δ occurs as a vertex including repetitions (see [4] for more details). For instance, if a vertex $\delta \in \mathfrak{B}_0$ occurs in polygons $V_{i_i}, V_{i_2}, \ldots, V_{i_m}$ for suitable indices, then the cyclic order is obtained by linearly ordering the list, say

$$V_{i_1}^{\mathfrak{d}_1} < V_{i_2}^{\mathfrak{d}_2} < \cdots < V_{i_m}^{\mathfrak{d}_m}, \quad \mathfrak{d}_{i_s} > 0, \tag{1}$$

  where, $V_{i_s}^{\mathfrak{d}_s} = V_{i_s}^{(1)} < V_{i_s}^{(2)} < \cdots < V_{i_s}^{(\mathfrak{d}_s)}$ means that vertex δ occurs $\mathfrak{d}_s$ times in polygon V_{i_s}, denoted $\mathfrak{d}_s = occ(\delta, V_{i_s})$. The cyclic order is completed by adding the relation $V_{i_m} < V_{i_1}$. Note that if $V_{i_1} < \cdots < V_{i_t}$ is the chosen ordering at vertex δ, then the same ordering can be represented by any cyclic permutation.
  The sequence (1) is said to be the successor sequence at vertex δ, denoted S_δ, which is unique up to permutations. Note that Green and Schroll [4] mentioned that different orientation choices are typically associated to nonisomorphic Brauer configuration algebras.
  Henceforth, in this paper, if a vertex $\delta' \neq \delta$ belongs to some polygons $V_{j_1}, V_{j_2}, \ldots V_{j_k}$ ordered according to the already defined cyclic ordering associated with the vertex δ, then we assume that up to permutations the cyclic ordering associated with the vertex δ' is built, taking into account that polygons $V_{j_1}, V_{j_2}, \ldots V_{j_k}$ inherit the order given by the successor sequence S_δ.
- If $\overline{V}$ denotes the underlying set defined by a polygon V (repetitions are not allowed in $\overline{V}$), then $\bigcup_{V \in \mathfrak{B}_1} \overline{V} = \mathfrak{B}_0$.

  If $\delta \in \mathfrak{B}_0$, then the valency $val(\delta)$ of δ is given by the identity

$$val(\delta) = \sum_{V \in \mathfrak{B}_1} occ(\delta, V). \tag{2}$$

If $\delta \in \mathfrak{B}_0$ is such that $\mu(\delta)val(\delta) = 1$, then δ is said to be *truncated* (it occurs once in just one polygon). Otherwise, δ is a nontruncated vertex. A Brauer configuration without truncated vertices is said to be *reduced*.

Algorithm 1 is a short version of an algorithm defined by Cañadas et al. in [5] to build a Brauer configuration algebra.

The following results describe the structure of Brauer configuration algebras [4,23].

Algorithm 1: Construction of a Brauer configuration algebra

1. **Input** A reduced Brauer configuration $\mathfrak{B} = (\mathfrak{B}_0, \mathfrak{B}_1, \mu, \mathcal{O})$.
2. **Output** The Brauer configuration algebra $\Lambda_{\mathfrak{B}} = \mathbb{F}Q_{\mathfrak{B}}/I_{\mathfrak{B}}$.
3. Construct the quiver $Q_{\mathfrak{B}} = ((Q_{\mathfrak{B}})_0, (Q_{\mathfrak{B}})_1, s, t)$ as follows:
   - $(Q_{\mathfrak{B}})_0 = \mathfrak{B}_1$.
   - For each covering $V_i < V_j$ in a successor sequence S_δ, define an arrow $V_i \overset{\alpha_i^\delta}{\to} V_j \in (Q_{\mathfrak{B}})_1$.
   - For each path defined by a successor sequence S_δ, construct a special cycle $C_\delta \in Q_{\mathfrak{B}}$ defined by the union $S_\delta \cup \{V_{i_m} < V_{i_1}\}$, where, $V_{i_1} = \min S_\delta$ and $V_{i_m} = \max S_\delta$.
4. Define the path algebra $\mathbb{F}Q_{\mathfrak{B}}$.
5. The admissible ideal $I_{\mathfrak{B}}$ is generated by relations of the following types:
   (a) Identify special cycles associated with nontruncated vertices in the same polygon (i.e., if $\delta_1, \delta_2 \in U$ with $U \in \mathfrak{B}_1$, then $C_{\delta_1}^{\mu(\delta_1)} \sim C_{\delta_2}^{\mu(\delta_2)}$).
   (b) If C_δ is a special cycle associated with a nontruncated vertex δ, then a product of the form $C^{\mu(\delta)}a \in I_{\mathfrak{B}}$ if a is the first arrow of δ.
   (c) Products of the form $ab \in \mathbb{F}Q_{\mathfrak{B}}$ with a induced by a covering in a special cycle C_{δ_1} and b induced by another special cycle C_{δ_2} (with $\delta_1 \neq \delta_2$) belong to $I_{\mathfrak{B}}$.
6. $\Lambda_{\mathfrak{B}} = \mathbb{F}Q_{\mathfrak{B}}/I_{\mathfrak{B}}$ is a Brauer configuration algebra with a basis consisting of classes of special cycles and classes of prefixes of special cycles.

Theorem 1 ([4], Theorem B, Proposition 2.7, Theorem 3.10, Corollary 3.12)**.** *Let $\Lambda_{\mathfrak{B}}$ be a Brauer configuration algebra with Brauer configuration $\mathfrak{B} = (\mathfrak{B}_0, \mathfrak{B}_1, \mu, \mathcal{O})$.*

1. *There is a bijection between the set of indecomposable projective $\Lambda_{\mathfrak{B}}$-modules and the polygons of $\mathfrak{B}$. Moreover, if P_V is an indecomposable $\Lambda_{\mathfrak{B}}$-module induced by a polygon V with r nontruncated vertices, then* $\operatorname{rad} P_V = \sum_{i=1}^{r} U_i$*, where for each i, U_i is a uniserial $\Lambda_{\mathfrak{B}}$-module. $U_i \cap U_j$ is uniserial for all $1 \leq i, j \leq r$.*
2. *I is admissible, and $\Lambda_{\mathfrak{B}}$ is multiserial and symmetric.*
3. *The number of summands in the* $\operatorname{rad} P_V / \operatorname{soc} P_V$ *of an indecomposable projective $\Lambda_{\mathfrak{B}}$-module P_V with* $\operatorname{rad}^2 P_V \neq 0$ *equals the number of nontruncated vertices of the polygon V counting repetitions.*
4. *If $V \in \mathfrak{B}_1$, $|V| \geq 3$, $\delta \in V$ is a truncated vertex, $\mathfrak{B}' = (\mathfrak{B}'_0, \mathfrak{B}'_1, \mu, \mathcal{O})$, $\mathfrak{B}'_0 = \mathfrak{B}_0 \backslash \{\delta\}$, $\mathfrak{B}'_1 = \mathfrak{B}_1 \backslash V \cup V'$ with $V' = V \backslash \{\delta\}$, then $\Lambda_{\mathfrak{B}}$ and $\Lambda_{\mathfrak{B}'}$ are isomorphic.*

Proposition 1 and Theorem 2 give formulas for the dimensions $\dim_{\mathbb{F}} \Lambda_{\mathfrak{B}}$ and $\dim_{\mathbb{F}} Z(\Lambda_{\mathfrak{B}})$ of a Brauer configuration algebra Λ and its center $Z(\Lambda_{\mathfrak{B}})$ [4,23].

Proposition 1 ([4], Proposition 3.13)**.** *Let $\Lambda_{\mathfrak{B}}$ be a Brauer configuration algebra associated with the Brauer configuration $\mathfrak{B}$ and let $\mathcal{C} = \{C_1, \ldots, C_t\}$ be a full set of equivalence class representatives of special cycles. Assume that for $i = 1, \ldots, t$, C_i is a special δ_i-cycle where δ_i is a nontruncated vertex in $\mathfrak{B}$. Then,*

$$\dim_{\mathbb{F}} \Lambda_{\mathfrak{B}} = 2|Q_0| + \sum_{C_i \in \mathcal{C}} |C_i|(n_i|C_i| - 1) = \sum_{\substack{\delta_i \in \Gamma_0 \\ \delta_i \text{ non-truncated}}} val(\delta_i)(n_i val(\delta_i) - 1),$$

where $|Q_0|$ denotes the number of vertices of Q, $|C_i|$ denotes the number of arrows in the δ_i-cycle C_i, and $n_i = \mu(\delta_i)$.

Theorem 2 (Theorem 4.9, [23]). *Let $\Lambda_{\mathfrak{B}} = \mathbb{F}Q/I$ be the Brauer configuration algebra associated with the connected and reduced Brauer configuration $\mathfrak{B}$. Then,*

$$\dim_{\mathbb{F}} Z(\Lambda_{\mathfrak{B}}) = 1 + \sum_{\delta \in \mathfrak{B}_0} \mu(\delta) + |\mathfrak{B}_1| - |\mathfrak{B}_0| + \#(Loops\ Q) - |\mathcal{C}_{\mathfrak{B}}|,$$

where $\mathcal{C}_{\mathfrak{B}} = \{\delta \in \Gamma_0 \mid val(\delta) = 1,\ and\ \mu(\delta) > 1\}$.

Proposition 2 ([4], Proposition 3.6). *Let $\Lambda_{\mathfrak{B}}$ be the Brauer configuration algebra associated with a connected Brauer configuration $\mathfrak{B}$. The algebra $\Lambda_{\mathfrak{B}}$ has a length grading induced from the path algebra $\mathbb{F}Q$ if and only if there is an $N \in \mathbb{Z}_{>0}$ such that for each nontruncated vertex $\delta \in \mathfrak{B}_0$ $val(\delta)\mu(\delta) = N$.*

2.2. The Message of a Brauer Configuration

The notion of the message of a Brauer configuration and labeled Brauer configurations were introduced by Espinosa et al. [3] to define suitable specializations of some Brauer configuration algebras. According to them, since polygons in a Brauer configuration $\mathfrak{B} = (\mathfrak{B}_0, \mathfrak{B}_1, \mu, \mathcal{O})$ are multisets, it is possible to assume that any polygon $U \in \mathfrak{B}_1$ is given by a word $w(U)$ of the form

$$w(U) = \delta_1^{s_1}\delta_2^{s_2}\dots\delta_{t-1}^{s_{t-1}}\delta_t^{s_t}, \tag{3}$$

where for each i, $1 \leq i \leq t$, $s_i = \mathrm{occ}(\delta_i, U)$.

The message is in fact an algebra of words element $\mathcal{W}_{\mathfrak{B}}$ associated with a fixed Brauer configuration, such that for a given field $\mathbb{F}$ the word algebra $\mathcal{W}_{\mathfrak{B}}$ consists of formal sums of words with the form $\sum_{\substack{\alpha_i \in \mathbb{F} \\ U \in \mathfrak{B}_1}} \alpha_i w(U)$, $0w(U) = \varepsilon$ is the empty word, and $1w(U) = w(U)$ for any $U \in \mathfrak{B}_1$. The product in this case is given by the usual word concatenation. The formal product (or word product)

$$M(\mathfrak{B}) = \prod_{U \in \mathfrak{B}_1} w(U) \tag{4}$$

is said to be the *message of the Brauer configuration* $\mathfrak{B}$.

If $\mathfrak{R}$ is a ring, then a *specialization* of a reduced Brauer configuration $\mathfrak{B} = (\mathfrak{B}_0, \mathfrak{B}_1, \mu, \mathcal{O})$ is a Brauer configuration

$$\mathfrak{B}^{\mathfrak{e}} = (\mathfrak{B}_0^{\mathfrak{e}}, \mathfrak{B}_1^{\mathfrak{e}}, \mu^{\mathfrak{e}}, \mathcal{O}^{\mathfrak{e}})$$

endowed with a suitable map $\mathfrak{e} : \mathfrak{B}_0 \to \mathfrak{R}$, such that

$$\begin{aligned}
\mathfrak{B}_0^{\mathfrak{e}} &= \mathrm{Img}\ e \subset \mathfrak{R},\\
\mathfrak{B}_1^{\mathfrak{e}} &= \mathfrak{e}(\mathfrak{B}_1) = \{\mathfrak{e}(H) \mid H \in \mathfrak{B}_1\}, \quad \text{if } H \in \mathfrak{B}_1 \text{ then } \mathfrak{e}(H) = \{\mathfrak{e}(\delta_i) \mid \delta_i \in H\} \in \mathfrak{e}(\mathfrak{B}_1),\\
w^{\mathfrak{e}}(U) &= ((\mathfrak{e}(\delta_1))^{f_1}(\mathfrak{e}(\delta_2))^{f_2}\dots(\mathfrak{e}(\delta_n))^{f_n}) \text{is the specialization under } \mathfrak{e} \text{ of a word}\\
w(U) &= \delta_1^{f_1}\delta_2^{f_2}\dots\delta_n^{f_n} \text{ associated with a polygon } U \in \mathfrak{B}_1.\\
\mu^{\mathfrak{e}}(\mathfrak{e}(\delta)) &= \mu(\delta), \text{ for any vertex } \delta \in \mathfrak{B}_0.
\end{aligned} \tag{5}$$

The orientation $\mathcal{O}^{\mathfrak{e}}$ is defined by the orientation $\mathcal{O}$, in such a way that if

$$S_\delta = U_{i_1}^{\mathfrak{d}_1} < U_{i_2}^{\mathfrak{d}_2} < \dots < U_{i_m}^{\mathfrak{d}_m}$$

is a successor sequence associated with a vertex $\delta \in \mathfrak{B}_0$ (see (1)), then, for some $\mathfrak{d}'_j > 0$, $1 \leq j \leq m$, it holds that

$$S'_{\mathfrak{e}(\delta)} = (\mathfrak{e}(U_{i_1}))^{\mathfrak{d}'_1} < (\mathfrak{e}(U_{i_2}))^{\mathfrak{d}'_2} < \dots < (\mathfrak{e}(U_{i_m}))^{\mathfrak{d}'_m}$$

is contained in the successor sequence $S_{\mathfrak{e}(\delta)}$ associated with $\mathfrak{e}(\delta) \in \mathfrak{B}_0^{\mathfrak{e}}$.

$M(\mathfrak{B}^{\mathfrak{e}}) = \underset{U \in \mathfrak{B}_1^{\mathfrak{e}}}{\propto} w^{\mathfrak{e}}(U)$ ($\propto$ is a suitable operation associated with the specialization ring) is the *specialized message* of the Brauer configuration $\mathfrak{B}$ provided that, in such a case, each word can be interpreted by a product of the specialization ring elements.

A Brauer configuration $\mathfrak{B} = (\mathfrak{B}_0, \mathfrak{B}_1, \mu, \mathcal{O})$ is said to be *S-labeled* (or simply labeled, if no confusion arises) by an integer sequence $S = \{n_1, n_2, \ldots, n_{|\Gamma_1|}\}$ if each polygon U_{i_j} is labeled by an integer number n_j, $1 \leq j \leq |\mathfrak{B}_1|$. In such a case, we often write $\mathfrak{B}_1 = \{(U_1, n_1), (U_2, n_2), \ldots, (U_k, n_k)\}$,

For each vertex $\delta \in \mathfrak{B}_0$, a corresponding cyclic ordering of labeled polygons where δ occurs is defined. One advantage of labeling Brauer configurations is that the set S can be used to systematically define the orientation associated with each vertex or obtain the polygons recursively [3].

It is worth noticing that any finite set can be used to label Brauer configurations. In this paper, we use finite well-ordered sets of lattice paths to label Brauer configurations.

3. Main Results

In this section, we give properties of Brauer configuration algebras $\Lambda_{\mathfrak{B}(x,y)}$ associated with points $(x, y) \in \mathcal{V}$.

If $(x, y) \in \{(r, s) \in \mathbb{R}^2 \mid r > 0, s > 0\} = \mathcal{D}$, then (x, y) defines a path $\mathcal{P}$ connecting $P_0 = (x, y)$ with $(0, 0)$. Internal vertices P_i are such that;

$$\begin{aligned} &P_i \in \{(\lfloor (x) \rfloor - (i-2), y), (x, \lfloor (y) \rfloor - (i-2))\}, \quad \text{for} \quad 2 \leq i \leq xy - 1. \\ &P_1 \in \{(\lfloor (x) \rfloor, y), (x, \lfloor (y) \rfloor)\}. \end{aligned} \tag{6}$$

An arrow $\alpha \in \mathcal{P}$ connecting vertices $(\lfloor (x) \rfloor - i, y), (\lfloor (x) \rfloor - (i+1), y)$ $((\lfloor (x) \rfloor, y - i), (\lfloor (x) \rfloor, y - (i+1)))$ is labeled by a symbol x (y). The same is assumed if the arrow α connects points $(\lfloor (x) \rfloor, y)$ and $(\lfloor (x) \rfloor, \lfloor (y) \rfloor)$ and $((x, \lfloor (y) \rfloor)$ and $(\lfloor (x) \rfloor, \lfloor (y) \rfloor))$. Since, it is easy to see that there are $\binom{\lfloor (x) \rfloor + \lfloor (y) \rfloor}{\lfloor (x) \rfloor} = c(x, y)$ of such paths, we endowed this set with a linear order $\preceq$. We let $\mathcal{L}^{(x,y)}$ denote such a set of paths, whose elements $\mathcal{L}_1^{(x,y)}, \mathcal{L}_2^{(x,y)}, \ldots, \mathcal{L}_{c(x,y)}^{(x,y)}$ are ordered in the form:

$$\mathcal{L}_1^{(x,y)} \prec \mathcal{L}_2^{(x,y)} \prec \cdots \prec \mathcal{L}_{c(x,y)}^{(x,y)}. \tag{7}$$

A labeled (by the set $\mathcal{L}^{(x,y)}$) Brauer configuration $\mathfrak{B}(x, y)$ is associated with each point $(r, s) \in \mathcal{D}$ in such a way that:

$$\begin{aligned} \mathfrak{B}^{(r,s)} &= (\mathfrak{B}_0^{(r,s)}, \mathfrak{B}_1^{(r,s)}, \mu^{(r,s)}, \mathcal{O}^{(r,s)}), \\ \mathfrak{B}_0^{(r,s)} &= \{x, y\}, \\ \mathfrak{B}_1^{(r,s)} &= \{(P_1, \mathcal{L}_1^{(r,s)}), (P_2, \mathcal{L}_2^{(r,s)}), \ldots, (P_{c(r,s)}, \mathcal{L}_{c(r,s)}^{(r,s)})\}, \\ \mu^{(r,s)}(x) &= \mu^{(r,s)}(y) = 1, \\ (P_i, \mathcal{L}_i^{(r,s)}) &< (P_{i+1}, \mathcal{L}_{i+1}^{(r,s)}) \quad \text{if and only if} \quad \mathcal{L}_i^{(r,s)} \prec \mathcal{L}_{i+1}^{(r,s)}, \quad 1 \leq i \leq c(r,s) - 1, \\ w(P_i) &= x^r y^s \quad \text{is the word associated with each polygon } P_i. \end{aligned} \tag{8}$$

Successor sequences S_x and S_y have the forms:

$$\begin{aligned} S_x = \ldots &(P_i, \mathcal{L}_1^{(r,s)})^{(1)} < (P_i, \mathcal{L}_2^{(r,s)})^{(2)} < \cdots < (P_i, \mathcal{L}_r^{(r,s)})^{(r)} \ldots \\ &< (P_{c(r,s)}, \mathcal{L}_1^{(r,s)})^{(1)} < \cdots < (P_{c(r,s)}, \mathcal{L}_r^{(r,s)})^{(r)}, \\ S_y = \ldots &(P_i, \mathcal{L}_1^{(r,s)})^{(1)} < (P_i, \mathcal{L}_2^{(r,s)})^{(2)} < \cdots < (P_i, \mathcal{L}_s^{(r,s)})^{(s)} \ldots \\ &< (P_{c(r,s)}, \mathcal{L}_1^{(r,s)})^{(1)} < \cdots < (P_{c(r,s)}, \mathcal{L}_s^{(r,s)})^{(s)}. \end{aligned} \tag{9}$$

The following Figure 3 shows a Brauer quiver $Q_{\mathfrak{B}^{(r,s)}}$ induced by a Brauer configuration $\mathfrak{B}^{(r,s)}$, with $\lceil (r) \rceil + \lceil (s) \rceil = 4$.

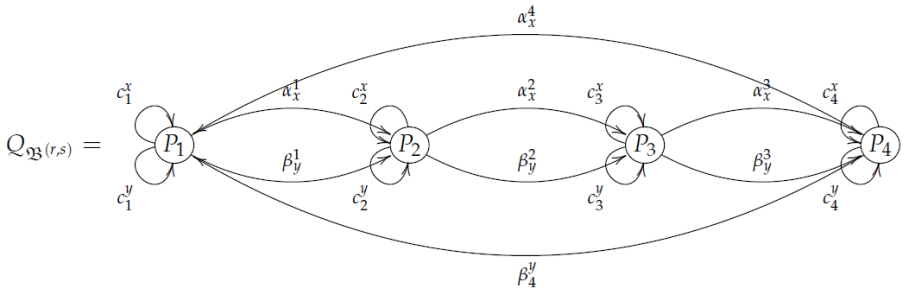

Figure 3. Example of a Brauer quiver associated with a Brauer configuration $\mathfrak{B}^{(r,s)}$. The symbol c_i^x (c_j^y) denotes a set of loops $c_{i,k}^x$ ($c_{j,m}^y$) associated with the vertex x (y), $1 \leq k \leq r$, $1 \leq m \leq s$.

For all possible values of i, j, and k, the admissible ideal $I_{\mathfrak{B}^{(r,s)}}$ is generated by relations of the form:

- $\alpha_x^i \alpha_x^{i+1}$, $\alpha_x^i \beta_y^{i+1}$, $\beta_y^i \beta_y^{i+1}$,
- $\alpha_x^i c_{i+1,k}^y$, $c_{j,k}^y \alpha_x^j$, $\beta_y^j c_{j+1,k}^x$,
- $(c_{i,j}^x)^2$, $(c_{i,j}^y)^2$.
- $C_x \sim C_y$, $C_r a$, where C_x (C_y) is a special cycle associated with vertex x (y), and a is the first arrow of a special cycle C_r, $r \in \{x, y\}$.

The following result gives the structure of Brauer configuration algebras $\Lambda_{\mathfrak{B}^{(r,s)}} = \mathbb{F}Q_{\mathfrak{B}^{(r,s)}} / I_{\mathfrak{B}^{(r,s)}}$ induced by Brauer configurations of type $\mathfrak{B}^{(r,s)}$.

Theorem 3. *For $r, s \in \mathcal{D}$, it holds that:*

1. *The Brauer configuration algebra $\Lambda_{\mathfrak{B}^{(r,s)}}$ is reduced and connected.*
2. *$\Lambda_{\mathfrak{B}(r,r)}$ has length grading induced by the path algebra $\mathbb{F}Q_{\mathfrak{B}(r,r)}$.*
3. *The number of summands in the heart $ht(P_i)$ of the indecomposable projective module P_i equals $r + s$, and for any i, $1 \leq i \leq r + s$.*
4. $\dim_{\mathbb{F}} \Lambda_{\mathfrak{B}^{(r,s)}} = (r^2 + s^2)c(\lceil r \rceil, \lceil (s) \rceil) - c(\lceil r \rceil, \lceil (s) \rceil)(r + s - 2)$.
5. *If $(x, y) \in (\lfloor (r) \rfloor, \lceil (r) \rceil] \times (\lfloor (s) \rfloor, \lceil (s) \rceil]$, then* $\dim_{\mathbb{F}} \Lambda_{\mathfrak{B}(x,y)} = \dim_{\mathbb{F}} \Lambda_{\mathfrak{B}(\lceil r \rceil, \lceil (s) \rceil)}$.
6. $\dim_{\mathbb{F}} Z(\Lambda_{\mathfrak{B}(\lceil r \rceil, \lceil (s) \rceil)} = 1 + (\lceil (r) \rceil + \lceil (s) \rceil - 1)c(\lceil r \rceil, \lceil (s) \rceil)$.

Proof.

1. Since each polygon contains vertices x and y, it follows that the Brauer configuration $\mathfrak{B}^{(r,s)}$ is reduced. Furthermore, $\bigcap_{i=1}^{c(\lceil (r) \rceil, \lceil (s) \rceil)} P_i \neq \varnothing$. Thus, $Q_{\mathfrak{B}^{(r,s)}}$ is connected.
2. Since $\mu^{(r,r)} val(x) = \mu^{(r,r)} val(y) = r\binom{2r}{r}$, the result holds as a direct consequence of Proposition 2.
3. Each polygon P_i has r vertices denoted by x and s vertices denoted by y. Thus, P_i has $r + s$ nontruncated vertices, counting repetitions.
4. $val(x) = rc(\lceil (r) \rceil, \lceil (s) \rceil)$, $val(y) = sc(\lceil (r) \rceil, \lceil (s) \rceil)$, and $|\mathfrak{B}_1^{(r,s)}| = c(\lceil (r) \rceil, \lceil (s) \rceil)$.
5. Since $\text{rad}^2 \Lambda_{\mathfrak{B}^{(r,s)}} \neq 0$, it suffices to observe that

$$\#(Loops\ Q_{\mathfrak{B}^{(r,s)}}) = (s-1)c(\lceil (r) \rceil, \lceil (s) \rceil) + (r-1)c(\lceil (r) \rceil, \lceil (s) \rceil).$$

6. We note that by definition $\mathcal{L}^{(r,s)} = \mathcal{L}^{(x,y)}$, if $(x, y) \in (\lfloor (r) \rfloor, \lceil (r) \rceil] \times (\lfloor (s) \rfloor, \lceil (s) \rceil]$.

□

Remark 1. *Similar results as those presented in Theorem 3 can be obtained in the other quadrants of $\mathcal{V} = \mathbb{R}^* \times \mathbb{R}^*$ by building sets $\mathcal{L}^{(r,s)}$ connecting consecutive vertices in an appropriated fashion.*

For instance, in the fourth quadrant, arrows in paths connect vertices $(\lfloor(r)\rfloor, \lfloor(s)\rfloor)$ *and* $(0,0)$ *with consecutive internal vertices of the form* $(\lceil(r)\rceil - i, y), (\lceil(r)\rceil - (i+1), y)$ *or of the form* $(x, \lceil(s)\rceil - i), (x, \lceil(s)\rceil - (i+1))$.

Mutation and Frozen Regions in the Plane

Let $f^2 \subset \mathcal{D}$ be a line segment containing points $(n,k), (x_1,0), (0,x_1) \in (\mathbb{Z}\times\mathbb{Z}) \cap \mathcal{D}$, with $x_1 = n+k$. Let f^1 be a differentiable increasingly monotone curve on the interval $[0, x_0]$ with $f^1(x_0) = 0$, and $f^1(0) = y_0$, x_0, and y_0 are real numbers such that $0 \leq x_0 \leq x_1$, and $0 \leq y_0 \leq x_1$, thus, $0 \leq f^1 \leq f^2$. These inequalities define a region $R_{f_r^1}$ (R_m) bounded by the coordinate axes f^1 and f^2 (bounded by the coordinate axes and f^1). $R_{f_r^1}$ is said to be a *frozen* (*mutation*) region. Figure 4 shows regions $R_{f_r^1}$ and R_m. We let $\mathcal{R}(f^1, f^2)$ denote the set $\{(x,y) \in \mathcal{D} \mid 0 \leq x \leq x_1, 0 \leq y \leq f^2(x)\}$.

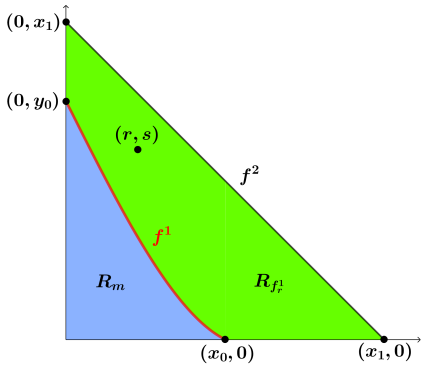

Figure 4. Frozen $(R_{f_r^1})$ and mutation (R_m) regions.

Mutation and frozen regions allow defining a new labeling for sets $\mathcal{L}^{(x,y)}$ of paths associated with Brauer configurations of type (8). Such a labeling is defined as follows:

- Arrows $\alpha = a \to b \in \mathcal{L}^{(x,y)}$ are labeled with symbols x (horizontal arrows), y (vertical arrows), and z (y mutations).
- An arrow $\alpha = a \xrightarrow{y} b \in \mathcal{L}^{(x,y)}$ is labeled with a new symbol z, i.e., $\alpha = a \xrightarrow{z} b$, if the target $b \in \text{Int } R_m \cup \text{bd}(R_m) \backslash \{f\}$. Where, Int X ($bd(X)$) denotes the interior (boundary) of a set X endowed with the usual topology of $\mathbb{R}^2$.
- The labeling of an arrow $\alpha = a \xrightarrow{y} b \in \mathcal{L}^{(x,y)}$ is kept without changes if $b \notin \text{Int } R_m$.

Figure 5 shows an example of this kind of labeling with $f^1(x) = -\frac{2}{3}x + 2$, $0 \leq x \leq 3$, and $f^2(x) = \sqrt{\frac{49}{4} - x^2}$, $0 \leq x \leq \frac{7}{2}$.

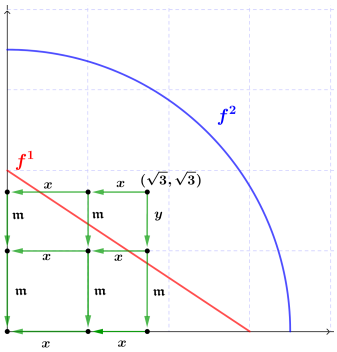

Figure 5. Example of a labeling with mutations associated with a region $\mathcal{R}(-\frac{2}{3}x + 2\ (0 \leq x \leq 3), \sqrt{\frac{49}{4} - x^2}\ (0 \leq x \leq \frac{7}{2}))$.

The Brauer configuration $\mathfrak{B}((\sqrt{3},\sqrt{3}),f^1,f^2)=(\mathfrak{B}_0^{(\sqrt{3},\sqrt{3})},\mathfrak{B}_1^{(\sqrt{3},\sqrt{3})},\mu^{(\sqrt{3},\sqrt{3})},\mathcal{O}^{(\sqrt{3},\sqrt{3})})$ is defined as follows:

- $\mathfrak{B}_0^{(\sqrt{3},\sqrt{3})}=\{x,y,\mathfrak{m}\}$.
- $|\mathfrak{B}_1^{(\sqrt{3},\sqrt{3})}|=|Q_0|=\binom{4}{2}=6$.
- $\mathfrak{B}_1^{(\sqrt{3},\sqrt{3})}=\left\{(U_i^{(\sqrt{3},\sqrt{3})},\mathcal{L}_i^{(\sqrt{3},\sqrt{3})})\mid 1\leq i\leq 6\right\}$.
- $\mathcal{L}_1^{(\sqrt{3},\sqrt{3})}=x\mathfrak{m}\mathfrak{m}x,\quad \mathcal{L}_2^{(\sqrt{3},\sqrt{3})}=x\mathfrak{m}x\mathfrak{m},\quad \mathcal{L}_3^{(\sqrt{3},\sqrt{3})}=xx\mathfrak{m}\mathfrak{m},$
  $\mathcal{L}_4^{(\sqrt{3},\sqrt{3})}=yxx\mathfrak{m},\quad \mathcal{L}_5^{(\sqrt{3},\sqrt{3})}=yx\mathfrak{m}x,\quad \mathcal{L}_6^{(\sqrt{3},\sqrt{3})}=y\mathfrak{m}xx.$
- $\mu^{(\sqrt{3},\sqrt{3})}(x)=\mu^{(\sqrt{3},\sqrt{3})}(y)=\mu^{(\sqrt{3},\sqrt{3})}(\mathfrak{m})=1$.
- The successor sequences S_x, S_y, and $S_\mathfrak{m}$ at vertices x, y, and $\mathfrak{m}$ have the following forms:

$$\begin{aligned}
S_x &= (U_1^{(\sqrt{3},\sqrt{3})})^{(2)}<(U_2^{(\sqrt{3},\sqrt{3})})^{(2)}<(U_3^{(\sqrt{3},\sqrt{3})})^{(2)}<(U_4^{(\sqrt{3},\sqrt{3})})^{(2)}\\
&\quad<(U_5^{(\sqrt{3},\sqrt{3})})^{(2)}<(U_6^{(\sqrt{3},\sqrt{3})})^{(2)}.\\
S_y &= U_4^{(\sqrt{3},\sqrt{3})}<U_5^{(\sqrt{3},\sqrt{3})}<U_6^{(\sqrt{3},\sqrt{3})}.\\
S_\mathfrak{m} &= (U_1^{(\sqrt{3},\sqrt{3})})^{(2)}<(U_2^{(\sqrt{3},\sqrt{3})})^{(2)}<(U_3^{(\sqrt{3},\sqrt{3})})^{(2)}<U_4^{(\sqrt{3},\sqrt{3})}<U_5^{(\sqrt{3},\sqrt{3})}<U_6^{(\sqrt{3},\sqrt{3})}.
\end{aligned}\tag{10}$$

- $val(x)=12,\quad val(y)=3,\quad val(\mathfrak{m})=9$.
- $\dim_\mathbb{F}\Lambda_{\mathfrak{B}(\sqrt{3},\sqrt{3})}=212$.
- $\dim_\mathbb{F}Z(\Lambda_{\mathfrak{B}(\sqrt{3},\sqrt{3})})=16$.

Figure 6 illustrates polygons, $U_1,\ldots,U_6$.

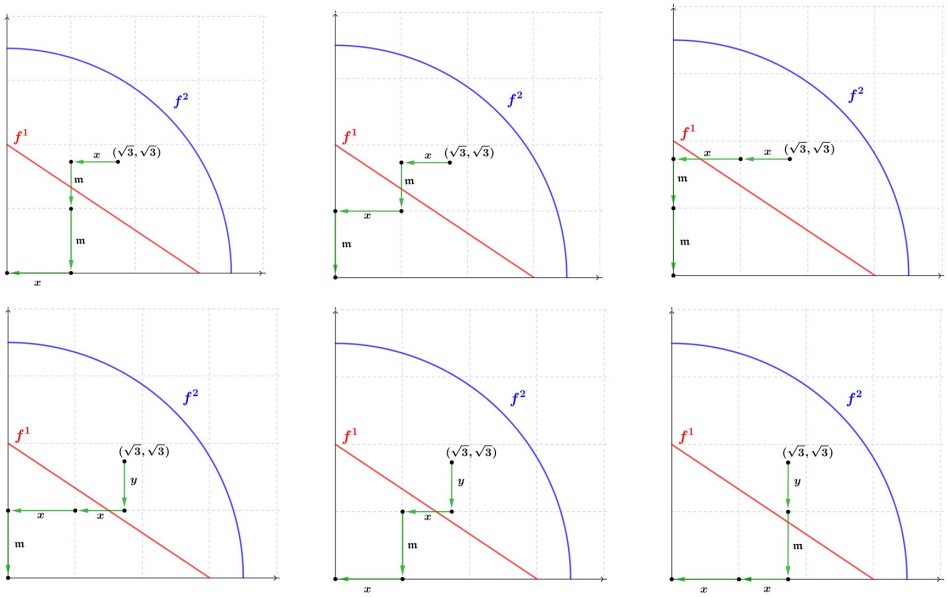

Figure 6. Labeled polygons $U_1,\ldots,U_6\in\mathfrak{B}_1^{(\sqrt{3},\sqrt{3})}$.

Figure 7 shows the Brauer quiver induced by the Brauer configuration $\mathfrak{B}((\sqrt{3},\sqrt{3}),f^1,f^2)$.

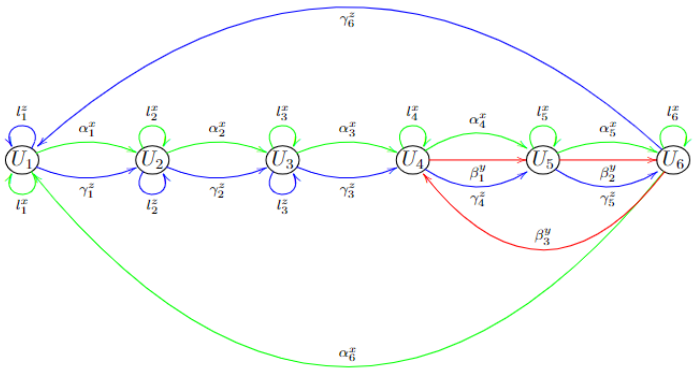

Figure 7. Brauer quiver $Q_{\mathfrak{B}((\sqrt{3},\sqrt{3}),f^1,f^2)}$ induced by the Brauer configuration $\mathfrak{B}((\sqrt{3},\sqrt{3}),f^1,f^2)$.

The admissible ideal is generated by the following relations (together with those related to special cycles) for all possible values of i, j, and h:

- $(l_i^x)^2$, $(l_i^y)^2$, and $(l_i^m)^2$.
- $l_i^a\, l_j^b$, if $a \neq b$.
- $\gamma_i^m\, l_j^x$, $l_j^x\, \gamma_i^m$.
- $\alpha_i^x\, l_j^m$, $l_j^m\, \alpha_i^x$.
- $\beta_j^y\, l_i^x$, $l_i^x\, \beta_j^y$.
- $\gamma_i^m\, \beta_j^y$, $\beta_j^y\, \gamma_i^m$.
- $\gamma_i^m\, \alpha_j^x$, $\alpha_j^x\, \gamma_i^m$.
- $\alpha_i^x\, \beta_j^y$, $\beta_j^y\, \alpha_i^x$.
- $s^x\alpha_h^x$, $s^y\beta_h^y$, and $s^m\gamma_h^m$.

Remark 2. *Henceforth, we consider the specialization* $\mathfrak{e} : \mathfrak{B}_0((x,y),f^1,f^2) \to \mathbb{C}$, *with* $\mathfrak{e}(x) = \mathfrak{e}(y) = 1$ *and* $e(\mathfrak{m}) = e^{i\phi}$ *for some fixed* ϕ, $0 \leq \phi \leq 2\pi$ *associated with the Brauer configuration* (8) *and its mutation. A specialized word is given by a product over* $\mathbb{C}$ *of the* x, y, *and* $\mathfrak{m}$ *specializations, whereas a message* $\mathfrak{M}(\mathfrak{B}((x,y),f^1,f^2)$ *of the Brauer configuration* $\mathfrak{B}((x,y),f^1,f^2)$ *is the sum over* $\mathbb{C}$ *of the specialized words.*

The following is an example of the specializations introduced in Remark 2:

$$\begin{aligned}
\mathfrak{e}(w(U_1^{(\sqrt{3},\sqrt{3})})) &= 1 \cdot e^{i\phi} \cdot e^{i\phi} \cdot 1,\\
\mathfrak{e}(w(U_2^{(\sqrt{3},\sqrt{3})})) &= 1 \cdot e^{i\phi} \cdot 1 \cdot e^{i\phi},\\
\mathfrak{e}(w(U_3^{(\sqrt{3},\sqrt{3})})) &= 1 \cdot 1 \cdot e^{i\phi} \cdot e^{i\phi},\\
\mathfrak{e}(w(U_4^{(\sqrt{3},\sqrt{3})})) &= 1 \cdot 1 \cdot 1 \cdot e^{i\phi},\\
\mathfrak{e}(w(U_5^{(\sqrt{3},\sqrt{3})})) &= 1 \cdot 1 \cdot e^{i\phi} \cdot 1,\\
\mathfrak{e}(w(U_6^{(\sqrt{3},\sqrt{3})})) &= 1 \cdot e^{i\phi} \cdot 1 \cdot 1.
\end{aligned} \tag{11}$$

The specialized message $\mathfrak{M}^{\mathfrak{e}}(\mathfrak{B}(\sqrt{3},\sqrt{3}),f^1,f^2) = 3e^{2i\phi} + 3e^{i\phi}$. If $\phi = \pi/2$, then $M^{\mathfrak{e}}(\mathfrak{B}(\sqrt{3},\sqrt{3}),f^1,f^2)\,|_{\phi=\pi/2} = -3+3i$.

If $x_1 = 4$, we can define new Brauer configuration algebras and corresponding messages associated with points $(r,s) \in \{(x,y) \in \mathcal{D} \cap (\mathbb{Z}\times\mathbb{Z}) \mid y \leq -x+4\}$ and straight lines given by the formulas $f_\alpha^1(x) = -x+\alpha$ for $(0 \leq x_0 = i \leq 4) \cap \mathbb{N}$, $\quad 0 \leq \alpha \leq 4$.

If $x_0 = x_1 = 4$, then $f_4^1(x) = f_4^2(x) = -x+4$, $\quad 0 \leq x \leq 4$.

The labeled Brauer configuration $\mathfrak{B}((2,2),f_4^1,f_4^2) = (\mathfrak{B}_0^{(2,2)},\mathfrak{B}_1^{(2,2)},\mu^{(2,2)},\mathcal{O}^{(2,2)})$ is defined as follows:

- $\mathfrak{B}_0^{(2,2)} = \{x, \mathfrak{m}\}$.
- $|\mathfrak{B}_1^{(2,2)}| = |Q_0| = \binom{4}{2} = 6$.
- $\mathfrak{B}_1^{(2,2)} = \left\{(U_i^{(2,2)}, \mathcal{L}_i^{(2,2)}) \mid 1 \leq i \leq 6\right\}$,

  $\mathcal{L}_1^{(2,2)} = xx\mathfrak{m}\mathfrak{m}, \quad \mathcal{L}_2^{(2,2)} = x\mathfrak{m}x\mathfrak{m}, \quad \mathcal{L}_3^{(2,2)} = x\mathfrak{m}\mathfrak{m}x,$
  $\mathcal{L}_4^{(2,2)} = \mathfrak{m}\mathfrak{m}xx, \quad \mathcal{L}_5^{(2,2)} = \mathfrak{m}x\mathfrak{m}x, \quad \mathcal{L}_6^{(2,2)} = \mathfrak{m}xx\mathfrak{m}.$
- $\mu^{(2,2)}(x) = \mu^{(2,2)}(\mathfrak{m}) = 1$.
- The successor sequences S_x and $S_\mathfrak{m}$ at vertices x and $\mathfrak{m}$ have the following form:

$$(U_1^{(2,2)})^{(2)} < (U_2^{(2,2)})^{(2)} < (U_3^{(2,2)})^{(2)} < (U_4^{(2,2)})^{(2)} < (U_5^{(2,2)})^{(2)} < (U_6^{(2,2)})^{(2)}.$$

- $val(x) = val(\mathfrak{m}) = 12$.
- $\dim_\mathbb{F} \Lambda_{\mathfrak{B}((2,2),f_3^1,f_3^2)} = 276$.
- $\dim_\mathbb{F} Z(\Lambda_{\mathfrak{B}((2,2),f_4^1,f_4^2)} = 19$.

$$\mathfrak{M}^e(\mathfrak{B}((2,2),f_4^1,f_4^2)) \mid_{\phi=\pi} = \mathfrak{M}(4,4,2) = 6e^{2i\pi} = 6. \tag{12}$$

Figure 8 shows the Brauer quiver induced by the Brauer configuration $\mathfrak{B}((2,2),f_4^1,f_4^2)$.

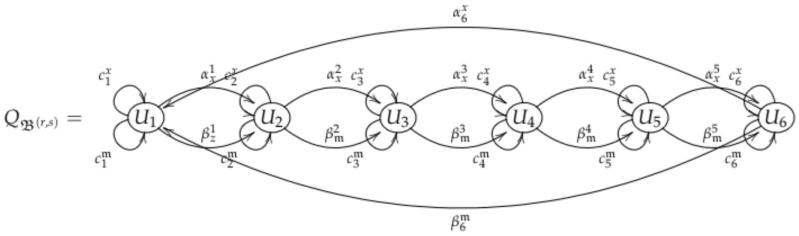

Figure 8. Brauer quiver associated with the labeled Brauer configuration $\mathfrak{B}((2,2),f_4^1,f_4^2)$.

The specialized messages $\mathfrak{M}^e(\alpha,\gamma,\beta)$ associated with Brauer configurations $\mathfrak{B}((r,s),f_\alpha^1(x) = -x+\alpha, f_4^2(x) = -x+4)$ with $0 \leq \alpha \leq 4$ are given in the following Table 1.

Table 1. Specialized messages of type $\mathfrak{M}^e(\mathfrak{B}((r,s),y=-x+\alpha,y=-x+4)), 0 \leq \alpha \leq 4$.

$\mathfrak{M}^e((3,1),f_0^1,f_4^2) = 4$	$\mathfrak{M}^e((2,2),f_0^1,f_4^2) = 6$	$\mathfrak{M}^e((1,3),f_0^1,f_4^2) = 4$
$\mathfrak{M}^e((3,1),f_1^1,f_4^2) = 2$	$\mathfrak{M}^e((2,2),f_1^1,f_4^2) = 0$	$\mathfrak{M}^e((1,3),f_1^1,f_4^2) = -2$
$\mathfrak{M}^e((3,1),f_2^1,f_4^2) = 0$	$\mathfrak{M}^e((2,2),f_2^1,f_4^2) = -2$	$\mathfrak{M}^e((1,3),f_2^1,f_4^2) = 0$
$\mathfrak{M}^e((3,1),f_3^1,f_4^2) = -2$	$\mathfrak{M}^e((2,2),f_3^1,f_4^2) = 0$	$\mathfrak{M}^e((1,3),f_3^1,f_4^2) = 2$
$\mathfrak{M}^e((3,1),f_4^1,f_4^2) = -4$	$\mathfrak{M}^e((2,2),f_4^1,f_4^2) = 6$	$\mathfrak{M}^e((1,3),f_4^1,f_4^2) = -4$

Notation $\mathfrak{M}^e((r,s),f_\alpha^1,f_4^2) = \mathfrak{M}(\alpha,4,\beta), 0 \leq \alpha \leq 4, 1 \leq \beta \leq 3$ in Table 1 gives rise to a 5×5-matrix $\mathfrak{M}_4$, for which $\mathfrak{M}(\alpha,4,0) = 1$ and $\mathfrak{M}(\alpha,4,4) = (-1)^\alpha$.

$$\mathfrak{M}_4 = \begin{bmatrix} \mathfrak{M}^e(0,4,0) & \mathfrak{M}^e(1,4,0) & \mathfrak{M}^e(2,4,0) & \mathfrak{M}^e(3,4,0) & \mathfrak{M}^e(4,4,0) \\ \mathfrak{M}^e(0,4,1) & \mathfrak{M}^e(1,4,1) & \mathfrak{M}^e(2,4,1) & \mathfrak{M}^e(3,4,1) & \mathfrak{M}^e(4,4,1) \\ \mathfrak{M}^e(0,4,2) & \mathfrak{M}^e(1,4,2) & \mathfrak{M}^e(2,4,2) & M^e(3,4,2) & \mathfrak{M}^e(4,4,2) \\ \mathfrak{M}^e(0,4,3) & \mathfrak{M}^e(1,4,3) & \mathfrak{M}^e(2,4,3) & \mathfrak{M}^e(3,4,3) & \mathfrak{M}^e(4,4,3) \\ \mathfrak{M}^e(0,4,4) & \mathfrak{M}^e(1,4,4) & \mathfrak{M}^e(2,4,4) & \mathfrak{M}^e(3,4,4) & \mathfrak{M}^e(4,4,4) \end{bmatrix}$$

We call these matrices *message-matrices associated with* $f_{x_1}^2$.

The next Theorem 4 gives some properties of message-matrices $\mathfrak{M}_\mathfrak{B} = (\mathfrak{M}(\alpha,\gamma,\beta)) = (\mathfrak{M}^e(\mathfrak{B}((\alpha,\gamma-\alpha),f_\alpha^1,f_\gamma^2))$ with $x_1 = \gamma > 1, \quad 0 \leq \alpha \leq \gamma$.

Theorem 4. *For fixed non-negative integers α, β, an integer $\gamma > 1$; the message-matrix $\mathfrak{M}_\gamma$ associated with a linear map f^2_γ satisfies the following identities:*

1. $\mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta - 1) + \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta) = \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma + 1, \beta)$.
2. $\mathfrak{M}^{\mathfrak{e}}(\alpha - 1, \gamma, \beta) + \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta) + \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta + 1) = \mathfrak{M}^{\mathfrak{e}}(\alpha - 1, \gamma, \beta + 1)$.
3. $\mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta - 1) = \sum\limits_{h \in \mathbb{N}} (-1)^h \binom{\gamma-1}{h} \binom{\beta-1-\gamma}{\alpha-h-1}$, $\quad \binom{v}{u} = 0$, *if* $u > v$.
4. $\mathfrak{M}^{\mathfrak{e}}(0, \gamma, \beta) = \binom{\gamma}{\beta}$.
5. $\mathfrak{M}^{\mathfrak{e}}(1, 2m + 1, m)$ *is the number of Dyck paths P in the (x, y) plane from $(0, 0)$ to $(2m, 0)$ with steps $(1, 1)$ and $(1, -1)$ that never pass below the x-axis.*
6. $(\mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta))^2 = 2^\gamma I_{\gamma+1}$, *where I_γ denotes the $\gamma \times \gamma$ identity matrix.*
7. *The determinant $|\mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta)| = (-2)^{t_\gamma}$, where $t_\gamma = \frac{\gamma(\gamma+1)}{2}$ denotes the γth triangular number.*

Proof.

1. The first item follows from the identity

$$\begin{aligned} \mathfrak{M}^{\mathfrak{e}}(\mathfrak{B}((\alpha, \gamma - \alpha), f^1_\alpha, f^2_{\gamma+1})) &= s(1) + t(1), \text{ where} \\ s(x) &= x\mathfrak{M}^{\mathfrak{e}}(\mathfrak{B}((\alpha - 1, \gamma - \alpha), f^1_\alpha, f^2_\gamma)), \\ t(y) &= y\mathfrak{M}^{\mathfrak{e}}(\mathfrak{B}((\alpha, \gamma - \alpha - 1), f^1_\alpha, f^2_\gamma)). \end{aligned} \tag{13}$$

2. (Induction) Suppose that entries of a $n \times n$-matrix B are obtained from entries of an $(n-1) \times (n-1)$ matrix A via the property 1 and that entries of matrix A satisfy the property 2. According to the description, it suffices to prove that $b_9 = b_5 + b_6 + b_{10}$. Indeed,

$$A = \begin{bmatrix} a_1 & a_2 & \bullet \\ a_4 & a_5 & \bullet \\ a_7 & a_8 & \bullet \end{bmatrix} \qquad B = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet \\ b_5 & b_6 & \bullet & \bullet \\ b_9 & b_{10} & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet \end{bmatrix}$$

Suppose that matrix A satisfies property 2, i.e., $\mathfrak{M}^{\mathfrak{e}}(\alpha - 1, \gamma, \beta - 1) + \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta) + \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta + 1) = \mathfrak{M}^{\mathfrak{e}}(\alpha - 1, \gamma, \beta + 1)$. Thus,

$$a_4 + a_7 = (a_1 + a_2 + a_5) + (a_4 + a_5 + a_8), \tag{14}$$

$$a_4 + a_7 = (a_1 + a_4) + (a_2 + a_5) + (a_5 + a_8), \tag{15}$$

therefore,

$$a_4 + a_7 = b_9, \quad a_1 + a_4 = b_5, \quad a_2 + a_5 = b_6 \quad \textit{and} \quad a_5 + a_8 = b_{10}, \tag{16}$$

$b_9 = b_5 + b_6 + b_{10}$ (see identities (14) and (15)).

3. The identity is a direct consequence of property 1. Note that $\mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta - 1) = \mathfrak{M}^{\mathfrak{e}}(\alpha - 1, \gamma, \beta - 2) + \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta - 2) = \sum\limits_{h \in \mathbb{N}} (-1)^h \binom{j-1}{h} \binom{\beta-2-\gamma}{\alpha-h-2} + \sum\limits_{h \in \mathbb{N}} (-1)^h \binom{\gamma-1}{h} \binom{\beta-2-\gamma}{\alpha-h-1} = \sum\limits_{h \in \mathbb{N}} \binom{\gamma-1}{h} [\binom{\beta-2-\gamma}{\alpha-h-2} + \binom{\beta-2-\gamma}{\alpha-h-1}] = \sum\limits_{h \in \mathbb{N}} (-1)^h \binom{\gamma-1}{h} \binom{\beta-1-\gamma}{\alpha-h-1} = \mathfrak{M}^{\mathfrak{e}}(\alpha, \gamma, \beta - 1)$.
4. If we proceed by induction, assuming that the assertion is true for any matrix $\mathfrak{M}^{\mathfrak{e}}(\alpha, \iota, \gamma)$, $1 \leq \iota \leq \beta - 1$, then it holds that $\mathfrak{M}^{\mathfrak{e}}(0, \gamma - 1, \beta - 1) + \mathfrak{M}^{\mathfrak{e}}(0, \gamma - 1, \beta) = \mathfrak{M}^{\mathfrak{e}}(0, \gamma, \beta) = \binom{\gamma-1}{\beta-1} + \binom{\gamma-1}{\beta} = \binom{\gamma}{\beta}$.
5. It follows from the identity $C_m = M^{\mathfrak{e}}(1, 2m + 1, m)$, where $C_m = \frac{1}{m+1} \binom{2m}{m+1}$ is the mth Catalan number.

6. By construction, we note that if $R^{\mathfrak{e}}_{\alpha_0}$ ($C^{\mathfrak{e}}_{\gamma_0}$) is the α_0th row (γ_0th column of $\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)$)., then the inner product

$$\langle R^{\mathfrak{e}}_{\alpha_0}, C^{\mathfrak{e}}_{\gamma_0}\rangle = \begin{cases} \sum_{\iota=0}^{\gamma} \binom{\gamma+1}{\iota}, & \text{if } \alpha_0 \neq \gamma_0, \\ 0, & \text{otherwise.} \end{cases}$$

7. $|\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)|^2 = (\pm|\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)|)^2 = 2^{\gamma(\gamma+1)}$

□

The following Corollary 1 uses the properties given in Theorem 4 to obtain explicit solutions of some linear and nonlinear systems of differential equations.

Corollary 1. *If* $i^2 = -1$, $cos(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t) = \frac{e^{it\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)}+e^{-it\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)}}{2}$, $sin(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t) = \frac{e^{it\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)}-e^{-it\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)}}{2}$. *And* $tg(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t) = sin(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t)(cos(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t))^{-1}$. *Then*

1. $cos(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t)$ *is a solution of the linear system of differential equations*

$$X''(t) + 2^{\gamma-1}X(t) = 0. \tag{17}$$

2. $tg(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t)$ *is a solution of the nonlinear system of differential equations*

$$X'(t) - X^2(t)\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t = \mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)t. \tag{18}$$

Proof. It suffices to note that

$$\begin{aligned}
e^{t\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)} &= \frac{1}{2^{\frac{\mathfrak{B}-1}{2}}} Sh(2^{\frac{\gamma-1}{2}}t)\mathfrak{M}^{\mathfrak{e}}(\alpha,\beta,\gamma) + Ch(2^{\frac{\beta-1}{2}}t)I_\gamma, \\
e^{-t\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)} &= -\frac{1}{2^{\gamma-1}2} Sh(2^{\frac{\gamma-1}{2}}t)\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta) + Ch(2^{\frac{\beta-1}{2}}t)I_\gamma, \\
\frac{d}{dt}(e^{t\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)}) &= Ch(2^{\frac{k-1}{2}}t)\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta) + \frac{2^{\frac{\beta-1}{2}}}{2^{\beta-1}} Sh(2^{\frac{\beta-1}{2}}t)(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta))^2 = \\
&\quad \mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)(Ch(2^{\frac{\beta-1}{2}}t)I_\gamma + \frac{1}{2^{\frac{\beta-1}{2}}} Sh(2^{\frac{\gamma-1}{2}}t)\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)) = \\
&\quad \mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)e^{t\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)}. \\
\frac{d}{dt}(e^{-t\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)}) &= -Ch(2^{\frac{\gamma-1}{2}}t)\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta) + \frac{2^{\frac{\gamma-1}{2}}}{2^{\gamma-1}} Sh(2^{\frac{\gamma-1}{2}}t)(\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta))^2 = \\
&\quad \mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)(-Ch(2^{\frac{\gamma-1}{2}}t)I_\gamma + \frac{1}{2^{\frac{\gamma-1}{2}}} Sh(2^{\frac{\gamma-1}{2}}t)\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)) = \\
&\quad -\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)e^{-t\mathfrak{M}^{\mathfrak{e}}(\alpha,\gamma,\beta)}.
\end{aligned} \tag{19}$$

where $Sh(x)$ ($Ch(x)$) denote the usual hyperbolic sine (cosine) function. □

4. Cayley Hash Values of Brauer Messages Associated with the Plane

In this section, we compute and analyze Cayley hash values of Brauer messages $\mathfrak{M}^{\mathfrak{e}}(\mathfrak{B}(r,s), f^1_\alpha, f^2_\gamma)$.

Cayley hash functions are examples of provably secure hash functions. The following Algorithm 2 allows building a hash function from an expander Cayley graph:

Algorithm 2: Construction of a Cayley hash

- Fix a finite (semi)group $\mathfrak{H}$ with a set of generators $\mathfrak{s}$ with the same size as the text alphabet $\mathfrak{A}$.
- Choose an injective function $\mathfrak{f} : \mathfrak{A} \to \mathfrak{s}$.
- The hash function of the text or word $x_1x_2 \dots x_k$ is the (semi)group element $\mathfrak{f}(x_1)\mathfrak{f}(x_2) \dots \mathfrak{f}(x_k)$.

Remark 3. *Note that, under these circumstances, message specializations* $\mathfrak{e} : \mathfrak{B}_0((x, y), f^1, f^2) \to \mathbb{C}$, *as those defined in Remark 2 with* $\mathfrak{e}(x) = i$ *and* $e(\mathfrak{m}) = -i$, *are nothing but Cayley hash functions if* $f^1 = f^2$. *In such a case, we assume that* $\mathfrak{H}$ *is the group of nth roots of unity for* $n > 1$ *fixed.*

As an example of the setting posed in Remark 3. We assume $n = 4$, $\mathfrak{H} = \{1, -1, i, -i\}$, and $\mathfrak{s} = \{i = \mathfrak{e}(x), -i = \mathfrak{e}(\mathfrak{m})\}$. Thus, up to permutation $\mathfrak{M}^{\mathfrak{e}}(\mathfrak{B}(2, 1), -x + 3, -x + 3) = xx\mathfrak{m}x\mathfrak{m}x\mathfrak{m}xx \mid_{x=i,\mathfrak{m}=-i} = i$ is the corresponding Cayley hash.

Figure 9 shows the Cayley graph defined in Remark 3. Labels of the arrows are given by the set $\mathfrak{s} = \{i, -i\}$, i.e., the arrow $a \xrightarrow{b} c$ denotes the identity $ab = c$.

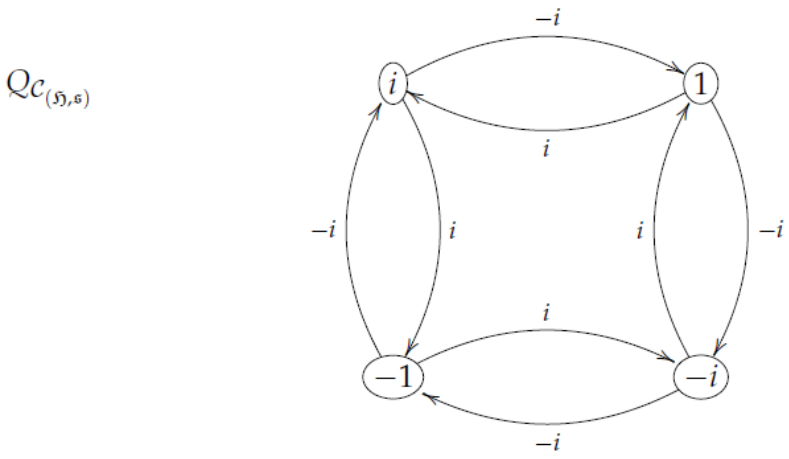

Figure 9. Example of a Cayley graph associated with the 4th unity roots, with $\mathfrak{s} = \{i, -i\}$.

It is worth pointing out that Tillich and Zémor introduced hash functions associated with the special group $\mathrm{Sl}(2, \mathcal{R})$ with $\mathcal{R} = \mathbb{Z}_2[x]/(p(x)) \cong \mathbb{Z}_2^n$, where $p(x)$ is an irreducible polynomial of degree n over $\mathbb{Z}_2$. $A_0 = \begin{bmatrix} x & 1 \\ 1 & 0 \end{bmatrix}$ and $A_1 = \begin{bmatrix} x & x+1 \\ 1 & 1 \end{bmatrix}$ are the generators of the Tillich–Zémor hash function. In such a case, if $\mathfrak{M} = m_1m_2 \dots m_k \in \{0, 1\}^*$, then the corresponding Tillich–Zémor hash function $H : \{0, 1\}^* \to \mathrm{Sl}(2, \mathcal{R})$ assigns the matrix product $A_{m_1}A_{m_2} \dots A_{m_k} \mod p(x)$, $m_i \in \{0, 1\}$ to the message $\mathfrak{M} = m_1m_2 \dots m_k \in \{0, 1\}^*$.

The Tillich–Zémor ideas were generalized by Sosnovski [2], who introduced the semigroup generated by the linear functions $f_0(x) = 2x + 1$ and $f_1(x) = 3x + 1$ over the field $\mathbb{Z}_p$ with $p > 3$. Her algorithm (Algorithm 3) goes as follows:

Algorithm 3: Sosnovski hash function.

- Consider the matrices $A_0 = \begin{bmatrix} 2 & 1 \\ 0 & 1 \end{bmatrix}$ and $A_1 = \begin{bmatrix} 3 & 1 \\ 0 & 1 \end{bmatrix}$.
- Apply the assignment $0 \to A_0, \quad 1 \to A_1$.
- Compute $h(b_1 b_2 \dots b_k) = A_{b_1} A_{b_2} \dots A_{b_k} \mod p$.
- The associated linear function has the form $L(x) = rx + s$.
- The hash value has the form $(r + s, s)$.

According to Sosnovski [2], a binary text is associated with a directed path P in the Cayley graph generated by f_0 and f_1 (the initial vertex $s(P)$ of P is given by the identity). Her algorithm hashes a bit string with a time complexity of at most $2n$ multiplications and $2n$ additions in $\mathbb{F}_p$. In $\mathbb{Z}_p$, each addition (multiplication) requires $O(log\ p)$ ($O(log^2\ p)$) bit operations. Thus, if $p \approx 2^m$ for m fixed, then the number of bit operations to evaluate a hash value of an n bit string is $O(m^2 n)$.

The family of hash functions has the following properties:

- Variable size input and fixed output size.
- The Cayley graphs of the semigroup have relatively large girth.
- Efficient computation.
- Pseudorandom.
- Collision resistant.

In this paper, we compute the Sosnovski hash function to the Brauer messages $\mathfrak{M}(\mathfrak{B}(x,y), f^1, f^2)$. To perform this, we apply the assignment $x \to A_0$, $y \to A_1$,

$$\mathfrak{m} = A_0^{-1} A_1 A_0 = A_2.$$

In other words, we obtain specialized Brauer messages

$$H(\mathfrak{M}(\mathfrak{B}(x,y), f^1, f^2)) = H(\mathfrak{M}^{\mathfrak{e}}(\mathfrak{B}(x,y), f^1, f^2) \mid_{x=A_0, y=A_1, \mathfrak{m}=A_2}).$$

For $i > 1$, the Sosnovski–Brauer hash $H(\mathfrak{M}_i) = \mathfrak{h}_i$ of a matrix $\mathfrak{M}_i$ is recursively obtained by applying the following procedure:

It is worth pointing out that the multiplication of two of these matrices takes $O(2^{2,3728596})$. Thus, constructing $\mathcal{H}_i$ requires $O(2^{2,3728596(i+1)})$ bit operations, which can be reduced by using Theorem 5 and Sosnovki [2] arguments regarding the complexity of her hash function.

The following algorithm allows obtaining any block B_j^i from the seeds B_0^{i-1} and B_0^i. Henceforth, all the products are assumed to be computed modulo a fixed prime number p.

Python gives the complexity and execution time of the algorithm proposed as $O(2^n)$ and $1.3 \times (0.97)^n$ seconds, respectively, if $\mathcal{H}_i$ consists of n, 2×2 matrices, bearing in mind that choosing an orientation $\mathcal{O}^{(r,s)}$ associated with a Brauer configuration $\mathfrak{B}^{(r,s)}$ at a point (r,s) has complexity $O(\binom{r+s}{s}^2)$. Such an orientation gives the appropriate matrix multiplication sequence. For instance, for modulo $2^{127} - 1$, we have that $\mathcal{H}_1 = A_0 A_1 A_0 A_2$, where $B_1^1 = A_0 A_2$.

$$
\begin{aligned}
B_{0,1}^2 &= A_0A_0,\\
B_{0,2}^2 &= A_1A_0,\\
B_{0,3}^2 &= A_0A_1,\\
B_{0,4}^2 &= A_1A_1,\\
B_0^2 &= A_0A_0A_1A_0A_0A_1A_1A_1,\\
B_1^2 &= A_0A_0A_1A_0A_0A_2A_1A_2,\\
B_2^2 &= A_0A_0A_2A_0A_0A_2A_2A_2,\\
\mathcal{H}_2 &= A_0A_0A_1A_0A_0A_1A_1A_1A_0A_0A_1A_0A_0A_2A_1A_2A_0A_0A_2A_0A_0A_2A_2A_2.\\
\mathfrak{h}_2 &= (3825661771, 1648879435) = (3825661771, 1648879435) \mod 2^{127}-1.
\end{aligned}
\tag{20}
$$

The following Theorem 5 gives formulas for some Sosnovski hash functions associated with arbitrary products of the matrices A_0, A_1, and A_2 (all the operations are assumed to be computed modulo a fixed prime number p).

Theorem 5. *If $j_0 = 0$, $j_i, h_i \geq 1$,*

$$
\begin{aligned}
a_{11} &= 2^{\sum_{t=1}^{m} j_{2t-1}} 3^{\sum_{t=1}^{m-1} j_{2t}},\\
a_{12} &= (2^{j_1}-1) + \sum_{k=1}^{m-1} 2^{\sum_{t=1}^{k} j_{2t-1}} 3^{\sum_{t=1}^{k} j_{2t}} (2^{j_{2k+1}}-1),\\
a'_{12} &= \sum_{k=1}^{m-1} 2^{\sum_{t=1}^{k} j_{2t-1}} 3^{\sum_{t=1}^{k} j_{2t-2}} (\frac{3^{j_{2k}}-1}{2}),\\
b_{11} &= 3^{\sum_{t=1}^{m} h_{2t-1}} 2^{\sum_{t=1}^{m-1} h_{2t}},\\
b_{12} &= \sum_{k=1}^{m} 3^{\sum_{t=1}^{k} h_{2t-1}} 2^{\sum_{t=1}^{k-1} h_{2t}} (2^{h_{2k}}-1),\\
b'_{12} &= \frac{3^{h_1}-1}{2} + \sum_{k=1}^{m} 3^{\sum_{t=1}^{k} h_{2t-1}} 2^{\sum_{t=1}^{k} h_{2t}} \frac{(3^{h_{2k+1}}-1)}{2},\\
c_{11} &= 2^{\sum_{\substack{t\equiv 1 \bmod 3\\ t\leq m}} j_t} 3^{\sum_{\substack{t\equiv 0,2 \bmod 3\\ 0<t<m}} j_t},\\
c_{12} &= \sum_{\substack{k\equiv 1 \bmod 3\\ k\leq m}} 2^{\sum_{\substack{t\equiv 1 \bmod 3\\ t<k}} j_t} 3^{\sum_{\substack{t\equiv 0,2 \bmod 3\\ 0<t<k}} j_t} (2^{j_k}-1),\\
c'_{12} &= \sum_{\substack{k\equiv 2 \bmod 3\\ k<m}} 2^{\sum_{\substack{t\equiv 1 \bmod 3\\ t<k}} j_t} 3^{\sum_{\substack{t\equiv 0,2 \bmod 3\\ 0<t<k}} j_t} \frac{(3^{j_k}-1)}{2},\\
c''_{12} &= \sum_{\substack{k\equiv 0 \bmod 3\\ k<m}} 2^{\sum_{\substack{t\equiv 1 \bmod 3\\ t<k}} j_t} 3^{\sum_{\substack{t\equiv 0,2 \bmod 3\\ 0<t<k}} j_t} \frac{3(3^{j_k}-1)}{4}.
\end{aligned}
\tag{21}
$$

Then,

$$
\begin{aligned}
\mathcal{H}(A_0^{j_1}A_1^{j_2}\dots A_0^{j_{2m-1}}) &= (a_{11}+a_{12}+a'_{12}, a_{12}+a'_{12}),\\
\mathcal{H}(A_1^{h_1}A_0^{h_2}\dots A_1^{h_{2m-1}}) &= (b_{11}+b_{12}+b'_{12}, b_{12}+b'_{12}),\\
\mathcal{H}(A_0^{j_1}A_1^{j_2}A_2^{j_3}\dots A_1^{j_{m-1}}A_2^{j_{m-1}}A_0^{j_m}) &= (c_{11}+c_{12}+c'_{12}+c''_{12}, c_{12}+c'_{12}+c''_{12}),\ \textit{for } m=1+3d, d\geq 0.
\end{aligned}
\tag{22}
$$

Proof. It suffices noting that for $k \geq 1$:

$$\begin{aligned}
\mathcal{H}(A_0^k) &= \mathcal{H}(\begin{bmatrix} 2^k & 2^k-1 \\ 0 & 1 \end{bmatrix}) = (2^{k+1}-1, 2^k-1), \\
\mathcal{H}(A_1^k) &= \mathcal{H}(\begin{bmatrix} 3^k & \frac{3^k-1}{2} \\ 0 & 1 \end{bmatrix}) = (\frac{3^{k+1}-1}{2}, \frac{3^k-1}{2}), \\
\mathcal{H}(A_2^k) &= \mathcal{H}(\begin{bmatrix} 3^k & \frac{3(3^k-1)}{4} \\ 0 & 1 \end{bmatrix}) = (\frac{3^{k+2}-1}{4}, \frac{3(3^k-1)}{4}).
\end{aligned} \tag{23}$$

□

Remark 4. *We recall that Cassaigne et al. [24] proved that if* $A = \begin{pmatrix} a & 0 \\ 0 & 1 \end{pmatrix}$, $B = \begin{pmatrix} b & 1 \\ 0 & 1 \end{pmatrix}$, *with* $a, b \in \mathbb{Q}\backslash\{-1, 0, 1\}$. *Then, the semigroup generated by* A *and* B *is free if* $|a| + |b| \leq 1$. *Additionally, there is a prime* p *for which* $v_p(a) and v_p(b) > 0$, *where* v_z *is the p-adic value of the number* z. *Such a result proves that two words in the semigroup generated by matrices* $A = \begin{pmatrix} a & b \\ 0 & c \end{pmatrix}$ *and* $B = \begin{pmatrix} a' & b' \\ 0 & c' \end{pmatrix}$ *are different if* A *and* B *do not commute.*

The following result proves that the Sosnovski function is collision-resistant for any Brauer message $\mathfrak{M}_i$, $i \geq 1$.

Corollary 2. *Let* $cll(\mathfrak{h}_n)$ *be the number of collisions associated with the Sosnovski hash values of Brauer messages for a fixed positive integer* n. *Then,* $\lim_{n\to\infty} cll(\mathfrak{h}_n) = 0$.

Proof. The result holds as a direct consequence of the results described in Remark 4. □

5. Experimental Data

This section gives some experimental results obtained by running Python routines in an Acer Predator Helios (intel core i7, 11th generation). We give execution times for Sosnovski and Tillich–Zémor hash values of Brauer messages associated with points in $\mathcal{V} = \mathbb{R}^* \times \mathbb{R}^*$. Figure 10 shows the number of bits of the compressed Brauer messages.

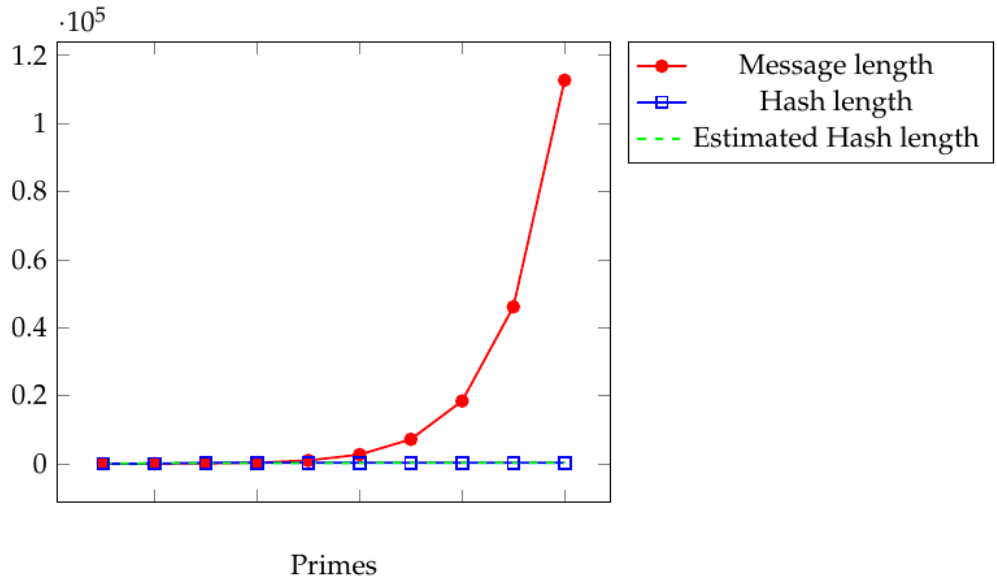

Figure 10. The specialized Brauer message given by a product of $(i+1)2^i$. Sosnovski matrices are compressed by $f(i)$ = 225.7621855345912 + 0.04323899371069182 bits.

Tables 2 and 3 give matrices $\mathcal{H}_i$, $1 \leq i \leq 3$ (see Algorithms 4 and 5), which are hash values of Brauer messages in the sense of Tillich–Zémor (for $x = 1$) and Sosnovski, respectively. These tables also show the number of collisions between these messages for the prime numbers considered by Sosnovski for her experimental data. Routine 4 in the appendix gives these values for arbitrary prime numbers and any matrix $\mathcal{H}_i$.

Algorithm 4: Sosnovski –Brauer hash function.

1. For $i > 1$, $1 \leq k \leq 2^i$, $0 \leq j \leq i$, and a fixed prime P.
2. $B^i_{j,k} = A^j_{h_1} A^j_{h_2} \dots A^j_{h_i} \mod p$, $A^j_{h_t} \in \{A_0, A_1, A_2\}$.
3. Matrix $A^j_{h_i - j}$ is in the jth position of the block $B^i_{j,k}$.
4. Define $B^i_j = B^i_{j,1} B^i_{j,2} \dots B^i_{j,2^i} \mod p$.
5. Define $\mathcal{H}_i = B^i_0 B^i_1 \dots B^i_i \mod p$. B^i_0 is the seed of $\mathcal{H}_i = \begin{pmatrix} r^i_{11} & r^i_{12} \\ 0 & 1 \end{pmatrix}$.
6. Define $\mathfrak{h}_i = (r^i_{11} + r^i_{12}, r^i_{12})$.

Algorithm 5: Building blocks B^i_j.

1. Define $B^1_{0,1} = A_0$, $B^1_{0,2} = A_1$, $B^1_0 = A_0 A_1$.
2. If $1 \leq k \leq 2^{i-1}$, then $B^i_{0,k} = B^{i-1}_{0,k} A_0$. If $k > 2^{i-1}$, then $B^i_{0,k} = B^{i-1}_{0,k \bmod 2^{i-1}} A_1$.
3. For $1 \leq j \leq i$, B^i_j is obtained by replacing any occurrence of $A_1 \in B^i_0$ in positions $\{0, 1, 2, \dots, j\}$ for A_2.

Tables 4 and 5 give the execution time to compute Tillich–Zémor and Sosnovski hash values of the Brauer messages. Figure 10 shows how Brauer messages are compressed by Sosnovski hash values. Figure 11 shows a comparison between execution times to compute Sosnovski and Tillich–Zémor hash values of the Brauer messages $\mathfrak{h}_3$, $\mathfrak{h}_4$, and $\mathfrak{h}_5$ for prime numbers $(2^{127} - 1)$ to $(2^{257} - 1053)$ given in Tables 4 and 5.

Table 2. Matrices $\mathcal{H}_i$, $1 \leq i \leq 3$ give Tillich–Zémor hash values of Brauer messages $\mathfrak{M}_i$.

P	Tillich–Zémor Hash			
	$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathcal{H}_3$	Collisions
$2^{127} - 1$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{137} - 555$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{147} - 387$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{157} - 213$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{167} - 771$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{177} - 919$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{187} - 477$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{197} - 775$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{207} - 429$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{217} - 675$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{227} - 721$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{237} - 949$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{247} - 309$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{256} - 1053$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0

Table 3. Matrices $\mathcal{H}_i$, $1 \leq i \leq 3$ are used to compute hash values $\mathfrak{h}_i$ (see Algorithm 4).

P	Matrices $\mathcal{H}_i$ Giving Sosnovski Hash Values $\mathfrak{h}_i$			
	$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathcal{H}_3$	Collisions
$2^{127}-1$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{137}-555$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{147}-387$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{157}-213$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{167}-771$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{177}-919$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{187}-477$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{197}-775$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{207}-429$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{217}-675$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{227}-721$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{237}-949$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{247}-309$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0
$2^{257}-1053$	$\begin{pmatrix} 4 & 3 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 12239 & 10101 \\ 9039 & 7460 \end{pmatrix}$	$\begin{pmatrix} 182280071328474839 & 157775084815395758 \\ 105888059171015748 & 91652902014389615 \end{pmatrix}$	0

Table 4. Execution time to compute Tillich–Zémor hash values of Brauer messages.

P	Tillich–Zémor Hash Execution Time				
	$\mathfrak{h}_1$	$\mathfrak{h}_2$	$\mathfrak{h}_3$	$\mathfrak{h}_4$	$\mathfrak{h}_5$
$2^{127}-1$	0.0004341602325439453	0.0006673336029052734	0.006798982620239258	0.03159761428833008	0.06526613235473633
$2^{137}-55$	0.000347137451171875	0.0006449222564697266	0.0022611618041992188	0.013956308364868164	0.05859231948852539
$2^{147}-387$	0.0003173351287841797	0.0006644725799560547	0.0023436546325683594	0.014194965362548828	0.060041189193725586
$2^{157}-213$	0.00032806396484375	0.0006654262542724609	0.002361774444580078	0.016431093215942383	0.06098580360412598
$2^{167}-771$	0.0003294944763183594	0.0006687641143798828	0.0022988319396972656	0.015888214111328125	0.06810140609741211
$2^{177}-919$	0.0004961490631103516	0.0006742477416992188	0.006218671798706055	0.015798568725585938	0.05859708786010742
$2^{187}-477$	0.00031685829162597656	0.0006656646728515625	0.002795696258544922	0.01443624496459961	0.07203435897827148
$2^{197}-775$	0.0003066062927246094	0.0006654262542724609	0.0022869110107421875	0.01398324966430664	0.056220293045043945
$2^{207}-429$	0.0002818107604980469	0.0006542205810546875	0.0022432804107666016	0.014203786849975586	0.05470752716064453
$2^{217}-675$	0.0002760887145996094	0.0006759166717529297	0.0022957324981689453	0.014241695404052734	0.05180048942565918
$2^{227}-721$	0.0002751350402832031	0.0006277561187744141	0.0022869110107421875	0.013881683349609375	0.05261635780334473
$2^{237}-949$	0.00027751922607421875	0.0006544589996337891	0.0023217201232910156	0.014061689376831055	0.06566166877746582
$2^{247}-309$	0.0002579689025878906	0.0006601810455322266	0.0023131370544433594	0.01390981674194336	0.05171465873718262
$2^{257}-1053$	0.0002589225769042969	0.0006580352783203125	0.002287626266479492	0.013902425765991211	0.06003284454345703

Table 5. Execution time to compute Sosnovski hash values of Brauer messsages.

P	Sosnovski Hash Execution Time				
	$\mathfrak{h}_1$	$\mathfrak{h}_2$	$\mathfrak{h}_3$	$\mathfrak{h}_4$	$\mathfrak{h}_5$
$2^{127}-1$	0.0038597583770751953	0.0007631778717041016	0.004364728927612305	0.01313328742980957	0.04121112823486328
$2^{137}-55$	0.0009815692901611328	0.0007708072662353516	0.003242969512939453	0.013436555862426758	0.043032169342041016
$2^{147}-387$	0.000997304916381836	0.0007264614105224609	0.004378318786621094	0.014588117599487305	0.044119834899902344
$2^{157}-213$	0.0009624958038330078	0.0007927417755126953	0.0033240318298339844	0.019335269927978516	0.041307926177978516
$2^{167}-771$	0.0012710094451904297	0.0008392333984375	0.0033843517303466797	0.013152837753295898	0.04137396812438965
$2^{177}-919$	0.0009815692901611328	0.0006961822509765625	0.0032160282135009766	0.012947797775268555	0.04337811470031738
$2^{187}-477$	0.0009517669677734375	0.0007886886596679688	0.0032584667205810547	0.01270604133605957	0.04137253761291504
$2^{197}-775$	0.001256704330444336	0.0008714199066162109	0.003565073013305664	0.022496700286865234	0.05827641487121582
$2^{207}-429$	0.0009984970092773438	0.0007412433624267578	0.0032558441162109375	0.013258218765258789	0.041890859603881836
$2^{217}-675$	0.0010428428649902344	0.0007922649383544922	0.0033080577850341797	0.01481318473815918	0.041900634765625
$2^{227}-721$	0.0010094642639160156	0.0009601116180419922	0.003567218780517578	0.01440572738647461	0.04564785957336426
$2^{237}-949$	0.003047943115234375	0.0007345676422119141	0.003594636917114258	0.014798164367675781	0.04285454750061035
$2^{247}-309$	0.0010731220245361328	0.0008180141448974609	0.003509044647216797	0.01487278938293457	0.043372392654418945
$2^{257}-1053$	0.0010068416595458984	0.0007958412170410156	0.003288745880126953	0.015767812728881836	0.04678201675415039

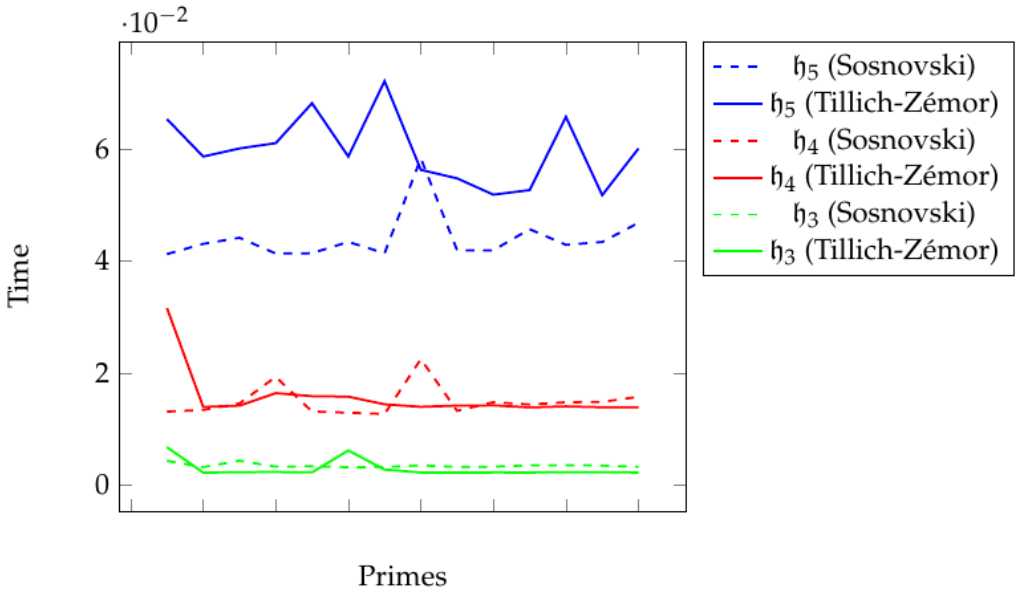

Figure 11. Execution time of the Tillich–Zémor and Sosnovski hash values of the Brauer messages $\mathfrak{h}_3 - \mathfrak{h}_5$.

Discussion

Brauer messages associated with regions without mutations are nothing but bit strings, so the Sosnovski algorithm can be applied to obtain their corresponding hash values without increasing the algorithm's execution time. On the other hand, the complexity of such an algorithm is exponential, and the execution time increases for Brauer messages associated with mutation regions.

Note that the hash values of Brauer messages associated with mutation regions obtained via Sosnovski matrices and their conjugates require fixing an orientation associated with a Brauer configuration $\mathfrak{B}^{(r,s)}$ at a point (r,s). To choose such an orientation $\mathcal{O}^{(r,s)}$ has complexity $O(((\binom{r+s}{s}))^2)$.

6. Concluding Remarks and Future Work

Brauer configuration algebras can be associated to each point $(x, y) \in \mathbb{R}^* \times \mathbb{R}^*$. Their dimensions can be computed by using appropriated binomial numbers. The same computations allow giving formulas for their centers. It is worth pointing out that suitable specializations allow obtaining matrices with applications in several scientific fields. For instance, such matrices give solutions to linear and no-linear differential equation systems. The specialized matrices give rise to elements in appropriated semigroups, which we call Brauer messages. The Sosnovski hash values of these messages are collision-resistant for large enough prime numbers. However, the execution time for giving such hashes is exponential when applied to Brauer messages with mutations.

Future Work

The following investigations are interesting tasks to be addressed in the future.

1. To introduce new classes of matrices defined by specializations of variables associated with general bounded regions.
2. To give explicit solutions for new classes of systems of differential equations by using specialized Brauer messages.
3. To investigate hash values of Brauer messages associated with more general groups and semigroups. For instance, there is no up-to-date hash function based on generators of any of the Thompson groups. These approaches would decrease the complexity and execution time when computing hash values of Brauer messages arising from mutation regions.

Author Contributions: Investigation, M.A.O.A., A.M.C., C.C.F., O.M.M. and R.-J.S.; writing—review and editing, M.A.O.A., A.M.C., C.C.F., O.M.M. and R.-J.S. All authors have read and agreed to the published version of the manuscript.

Funding: Seminar Alexander Zavadskij on Representation of Algebras and their Applications, Universidad Nacional de Colombia. The fourth author was partially supported by MinCiencias-Colombia, Convocatoria 907 de 2021.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

$\mathfrak{B}_0$	Set of vertices of a Brauer configuration $\mathfrak{B}$
$\mathbb{C}$	Complex numbers field
$\dim_{\mathbb{F}} \Lambda_{\mathfrak{B}}$	Dimension of a Brauer configuration algebra
$\dim_{\mathbb{F}} Z(\Lambda_{\mathfrak{B}})$	Dimension of the center of a Brauer configuration algebra
$\mathfrak{h}_i$	The Sosnovski hash value associated with a matrix Brauer message $\mathfrak{M}_i$
$\mathbb{N}$	The set of natural numbers
$\mathrm{occ}(\alpha, V)$	Number of occurrences of a vertex α in a polygon V
$\mathbb{R}$	The set of real numbers
t_i	ith triangular number
$V_i^{(\alpha)}$	Ordered sequence of polygons
$val(\alpha)$	Valency of a vertex α
$w(V)$	The word associated with a polygon V
$\lfloor (x) \rfloor$	The greatest integer less than or equal to x
$\lceil (x) \rceil$	The smallest integer greater than or equal to x

Appendix A. Python Routines

In this section, we give python routines to compute lattice paths (routine [2]), words associated with each point $(x, y) \in \mathbb{R}^* \times \mathbb{R}^*$, i.e., (routine [3]). In other words, routine [3] computes the set $\mathfrak{B}_1^{(x,y)}$ of the Brauer configuration $\mathfrak{B}^{(x,y)}$. Routine [4] computes matrices $\mathcal{H}_i$ associated with the semigroup generated by matrices A_0, A_1, and A_2 over $\mathbb{Z}$. Matrices $\mathcal{H}_i$, execution times, and collisions between hash values of specialized Brauer messages are computed modulo a prime number p via routine [7].

[1]:
```
import numpy as np
import itertools
from itertools import permutations
import matplotlib.pyplot as plt
import math
from math import factorial
from copy import deepcopy
import random
from random import sample
import time
import sys
import datetime
```

[2]:
```
def lattice_paths(n,k,f):
    allPaths = [ "".join( x ) for x in itertools.product( "xy", repeat=n+k )
             if x.count( "x" ) == n and x.count( "y" ) == k ]
    length =1
    width= 0.01
    head_width = 0.2
    x1=np.linspace(-1, n+1, 100)
    s=int(factorial(n+k)/(factorial(k)*factorial(n)))
    def drawPath(ax,path):
        x = n
        y = k
        ax.set_aspect('equal','box')
        for d in path:
            if d == "y":
                dx = 0.0
                dy = length
                if y-dy>=-x+f:
                    ax.arrow( x, y, -dx, -dy,
                          width = width,
                          head_width = head_width,
                          color = "blue",
                          length_includes_head = True)
                else:
                    ax.arrow( x, y, -dx, -dy,
                          width = width,
                          head_width = head_width,
                          color = "orange",
                          length_includes_head = True)
            else:
                dx = length
                dy = 0.0
                ax.arrow( x, y, -dx, -dy,
                      width = width,
                      head_width = head_width,
                      color = "red",
                      length_includes_head = True )
            x += -dx
            y += -dy
    for i, p in enumerate(allPaths):

        ax= plt.subplot(1,s,i+1)
        ax.plot(x1, -x1+f,'--', color='green')
```

```
        ax.plot(x1, -x1+n+k,'--', color='green')
        ax.axis( "off" )
        drawPath(ax,p)
```

[3]:
```
def allwords(n,k,f):
    def words(n,k,f,i):
        x = n
        y = k
        length =1
        allPaths = [ "".join( x ) for x in itertools.product("xy", repeat=n+k)
                 if x.count( "x" ) == n and x.count( "y" ) == k ]
        s=int(factorial(n+k)/(factorial(k)*factorial(n)))
        A=[list(allPaths[i]) for i in range(s)]
        for d in range(len(A[0])):
            if A[i][d]=='y':
                dx = 0
                dy = length
                if y-dy>=-x+f:
                    pass
                else:
                    A[i][d]='m'
            else:
                dy = 0
                dx = length
            x += -dx
            y += -dy
        return "".join(A[i])
    s=int(factorial(n+k)/(factorial(k)*factorial(n)))
    allwords=[words(n,k,f,i) for i in range(s)]
    return allwords
```

[4]:
```
A0=np.array([[2,1],[0,1]])
A1=np.array([[3,1],[0,1]])
A2=np.matmul(np.matmul(np.linalg.inv(A0),A1),A0)
def Hash(n,k,f):
    Words=allwords(n,k,f)
    Codification=[]
    for words in Words:
        matrix=[]
        for letter in words:
            if letter=='x':
                matrix.append(A0)
            if letter=='m':
                matrix.append(A2)
            if letter=='y':
                matrix.append(A1)
        Codification.append(matrix)
    Hash=[]
    for code in Codification:
        multiplication=np.identity(2)
        for i in  range(len(code)):
            multiplication=np.matmul(multiplication,code[i])
        Hash.append(multiplication)
    return Hash
def brauer_message(n,k,f):
    Multiplication=np.identity(2)
    for matriz in Hash(n,k,f):
        Multiplication=np.matmul(Multiplication,matriz)
    return print(Multiplication.astype(int))
```

[5]:
```
sys.float_info.max
sys.maxsize
def Module_Bigger_numbers(numb,mod):
    division=math.floor(numb/mod)
    res=numb-division*mod
    return int(res)

def Mult_matrix(matrix1,matrix2):
    Mult=[[0,0],[0,0]]
    for i in range(2):
        for j in range(2):
            for k in range(2):
                Mult[i][j] += int(matrix1[i][k] * matrix2[k][j])
    return Mult
def Num_bit(a,b,mod):
    sum=a+b
    s=Module_Bigger_numbers(sum,mod)
    sum_bit=sum.bit_length()
    b_bit=b.bit_length()
    return sum_bit+b_bit
```

[6]:
```
def collision(List_Hash):
    Colision=[]
    numcolision=[]
    for k in range(len(List_Hash)):
    List_Hash_copy=deepcopy(List_Hash)
    #print('Hashcopy',Hashcopy)
    Messages=[]
    #print(p,Hashcopy)
    for i in range(len(List_Hash_copy)):
        for j in range(len(List_Hash_copy)):
            if i!=j and i<j:
                if np.array_equal(List_Hash_copy[i],List_Hash_copy[j])==True:
                    Messages.append([List_Hash_copy[i],List_Hash_copy[j],i,j])
    Colision.append(Messages)
    #print(Mensajes)
    numcolision.append(len(Messages))
    return numcolision
```

[7]:
```
List_Hash=[]
Number_seed=[]
def Block_Hash(A0,A1,b,prime_list):
    List_Hash=[]
    Mod=[prime_list]
    Time=[]
    Collision=[]
    c=0
    Bit=[]
    for k in range(len(inv)):
        numb_matrix=0
        list_h=[]
        BBit=[]
        times=[]
        #print(Mod[k])
        st = time.time()
        mod=Mod[k]
        A0=A0.astype(int)
        A1=A1.astype(int)
        A20=np.multiply(np.linalg.det(A0),np.linalg.inv(A0))
```

```
        A20=A20.astype(int)
        #print(A20)
        H=np.matmul(np.matmul(A20,A1),A0)
        A2=np.multiply(inv[k],np.matmul(np.matmul(A20,A1),A0))
        for i in range(2):
            for j in range(2):
                A2[i][j]=Module_Bigger_numbers(A2[i][j],mod)
        #print(A2)
        Id=np.identity(2)
        Matrix=[A0,A1,A2]
        Seed=[A0,A1]
        HH=[A0,A1,A0,A2]
        H1=Id
        seed=[]
        for i in HH:
            H1=Mult_matrix(H1,i)
            numb_matrix+=1
        seed.append(numb_matrix)
        for i in range(2):
            for j in range(2):
                H1[i][j]=Module_Bigger_numbers(H1[i][j],mod)
        print(H1)
        et=time.time()
        elapsed_time=et-st
        times.append(elapsed_time)
        bit=Num_bit(H1[0][0],H1[0][1],mod)
        BBit.append(bit)
        cont=2
        list_h.append(H1)
        for k in range(b):
            numb_matrix=0
            st = time.time()
            #print('Start time',st)
            Block_seed=[]
            Seedc=deepcopy(Seed)
            if len(Seed)==2:
                for i in range(2):
                    for j in range(len(Seed)):
                        Block_seed.append([Seed[j],Matrix[i]])
            else:
                for i in range(2):
                    for j in range(len(Seedc)):
                        Seedc[j].append(Matrix[i])
                        Block_seed.append(Seedc[j])
                    Seedc=deepcopy(Seed)
            List_Block_mutations=[]
            next=0
            position=len(Block_seed[0])
            Block_seedc=deepcopy(Block_seed)
            while next!=position:
                Block_mutations=[]
                for i in range(len(Block_seedc)):
                    Block=[]
                    if np.array_equal(Block_seedc[i][position-1-next],A1)==True:
                        Block_seedc[i][position-1-next]=A2
                    for j in range(len(Block_seedc[i])):
                        Block.append(Block_seedc[i][j])
                    Block_mutations.append(Block)
                    for s in Block:
```

```
                        List_Block_mutations.append(s)
                Block_seedc=Block_mutations
                next+=1
            Hashmessage=Id
            for m in Block_seed:
                for n in m:
                    Hashmessage=Mult_matrix(Hashmessage,n)
                    numb_matrix+=1
                    for i in range(2):
                    for j in range(2):
                        ␣
↪Hashmessage[i][j]=Module_Bigger_numbers(Hashmessage[i][j],mod)
                    Hm=Hashmessage
            for t in List_Block_mutations:
                Hm=Mult_matrix(Hm,t)
                numb_matrix+=1
                for i in range(2):
                    for j in range(2):
                        Hm[i][j]=Module_Bigger_numbers(Hm[i][j],mod)
            seed.append(numb_matrix)
            print(Hm)
            list_h.append(Hm)
            Seed=Block_seed
            et= time.time()
            elapsed_time=et-st
            bit=Num_bit(Hm[0][0],Hm[0][1],mod)
            BBit.append(bit)
            times.append(elapsed_time)
            cont+=1
        List_Hash.append(list_h)
        Number_seed.append(seed)
        Time.append(times)
        Bit.append(BBit)
        coll=collision(list_h)
        Collision.append(coll)
        c+=1
    return List_Hash,Time, Bit, Collision, Mod, Number_seed
```

References

1. Stinson, D.; Paterson, M. *Cryptography: Theory and Practice*, 4th ed.; Chapman and Hall/CRC Press: Boca Raton, FL, USA, 2019. [CrossRef]
2. Sosnovski, B. Cayley Graphs of Semigroups and Applications to Hashing. Ph.D. Thesis, City University of New York, New York, NY, USA, 2016.
3. Espinosa, P.F.F. Categorification of Integer Sequences and Its Applications. Ph.D. Thesis, Universidad Nacional de Colombia, Bogotá, Colombia, 2020.
4. Green, E.L.; Schroll, S. Brauer configuration algebras: A generalization of Brauer graph algebras. *Bull. Sci. Math.* **2017**, *141*, 539–572. [CrossRef]
5. Cañadas, A.M.; Espinosa, P.F.F.; Rios, G.B. Wargaming with quadratic forms and Brauer configuration algebras. *Mathematics* **2022**, *10*, 729. [CrossRef]
6. Cañadas, A.M.; Gaviria, I.D.M.; Vega, J.D.C. Relationships between the Chicken McNugget Problem, Mutations of Brauer Configuration Algebras and the Advanced Encryption Standard. *Mathematics* **2021**, *9*, 1937. [CrossRef]
7. Green, E.L.; Hille, L.; Schroll, S. Algebras and varieties. *Algebr. Represent. Theor.* **2021**, *24*, 367–388. [CrossRef]
8. National Academies of Sciences, Engineering, and Medicine. *Quantum Computing: Progress and Prospects*; The National Academies Press: Washington, DC, USA, 2018. [CrossRef]
9. Zémor, G. Hash functions and graphs with large girths. In *Advances in Cryptology-EUROCRYPT'91*; Springer: Berlin/Heidelberg, Germany, 1991; Volume 547, pp. 508–511. [CrossRef]
10. Tillich, J.P.; Zémor, G. Group-theoretic hash functions. In *Algebraic Coding: First French-Israeli Workshop, Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 1994; Volume 781, pp. 90–110. [CrossRef]
11. Tillich, J.P.; Zémor, G. Hashing with SL_2. In *Advances in Cryptology-CRYPTO'94*; Springer: Berlin/Heidelberg, Germany, 1994; Volume 839, pp. 40–49. [CrossRef]

12. Charles, D.X.; Lauter, K.E.; Goren, E.Z. Cryptographic hash Functions from expander graphs. *J. Cryptol.* **2007**, *22*, 93–113. [CrossRef]
13. Petit, C. On Graph-Based Cryptographic Hash Functions. Ph.D. Thesis, Universit Catholique de Louvain, Ottignies-Louvain-la-Neuve, Belgium, 2009.
14. Petit, C.; Lauter, K.; Quisquater, J.-J. Full cryptanalysis of LPS and Morgenstern hash functions. In *Security and Cryptography for Networks, Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2008; Volume 5229. [CrossRef]
15. Lubotzky, A.; Phillips, R.; Sarnak, P. Ramanujan graphs. *Combinatorica* **1988**, *3*, 261–277. [CrossRef]
16. Grassl, M.; Ilić, I.; Magliveras, S.; Steiwandt, R. Cryptanalysis of the Tillich-Zémor hash function. *J. Cryptol.* **2011**, *24*, 148–156. [CrossRef]
17. Petit, C.; Quisquater, J.-J. Preimages of the Tillich-Zémor hash function. In *Selected Areas in Cryptography. SAC 2010. Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6544. [CrossRef]
18. Grassl, M.; Langenber, B.; Roetteler, M.; Steiwandt, R. Applying Grover's algorithm to AES: Quantum resource estimates. In *Post-Quantum Cryptography. PQCrypto 2016. Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9606. [CrossRef]
19. Mullan, C.; Tsaban, B. SL_2 homomorphic hash functions: Worst case to average case reduction and short collision search. In *Designs, Codes and Cryptography*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 81, pp. 83–107. [CrossRef]
20. Agudelo Muñeton, N.; Cañadas, A.M.; Gaviria, I.D.M.; Fernández, P.F.F. $\{0,1\}$-Brauer configuration algebras and their applications in the graph energy theory. *Mathematics* **2021**, *9*, 3042. [CrossRef]
21. Cañadas, A.M.; Angarita, M.A.O. Brauer configuration algebras for multimedia based cryptography and security applications. *Multimed Tools Appl.* **2021**, *80*, 23485–23510. doi: 10.1007/s11042-020-10239-3. [CrossRef]
22. Assem, I.; Skowronski, A.; Simson, D. *Elements of the Representation Theory of Associative Algebras*; Cambridge University Press: Cambridge, UK, 2006. [CrossRef]
23. Sierra, A. The dimension of the center of a Brauer configuration algebra. *J. Algebra* **2018**, *510*, 289–318. [CrossRef]
24. Cassaigne, J.; Harju, T.; Karhumäki, J. On the undecidability of freeness of matrix semigroups. *Intern. J. Algebra Comput.* **1999**, *9*, 295–305. [CrossRef]