Constructions for the cap set problem

Asymptotic lower bounds on the size of cap sets

by

I. Pelupessy

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Friday June 28, 2024 at 15:30.

Project duration:	April 22, 2024 – June 28, 2024	
Thesis committee:	D.C. Gijswijt,	TU Delft, supervisor
	E. Lorist,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

Layman's abstract

This thesis aims to introduce the reader to the cap set problem: a combinatorial problem that can be formulated in terms of the game SET. This is a card game where each card pictures symbols in various ways. The cards of SET have four features and each feature can take three values. For example a card can have the colour red, green, or purple. Players simultaneously search for SETs: three cards that for every feature are all the same or all different, see Figure 1. The game starts with twelve cards faced up, but if the players agree that these do not contain a SET, three extra cards may be added. A natural thing to question is how many cards can be added until the cards *must* contain a SET. Or in other words, what is the biggest possible collection of cards *without* SETs? We call such a collection of cards a cap, hence the name of the cap set problem.

The card game SET has four features, so we say that the dimension is four. In dimension four the problem has been solved. The cap set problem tries to solve the problem for a general dimension: how large can a collections of cards without SETs be, where the cards have a general number of features?

To figure out what the maximum size of a cap in a general dimension is, we will try to find a constant c such that a maximum cap in dimension d contains at least c^d cards. We refer to this constant c as an 'asymptotic lower bound' on the size of cap sets. To find good asymptotic lower bounds, we need to construct caps in high dimensions. This can be done by combining some big caps we know in low dimensions. This thesis explains a construction (called the extended product construction) that combines caps in a smart way and how this construction has been used to improve the asymptotic lower bound.

This thesis adds to the literature by expanding on the concepts that are used in the extended product construction. We incorporate examples of some special collections of caps (called extendable collections) in dimensions 1, 2, and 3. Moreover, we prove that there exist 'recursively admissible sets' where all elements of the set consist of exactly 2 or 3 non-zero coordinates by giving explicit constructions and proving that the resulting sets satisfy all necessary conditions. Lastly, we show that how effective the extended product construction may be in high dimensions, it does not allow us to build maximum caps in low dimensions.

I. Pelupessy
Delft, June 2024

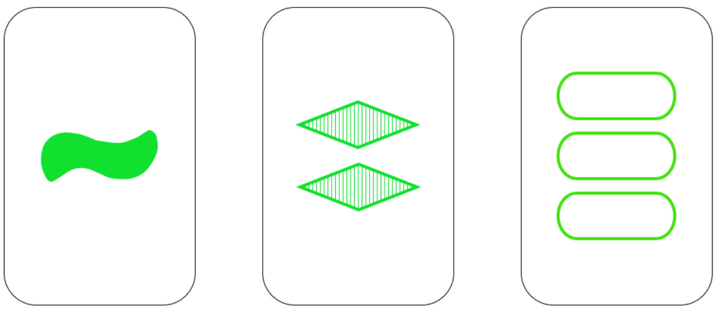


Figure 1: The three cards have the same colour, but for all other features they are all different.

Abstract

The objective of the cap set problem is finding the maximum size of a d-cap: a subset of $\mathbb{F}_3^d$ not containing three elements in line. This thesis aims to give a comprehensive overview of constructions for the cap set problem, with a focus on improvements of the asymptotic lower bound on the size of caps that have already been made.

Finding an asymptotic lower bound on the size of caps boils down to finding a cap C in a dimension d such that its solidity, given by $\sqrt[d]{|C|}$, is as large as possible. We start with studying caps in low dimensions, of which the maximum sizes are exactly known. Then to further improve the asymptotic lower bound we turn to caps in higher dimensions. Here, the art lies in carefully combining large caps in low dimensions to construct large caps in higher dimensions by taking products. One construction that allows us to do this is the extended product construction, which extends extendable collections of caps with admissible sets.

This thesis explains the extended product construction and gives an overview of how it has been used and expanded to repeatedly increase the asymptotic lower bound. As the literature sometimes lacks detail, this thesis adds to the literature by incorporating examples, explicit constructions of (recursively) admissible sets, and experiments with the extended product construction.

In Chapter 6, we prove the existence of recursively admissible sets of constant weight 2 and 3 for any dimension k by giving explicit constructions and proving that the resulting sets satisfy all necessary conditions. Moreover, we classify all admissible sets in dimensions 2 and 3 and all extendable collections in dimensions 1, 2, and 3. Then, we use these to deduce that the extended product construction is less effective in *low* dimensions by showing that the largest possible caps we can construct this way in dimensions 4, 6, and 8 are never as large as caps constructed by taking direct products of maximum caps.

I. Pelupessy
Delft, June 2024

Contents

1 Introduction

In 1974, geneticist Marsha Falco was doing research into the heredity of epilepsy among German Shepherds when she invented the card game SET. To study the genes, Falco made cards with symbols to represent blocks of data for each dog. She used symbols with different properties to indicate different combinations of genes. She then tried to find patterns among the cards and realized that this was quite a fun game. 17 years later SET was shared with the public [1].

The cards of SET have four features and each feature can take three possible values. For example the feature colour can take values red, green, or purple. Three cards form a SET together if for every feature the three cards are all different or all the same (see Figure 1 for an example). The players simultaneously look for SETs among a collection of faced up cards: the fastest player may keep the SET they find and whoever has the most SETs at the end of the game wins. The game starts with twelve cards on the table, but if the players agree that these do not contain a SET, three extra cards may be added. Now, one may wonder how many cards we can keep adding until a SET is guaranteed. Or equivalently: what is the biggest possible collection of cards without three cards forming a SET?

Mathematically this problem is described as finding the biggest possible subset of $\mathbb{F}_3^4$ *without* three points in line, where $\mathbb{F}_3$ denotes the field of three elements. We call a subset of $\mathbb{F}_3^d$ with no three points in line a cap set (or cap), hence the name of the cap set problem. As for the game SET, where the dimension is four, this problem is solved. In fact, for dimensions one through six the maximum sizes of caps are known. For dimensions seven and up this is still an open problem.

The cap set problem is a combinatorial problem that has many connections with other areas of mathematics. The solution to the cap set problem would for instance partially prove the sunflower conjecture. A k-sunflower is a collection of sets $A_1, A_2, \dots, A_k$ such that the pairwise intersections equal the k-wise intersection. Imagine the Venn diagram of such a collection and see where a sunflower gets its name. The sunflower conjecture states that given any k there is a constant b_k such that we can find a k-sunflower in any subset of $(\mathbb{Z}/m\mathbb{Z})^d$ of size at least $(b_k)^d$. The solution to the cap set problem would solve the sunflower conjecture for $m = 3$. Another application of the cap set problem is the Games graph: the largest known locally linear strongly regular graph, consisting of 729 vertices, each having 112 incident edges. This graph was constructed with the help of caps. Lastly, the upper bounds on caps imply lower bounds on certain types of algorithms for matrix multiplication. For a more detailed exposition of the applications of the cap set problem the reader is referred to [2].

The aim of this thesis is to give an overview of the best constructions for the cap set problem, with a focus on the asymptotic solidity of cap sets. This is the number that the solidity of maximum caps approaches as the dimension grows larger. This is given by $\sup_d \sqrt[d]{a_d}$ (as defined in [3]), where a_d denotes the maximum size of a cap in $\mathbb{F}_3^d$. Since the asymptotic solidity is at least the solidity of any cap we find, finding an asymptotic lower bound on the size of cap sets comes down to finding a cap such that its solidity is as large as possible. This thesis mainly focuses on explaining constructions for large cap sets in high dimensions, where the art lies in carefully combining large caps in low dimensions by taking products, such that their solidity improves the asymptotic lower bound.

This report is structured as follows. First, Chapter 2 introduces the space $\mathbb{F}_3^d$ and explains how the elements of this space correspond to cards of a d-dimensional SET game. Then, Chapter 3 treats what is known about maximum caps in low dimensions. After this we proceed with higher dimensions, where the problem boils down to creating caps such that their solidity is as large as possible. This can be done by combining low dimensional maximum caps to create a large cap in a higher dimension. One construction that takes products of caps in a very smart way, is the extended product construction by Edel, which is explained in Chapter 4. The extended product construction was used several times in literature to repeatedly improve the asymptotic lower bound, of which Chapter 5 gives an overview. Next, Chapter 6 adds to the literature by expanding on the concepts 'extendable collections' and 'admissible sets' used in the extended product construction. This chapter classifies extendable collections and admissible sets in low dimensions, gives explicit constructions for (recursively) admissible sets, and experiments with constructing maximum caps in low dimensions (of which the maximum size is known) using the extended product construction. Lastly, Chapter 7 gives conclusions and recommendations for further research.

2
The cap set problem

This chapter gives an introduction to the cap set problem. First it is explained how we go from the cap set problem in terms of the card game SET to the problem of finding the largest subset of $\mathbb{F}_3^d$ without lines. Then we elaborate on $\mathbb{F}_3^d$, the d-dimensional vector space over the field of three elements, and its subspaces.

2.1. From SET cards to elements of $\mathbb{F}_3^4$

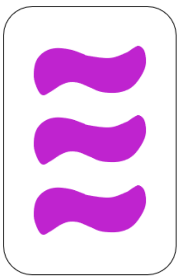
This section is based on 'The card game SET', by Davis and Maclagan [3], which gives a nice and clear introduction to the cap set problem. In this section it is explained how the cards of SET can be seen as elements of $\mathbb{F}_3^4$.

The cards of SET have four features: shape, colour, shading, and number of shapes. Each feature has three possible outcomes. The shape, for example, can be oval, squiggle, or diamond. The features and their possible outcomes are in Table 2.1.

Table 2.1: The four features of SET cards.

	Shape	**Colour**	**Shading**	**Number**
0	Oval	Red	Solid	One
1	Squiggle	Green	Striped	Two
2	Diamond	Purple	Open	Three

Suppose that for each feature, the three outcomes are numbered 0, 1 and 2. In Table 2.1 the rows are numbered accordingly. Each card of SET can then be modelled as a 4-dimensional vector, where a position corresponds to a feature and the entry in that position corresponds to the outcome of that feature. For example, if the second entry of a vector is 0, then the colour of the figures on the corresponding card is red. In Figure 2.1 it is demonstrated how a card that pictures three solid, purple squiggles is translated to a vector.



$$\vec{v} = \begin{bmatrix} \text{shape = squiggle} \\ \text{colour = purple} \\ \text{shading = solid} \\ \text{number = three} \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 0 \\ 2 \end{bmatrix}$$

Figure 2.1: The SET card corresponding to the vector $\vec{v} = (1, 2, 0, 2)^T$.

Observe that this way each of the 81 cards of SET corresponds to a unique element in $\{0,1,2\}^4$. We can define the set $\{0,1,2\}$ as a field of three elements, denoted $\mathbb{F}_3$. Definition 2.1 gives the formal definition of a field.

Definition 2.1. Field.
A field F *is a set of elements together with two binary operations, called addition and multiplication, such that:*

- *Addition and multiplication are associative, that is, for all* $a, b, c \in F$*:*
  $(a+b)+c = a+(b+c)$ *and* $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.
- *Addition and multiplication are commutative, that is, for all* $a, b, c \in F$*:*
  $a+b = b+a$ *and* $a \cdot b = b \cdot a$.
- *There are distinct elements* 0 *and* 1*, called the additive- and multiplicative identity respectively, such that for all* $a \in F$*:*
  $a+0 = a$ *and* $a \cdot 1 = a$.
- *Every element* a *has an additive inverse, denoted* $(-a)$*, such that* $a + (-a) = 0$.
- *Every non-zero element* a *has a multiplicative inverse, denoted* a^{-1}*, such that* $a \cdot a^{-1} = 1$.
- *Multiplication is distributive over addition, that is, for all* $a, b, c \in F$*:*
  $a \cdot (b+c) = (a \cdot b) + (a \cdot c)$.

Some commonly known *infinite* fields are the rational numbers $\mathbb{Q}$ and the real numbers $\mathbb{R}$. Consider the *finite* field $\mathbb{F}_3 = \{0,1,2\}$. Addition and multiplication are as how we normally use '$+$' and '$\cdot$', but then we calculate modulo 3. Then the elements of $\mathbb{F}_3$ add and multiply as follows:

$+$	0	1	2
0	0	1	2
1		2	0
2			1

$\cdot$	0	1	2
0	0	0	0
1		1	2
2			1

Note that the additive identity is 0 and the multiplicative identity is 1. The additive inverses are:

- $0+0=0$
- $1+2=0$
- $2+1=0$

This should not come as a surprise since $-1 \equiv 2 \pmod 3$ and $-2 \equiv 1 \pmod 3$. The multiplicative inverses of the non-zero elements are:

- $1 \cdot 1 = 1$
- $2 \cdot 2 = 4 \equiv 1 \pmod 3$

As mentioned before, the SET cards correspond to 4-dimensional vectors. More specific, each card is an element of the 4-dimensional vector space over the field of three elements $\{0,1,2\}^4 = \mathbb{F}_3^4$. Informally, a vector space is a set whose elements may be added together and multiplied ("scaled") by elements of a field, called *scalars*. The elements of the vector space are called *vectors*. Any vector space may be seen as an *affine space* by "forgetting" the special role played by the zero vector, as in an affine space there is no distinguished point that serves as an origin. The elements of an affine space are commonly called points.

2.2. Lines in $\mathbb{F}_3^d$

In the previous section we have seen that a card of SET can be modelled as an element of $\mathbb{F}_3^4$. But looking at elements of $\mathbb{F}_3^4$ makes the SET game more abstract, as the pictures have been omitted. When do three elements of $\mathbb{F}_3^4$ form a SET together? To define this, consider the following proposition:

Proposition 2.2. *If* a, b *and* c *are three distinct elements of* $\mathbb{F}_3$*, then* $a+b+c=0$ *if and only if* $a=b=c$ *or* $\{a,b,c\} = \{0,1,2\}$.

Proof. It is trivial that if $a = b = c$ or $\{a, b, c\} = \{0, 1, 2\}$, then $a + b + c = 0 \pmod 3$.

For the other direction, suppose that $a + b + c = 0$. If a, b and c are all different or all the same we are done. We are left with the case where only two are the same. But if $a = b$, then since $a + b + c = 0 \iff a - b = b - c$, we also have $b = c$, so this case cannot occur. Therefore if $a + b + c = 0$ then $a = b = c$ or $\{a, b, c\} = \{0, 1, 2\}$. □

Recall that three cards form a SET if for every feature they are all the same or all different, so for each feature either $a = b = c$ or $\{a, b, c\} = \{0, 1, 2\}$ in the corresponding coordinate. Thus by Proposition 2.2, three cards form a SET if and only if their associated vectors add up to zero. This argument works if the dimension 4 is replaced for any positive integer d.

Three elements of $\mathbb{F}_3^d$ form a *line* if and only if the vectors add up to zero. Therefore essentially, in a d-dimensional SET game the players are searching for lines contained in a subset of $\mathbb{F}_3^d$ (the cards on the table). The cap set problem asks how many cards can be on the table *without* any sets, that is, we are looking for subsets of $\mathbb{F}_3^d$ that do not contain any lines. A subset that does not contain any lines is called a d-cap:

Definition 2.3. d-cap.
$A \subset \mathbb{F}_3^d$ is called a d-cap if A does not contain any lines, i.e. for all distinct $x, y, z \in A$ we have $x+y+z \neq 0$.

Strictly speaking, the term ‘cap’ in general refers to a subset of a finite affine space with no three in line. In the specific case where the affine space is over the field $\mathbb{F}_3$ the term ‘cap set’ is used. However, Definition 2.3 defines the field to be $\mathbb{F}_3$, so in this report we will refer with the term ‘cap’ to a cap set in $\mathbb{F}_3^d$.

In terms of the card game SET, the cap set problem can be defined as follows:

Question 1. *What is the largest possible collection of SET cards not containing any SETs?*

We can reformulate this question in terms of the d-dimensional vector space over the field of three elements:

Question 2. *What is the maximum size of a d-cap in $\mathbb{F}_3^d$?*

2.3. Caps of the same type

A natural question is if maximum d-caps we find are unique. The answer is no. To demonstrate this consider the cap in Figure 2.2(a). Permuting the colours obtains a cap of the same type, given in Figure 2.2(b).

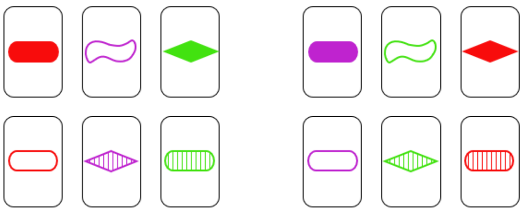


Figure 2.2: (a) A cap. (b) A cap with permuted colours.

Besides permuting the colours there are many more permutations possible to produce new caps of same size as the original one. Stated as in [3]: permutations of $\mathbb{F}_3^d$ that take caps to caps are exactly those that take lines to lines and are called *affine transformations*. These are of the form $\sigma(\vec{v}) = A\vec{v} + \vec{b}$, where A is an invertible $d \times d$-matrix with entries in $\mathbb{F}_3$ and $\vec{v}, \vec{b} \in \mathbb{F}_3^d$. We say that two caps are of the same type if there is an affine transformation taking one to the other. Note that since A is invertible, the operation is reversible, so we define an equivalence relation.

The set of $d \times d$ invertible matrices with entries in $\mathbb{F}_3$ is a group ($GL_d(\mathbb{F}_3)$) that is generated by the elementary matrices. That means that A in $\sigma(\vec{v})$ is a product of elementary matrices. An elementary matrix is a matrix which differs from the identity matrix by one single elementary row operation. The elementary row operations are switching rows, taking nonzero scalar multiples of rows, and adding (multiples of) rows to rows. These operations preserve invertibility. We show next that these operations also take lines to lines:

Let $x, y, z \in \mathbb{F}_3^d$. We call the entries in the same position a row, e.g. $\{x_i, y_i, z_i\}$ is the i'th row. Suppose that x, y and z are a line, i.e. $x + y + z = 0$.

- Switching rows.
  For all coordinates i we have $x_i + y_i + z_i = 0$, so the rows can be reordered.
- Scalar multiples.
  Let $a \in \mathbb{F}_3$. Multiplying row i with a gives
  $ax_i + ay_i + az_i = a(x_i + y_i + z_i) = a0 = 0$.
- Adding (multiples of) rows to rows.
  Let $a \in \mathbb{F}_3$. Suppose we add a times row i to row j. Then row j becomes
  $(x_j + ax_i) + (y_j + ay_i) + (z_j + az_i) = (x_j + y_j + z_j) + a(x_i + y_i + z_i) = 0 + a0 = 0$.

Translations are given by $\vec{b}$ in $\sigma(\vec{v})$ and also take lines to lines:

- Translations.
  Let $a \in \mathbb{F}_3$. Translating row i with a gives
  $(x_i + a) + (y_i + a) + (z_i + a) = (x_i + y_i + z_i) + 3a = 0$.

Note that this section demonstrates why the previously defined d-dimensional *vector space* over the field of three elements $\mathbb{F}_3^d$ is in fact an *affine* space, as any point can be mapped to the zero vector by an affine transformation, so there is no distinguished point that serves as an origin.

2.4. Subspaces of $\mathbb{F}_3^d$

This section treats subspaces of $\mathbb{F}_3^d$ and some implications which will be used in proving the maximum sizes of caps in dimensions 2 and 3 in Section 3.1.

Definition 2.4. Subspace.
A subset S of a vector space V is called a subspace *if it is a vector space itself with the same addition and scalar multiplication as V. That is, if $x, y \in S$ and $\alpha, \beta \in \mathbb{F}_3$, then $\alpha x + \beta y \in S$.*

Definition 2.5. Basis.
A set of linearly independent vectors B is called a basis *for a vector space V if every vector in V is a linear combination of elements in B, i.e. B spans V.*

Definition 2.6. Dimension of a subspace.
The dimension *of a subspace S of $\mathbb{F}_3^d$ equals the number of elements in a basis B for S.*

- a 0-dimensional subspace is called a *point*.
- a 1-dimensional subspace is called a *line*.
- a 2-dimensional subspace is called a *plane*.
- a $(d-1)$-dimensional subspace of $\mathbb{F}_3^d$ is called a *hyperplane*.

Consider a 1-dimensional subspace S, with basis $B = \{a\}$, where $a \in \mathbb{F}_3^d$. Since S is closed under scalar multiplication, $0a = 0 \in S$ and $2a \in S$, so each 1-dimensional subspace of $\mathbb{F}_3^d$ contains two non-zero vectors. That means that the number of lines through 'the origin' is given by:

$$\frac{3^d - 1}{2} \tag{2.1}$$

Since there are 3^d vectors in total, one is zero, and each line contains two non-zero vectors. Recall, however, that in an affine space we "forget" the special role played by the zero vector: any $a \in \mathbb{F}_3^d$ can be mapped to 0 with an affine transformation, so we can in fact say for any point in $\mathbb{F}_3^d$ that the number of lines containing that point equals (2.1). Another way to look at this is by considering the fact that any pair of points determines a unique third point to make a line. Thus for any point a there are $3^d - 1$ points in $\mathbb{F}_3^d$ unequal to a, and half of these determine the lines through a. Figure 2.3 shows two examples of how for $d = 2$ each point is contained in exactly 4 lines. In Section 3.1 we will use this fact in the proof for the maximum size of a 2-cap.

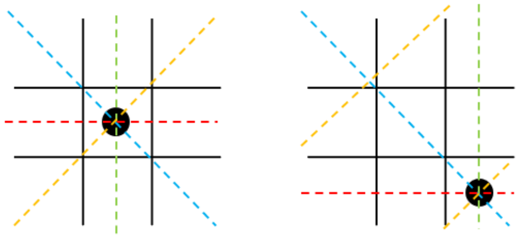


Figure 2.3: Every point in $\mathbb{F}_3^2$ lies on exactly four lines.

Since $\mathbb{F}_3^d$ contains 3^d points, the number of lines through each point equals (2.1), and each line contains three points (so each line in $\mathbb{F}_3^d$ is counted three times), the total number of lines in $\mathbb{F}_3^d$ is given by:

$$3^{d-1} \cdot \frac{3^d - 1}{2} \tag{2.2}$$

Formula (2.2) gives the total number of possible SETs in a d-dimensional SET game.

Each pair of points $a, b \in \mathbb{F}_3^3$ is contained in exactly four planes. To see this, remember that a pair of points determines a line and observe that non-parallel planes intersect at a line (see Figure 2.4). The line determined by a and b contains three points, so there are $3^3 - 3 = 24$ points in $\mathbb{F}_3^3$ apart from the line. On the other hand, each plane has $9 - 3 = 6$ points not on the line. That means that each line is contained in $\frac{24}{6} = 4$ planes. In Section 3.1 we will use this fact in the proof for the maximum size of a 3-cap.

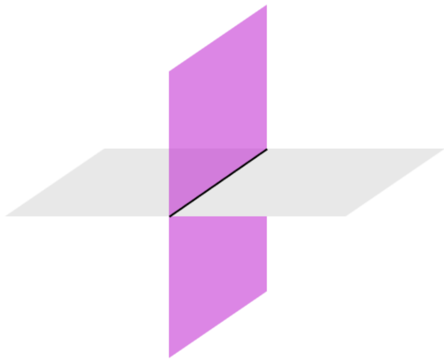


Figure 2.4: Non-parallell planes intersect at a line.

3

Maximum caps

This chapter gives an overview of maximum caps in low dimensions: for dimensions 1 through 6 the maximum sizes are exactly known, and for dimension 7 the largest known cap is given. Then we elaborate on finding maximum caps in high dimensions, where the cap set problem amounts to finding an asymptotic lower bound on the size of caps.

3.1. Maximum caps in low dimensions

The cap set problem tries to find the *maximum* size of caps in fixed dimensions. That means that when we find a maximum cap in some dimension d, no other element of $\mathbb{F}_3^d$ can be added to the cap and there is no way to construct another cap in $\mathbb{F}_3^d$ of bigger size. Let a_d denote the maximum size of a cap in dimension d. A trivial lower bound for a_d is the size of any d-cap we find: surely the *maximum* size of a d-cap will be at least the size of the cap we have found.

Next, we will look at the maximum size of caps in low dimensions. Up until $d = 4$ we can view $\mathbb{F}_3^d$ as cards of SET. For example, for $d = 3$ you can imagine playing an simpler version of SET by playing only with cards that picture one shape. In the next sections we start with one feature and then add features one by one.

Dimension 1: $\mathbb{F}_3^1 = \{0, 1, 2\}$

Suppose we play SET with only one feature, say shape. We then have only three cards, pictured in Figure 3.1. Note that the three cards form a SET: they are all different in the feature 'shape'. In the one dimensional SET game it is trivial to see what the maximum size of a cap is: any subset of two elements is a cap, but a 1-cap cannot contain all three elements of $\mathbb{F}_3$, so $a_1 = 2$.

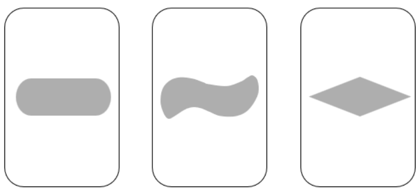


Figure 3.1: The cards of a one dimensional SET game.

Dimension 2: $\mathbb{F}_3^2 = \{0, 1, 2\}^2$

Let us now consider a two dimensional SET game by adding a feature, say colour. This results in nine cards, pictured in Figure 3.2. Taking the four corner cards gives us a 2-cap: see for yourself that these four cards do not contain a SET. By the existence of a 2-cap of size four, we know $a_2 \geq 4$. Now the next question is if it is possible to construct a 2-cap of size 5. Proposition 3.1 states that no such cap exists.

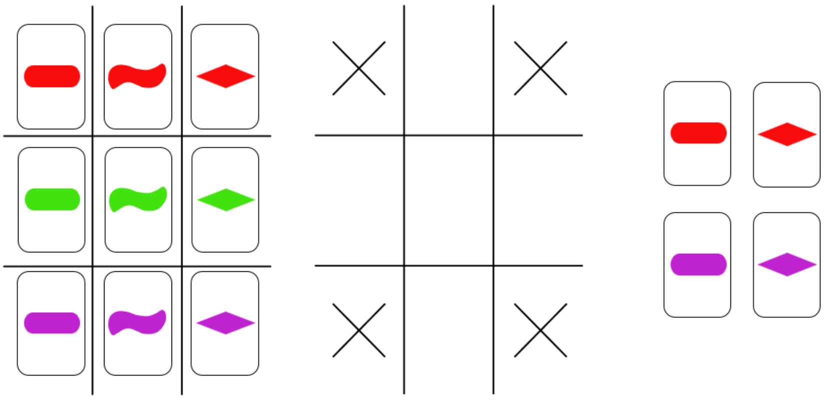


Figure 3.2: The cards of a two dimensional SET game, a 2-cap schematically, and four cards of a 2-cap.

Proposition 3.1. *A maximum 2-cap has size four [3].*

Proof. Assume to a contradiction that there is a 2-cap with five points. The plane $\mathbb{F}_3^2$ can be decomposed as the union of three parallel lines, as in Figure 3.3(a), and each of these lines can contain maximum two points. Then one of these lines, say H, contains only one point, x_5, of the cap. Any point of $\mathbb{F}_3^2$ lies on exactly four lines that cover the plane together, thus apart from H, there are exactly three other lines that contain x_5: L_1, L_2, and L_3, see Figure 3.3(b). By the pigeonhole principle[1] one L_i has to contain two points of the cap other than x_5. That means that there is a line with three points of the cap, contradicting that the five points are a 2-cap. We conclude that $a_2 = 4$. □

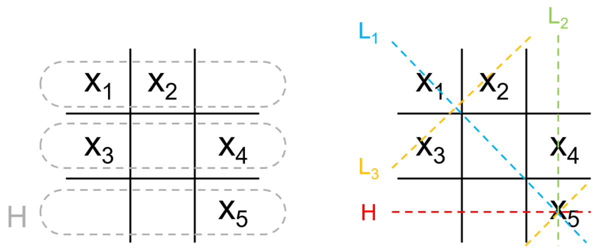


Figure 3.3: (a) Three parallel lines in $\mathbb{F}_3^2$. (b) The lines containing x_5.

Dimension 3: $\mathbb{F}_3^3 = \{0, 1, 2\}^3$

Let us now consider a three dimensional SET game by adding another feature, say shading. This results in 27, pictured in Figure 3.4, together with a 3-cap of size 9 schematically. We see that $a_3 \geq 9$. In fact, by Proposition 3.2, $a_3 = 9$.

Proposition 3.2. *A maximum 3-cap has size nine [3].*

Proof. The proof is similar to the proof of Proposition 3.1. Assume to a contradiction that there is a 3-cap with ten points. The space $\mathbb{F}_3^3$ can be decomposed into three parallel planes. The plane containing the fewest number of points of the cap, say H, has at least two points, say a and b, and at most three points, leaving at least seven points that are not on H. Now a and b are contained in exactly three *other* planes than H. The seven points that are *not* on H need to be distributed over these three planes, so one of the planes that contains a and b must also contain three of the remaining points. This means we find a plane with five points of the cap: a contradiction with $a_2 = 4$. □

For a more detailed proof the reader is referred to [3].

[1] Also known as Dirichlet's box principle: if n items are put into m boxes, with $n > m$, then at least one box must contain more than one item.

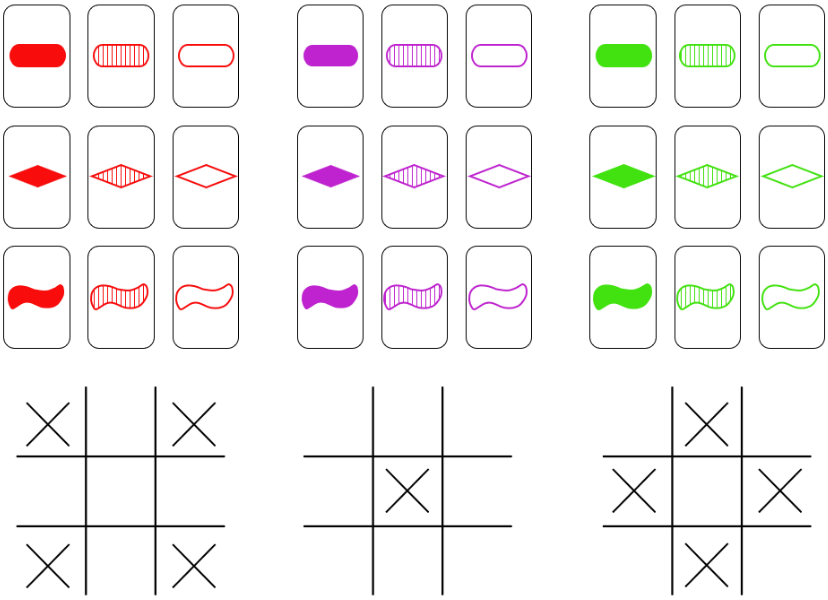


Figure 3.4: The cards of a three dimensional SET game and a 3-cap schematically.

Dimension 4 and 5

Now suppose we add another feature (number of shapes), with which we obtain the complete card game of SET. We then have 81 cards, representing the elements of $\mathbb{F}_3^4 = \{0, 1, 2\}^4$. A 4-cap of size 20 is shown schematically in Figure 3.5. By the existence of a 4-cap of size 20, we know $a_4 \geq 20$. Pellegrino proved in 1971 already that $a_4 < 21$ in [4], but a nice proof is also given in [3], which uses the method of counting marked hyperplanes via hyperplane triples.

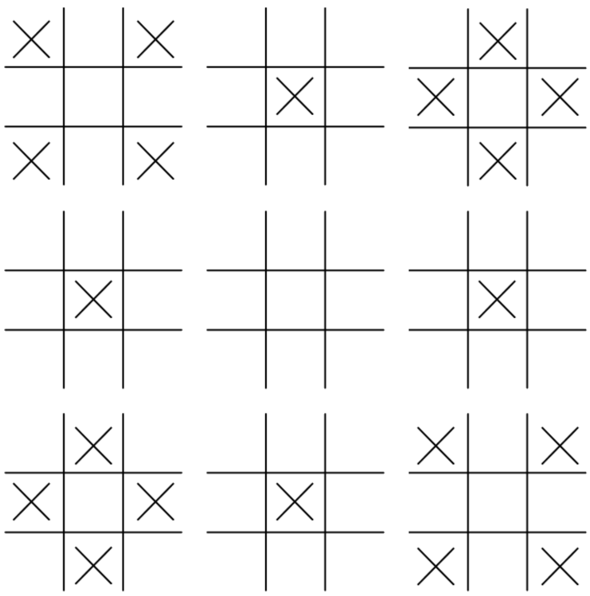


Figure 3.5: A 4-cap of size 20 schematically.

Now suppose that the cards of SET have an extra feature. Imagine, for example, that the cards have a furry, sticky or smooth texture. We then obtain a five dimensional SET game, where we 'play' with the elements of $\mathbb{F}_3^5 = \{0, 1, 2\}^5$. A 5-cap of size 45 is known to exist. In 2002 Edel, Ferret, Landjev, and Storme proved that a cap of size 46 does not exist in [5], making use of an exhaustive computer search. For a summary of the proof the reader is referred to [3]. It has to be mentioned that the proof uses concepts that are not defined yet and will not be defined in this report, such as hyperplanes and the Fourier analysis bound.

3.2. The Hill cap and the Calderbank-Fishburn cap

In this section the Hill cap is explained, which gives a 6-cap of size 112. Potechin proved in 2008 that this is the maximum size of a cap in $\mathbb{F}_3^6$ in [6]. The Calderbank-Fishburn cap, which is based on the Hill cap, gives a 7-cap of size 236, the biggest cap known in dimension 7.

The Hill cap is based on a 2-$(6,3,2)$ block design D, given in Table 3.1. The blocks represent which positions in a vector $\vec{v} \in \mathbb{F}_3^6$ are non-zero. Thus each block contains 2^3 different vectors. We call the positions where a vector is non-zero the *support* of the vector. We call the number of positions where a vector is non-zero the *weight* of a vector.

Table 3.1: The blocks of D: 2-$(6,3,2)$

123	236
124	245
135	256
146	345
156	346

The following comments give some further explanation on the parameters in the notation '2-(v,k,λ)', where $v = 6$, $k = 3$, and $\lambda = 2$, and the corresponding properties of the block design D:

- A vector $\vec{v} \in \mathbb{F}_3^6$ has 6 potential positions where the entry could be non-zero. Thus there are $v = 6$ elements that the blocks can contain: $\{1,2,3,4,5,6\}$.
- The size of each block is $k = 3$, meaning that each vector $\vec{v} \in \mathbb{F}_3^6$ has weight 3.
- Each pair of positions occurs in exactly $\lambda = 2$ blocks, where the fact that we are considering pairs is indicated by the first 2. For example the pair 12 is contained in exactly two blocks: 123 and 124. A result of this property is that whenever two blocks have two points in common, there is no third block contained in their union, i.e. all other blocks have at least one non-zero coordinate where the two original blocks have zeros.

To summarize, D is a set of vectors $\in \mathbb{F}_3^6$ such that each vector has weight 3 and support given by one of the blocks of D. There are $10 \cdot 2^3 = 80$ vectors contained in D.

Let R be the vectors of $\mathbb{F}_3^6$ with weight 6 and an even number of 2's (and thus also an even number of 1's). Then R contains 32 vectors and the union of R with D gives a cap:

Proposition 3.3. The Hill cap.
$D \cup R$ is a cap of size 112 in $\mathbb{F}_3^6$.

Proof. The fact that D is a cap follows from the properties of a block design. R is a cap since for any $x, y, z \in R$ there is a coordinate where $\{x_i, y_i, z_i\} = \{1,2\}$ (since all vectors in R have weight 6).

To check if $D \cup R$ is a cap, we need to check two cases:

1. $x, y \in D$ and $z \in R$:
   First observe that each pair of vectors $x, y \in D$ has overlapping support in one or two positions. Since each pair has at least one common non-zero position and the vectors have weight 3 and dimension 6, there is a coordinate i where $x_i = y_i = 0$. Since $z \in R$ has weight 6, we find $(x_i, y_i, z_i) = (0, 0, *)$, where $*$ is a non-zero entry. This is visualised in Figure 3.6.
2. $x \in D$ and $y, z \in R$:
   Since $y + z$ has even weight for any $y, z \in R$ and $-x$ has weight 3, we cannot have $y + z = -x$ and therefore the sum of x, y and z cannot be equal to 0.

We conclude that in any case $x + y + z \neq 0$. Thus $D \cup R$ is a cap of size $|D| + |R| = 80 + 32 = 112$. □

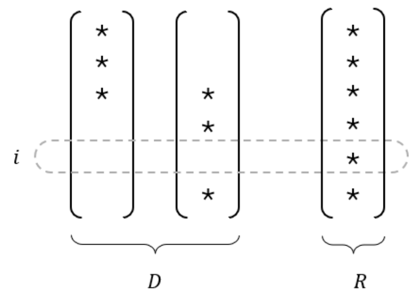


Figure 3.6: D has at least one overlapping non-zero entry and R has no zeros.

Let $\bar{D}$ be the complement of D, where $123^C = 456$, for example. Then $\bar{D}$ contains the weight 3 vectors of $\mathbb{F}_3^6$ with supports that are not given in D. Note that there are $\binom{6}{3} \cdot 2^3 = 160$ vectors of weight 3 in total, so D and $\bar{D}$ partition all weight 3 vectors of $\mathbb{F}_3^6$ into two equal parts. $\bar{D} \cup R$ is an isomorphic copy of $D \cup R$ and is also a cap.

Now let U be all 12 weight 1 vectors of $\mathbb{F}_3^6$. U is a cap since if we take three distinct elements $x, y, z \in U$, in any case there will be a coordinate i where $\{x_i, y_i, z_i\}$ has two 0's and one non-zero entry, so $x+y+z \neq 0$. We can combine D, $\bar{D}$, R and U into a cap in $\mathbb{F}_3^7$, where we use Notation 1:

Notation 1. *Let $(a, X) \subset \mathbb{F}_3^{d+1}$ denote $\{a\} \times X$ for $a \in \mathbb{F}_3$ and $X \subset \mathbb{F}_3^d$.*

Proposition 3.4. The Calderbank-Fishburn cap.
$(0, D) \cup (0, R) \cup (1, \bar{D}) \cup (1, R) \cup (2, U)$ *is a cap of size* 236 *in* $\mathbb{F}_3^7$.

Proof. Suppose we have x, y, z in the collection with $x + y + z = 0$. Then the first coordinate has to add up to 0, so the prefixes are either all the same or all different. Since $D \cup R$, $\bar{D} \cup R$, and U are caps, the prefixes can not all be 0, all be 1, or all be 2, so the prefixes are $\{0, 1, 2\}$. Let $z \in (2, U)$. Then z has weight 2 and $x + y$ must also have weight 2 for $x + y + z = 0$, but in all cases $x + y$ has weight unequal to 2:

1. $x \in (0, D)$ and $y \in (1, \bar{D})$:
   Since no element of D has the same support as an element of $\bar{D}$, vectors in $D + \bar{D}$ have weight ≥ 2, so $x + y$ has weight ≥ 3.
2. $x \in (0, D)$ and $y \in (1, R)$ or $x \in (0, R)$ and $y \in (1, \bar{D})$:
   Vectors in $D + R$ or $\bar{D} + R$ have weight ≥ 3, so $x + y$ has weight ≥ 4.
3. $x \in (0, R)$ and $y \in (1, R)$:
   Vectors in $R + R$ have even weight, so $x + y$ has odd weight and is thus unequal to 2.

We conclude that there exist no x, y, z such that $x + y + z = 0$. □

3.3. Asymptotic lower bound

So far we have seen that we already know a lot about maximum caps in low dimensions. But what can be said about maximum caps in high dimensions? Maximum caps in low dimensions can be combined into higher dimensional caps by the direct product construction, which is stated in Proposition 3.5.

Proposition 3.5. The Direct Product Construction.
If $A \subset \mathbb{F}_3^n$ and $B \subset \mathbb{F}_3^m$ are caps, then $A \times B = \{(a, b) : a \in A, b \in B\} \subset \mathbb{F}_3^{n+m}$ is a cap of size $|A||B|$.

Proof. Suppose there are $x = (x_a, x_b), y = (y_a, y_b), z = (z_a, z_b) \in A \times B$, with $x_a, y_a, z_a \in A$ and $x_b, y_b, z_b \in B$ such that $x + y + z = 0$. Then we must have $x_a + y_a + z_a = 0$ and $x_b + y_b + z_b = 0$, contradicting that A and B are caps. □

By taking the direct product of $\{0, 1\}$ (which is a cap in $\mathbb{F}_3$), we see that $\{0, 1\}^d \subset \mathbb{F}_3^d$ is a cap for every dimension d, so we know that the size of a maximum cap in any dimension is at least 2^d. Writing this differently gives $2 \leq \sqrt[d]{a_d}$. This brings us to the concept of solidity, which is used to compare the "largeness" of caps from different dimensions.

Definition 3.6. Solidity.
Let A be a cap in $\mathbb{F}_3^d$. The solidity *of A is defined as $c(A) = \sqrt[d]{|A|}$.*

Now, the question is what happens to $c(A)$ as we take maximum caps in dimension $d \to \infty$. Thus, we are interested in lower (and upper) bounds of asymptotic solidity:

Definition 3.7. Asymptotic Solidity.
The asymptotic solidity of maximum caps is defined as $c = \sup_d \sqrt[d]{a_d}$, where a_d denotes the maximum cap size in dimension d.

Since any cap contains less than 3^d points, we know that $c \leq 3$. In fact, in 2017 Ellenberg and Gijswijt proved that $c \leq 2.756$ in [7] by adapting a proof of Croot, Lev, and Pach for a related problem in [8].

Since the asymptotic solidity is at least the solidity of any cap, c is at least 2. The following theorem shows how the direct product construction can be used to derive an asymptotic lower bound for the cap set problem:

Theorem 3.8. *Let $A \subset \mathbb{F}_3^n$ be a cap of size c^n. Then for any $\epsilon > 0$ there is an M such that for all $m \geq M$ there is a cap in $\mathbb{F}_3^m$ of size greater than $(c-\epsilon)^m$.*

Proof. Suppose that A is a cap in $\mathbb{F}_3^n$ of size c^n, thus $\sqrt[n]{|A|} = c$.

For any m we can write $m = nk + r$, where $0 \leq r < n$. Since $r < n$, for any $m \geq M$ we have $\frac{r}{m} < \frac{n}{M}$, so $1 - \frac{n}{M} < 1 - \frac{r}{m}$.

Let $\epsilon > 0$. We can always choose M large enough such that $c^{1-\frac{n}{M}} > c - \epsilon$. To see this note that as $M \to \infty$, $1 - \frac{n}{M}$ will approach 1, so $c^{1-\frac{n}{M}}$ will approach c. By applying the direct product construction k times to A we find a cap in $\mathbb{F}_3^{nk}$ with:

$$|\underbrace{A \times \cdots \times A}_{k\ times}| = c^{nk} = c^{m-r} = (c^{1-\frac{r}{m}})^m > (c^{1-\frac{n}{M}})^m > (c-\epsilon)^m$$

Since $nk \leq m$, $\underbrace{A \times \cdots \times A}_{k\ times}$ is a cap in $\mathbb{F}_3^m$ of size greater than $(c-\epsilon)^m$. □

Theorem 3.8 shows that finding an asymptotic lower bound comes down to finding a cap $A \subset \mathbb{F}_3^d$ where $\sqrt[d]{|A|}$ is as large as possible. By taking the maximum cap of size 112 in $\mathbb{F}_3^6$ we see that $c \geq \sqrt[6]{112} = 2.196$. Note that the asymptotic lower bound allows us to draw conclusions about maximum cap sizes in *high* dimensions, so for large enough d. To demonstrate what happens for a dimension that is too low, consider that in two dimensions we would conclude $a_2 \geq (2.196)^2 > 4$, while we know that $a_2 = 4$.

3.4. Conclusion

In summary, the cap set problem tries to find the biggest possible subset of $\mathbb{F}_3^d$ not containing lines. For low dimensions much is already known about the maximum sizes a_d of caps. Table 3.2 gives an overview of the exactly known maximum sizes a_d in dimensions 1 up until 6, and the size of the biggest known cap in dimension 7.

Table 3.2: Sizes of maximum caps in dimensions 1 up to 6 and the best lower bound in dimension 7.

d	1	2	3	4	5	6	7
a_d	2	4	9	20	45	112	≥ 236

For large enough d, we have found the lower bound c^d on the maximum size of a cap, where c denotes the asymptotic solidity from Definition 3.7. The 6-cap of size 112 gives an asymptotic lower bound of 2.196. The aim is now to construct a cap $A \subset \mathbb{F}_3^d$ such that $\sqrt[d]{|A|}$ is as large as possible.

4

The extended product construction

In this chapter the extended product construction will be explained. This construction takes maximum caps in lower dimensions and combines them carefully to create large caps in higher dimensions.

4.1. The extended product construction

In this section an explanation of the extended product construction (due to Edel) is given, based on [9]. The extended product construction relies on the concepts ‘extendable collections of caps’ and ‘admissible sets’.

The basic idea of the extended product construction is applying an m-dimensional *admissible* set $S \subset \{0,1,2\}^m$ to an *extendable collection* of n-caps $A_0, A_1, A_2 \subset \mathbb{F}_3^n$ to obtain a new cap in $\mathbb{F}_3^{nm}$. If S is not only admissible, but also *recursive*, we even find a new extendable collection of caps in $\mathbb{F}_3^{nm}$, to which we can again apply an admissible set to obtain a higher-dimensional cap.

Now, let us go into detail. Suppose we have three sets $A_0, A_1, A_2 \subset \mathbb{F}_3^n$. These sets form an extendable collection of caps if they are all caps and if they satisfy the two conditions in Definition 4.1, to be later referred to as ‘Ext1’ and ‘Ext2’.

Definition 4.1. Extendable collection of caps
A collection of caps $A_0, A_1, A_2 \subset \mathbb{F}_3^n$ is said to be extendable *if the following two conditions hold:*

1. *If $x, y \in A_0$ and $z \in A_1 \cup A_2$ then $x + y + z \neq 0$*
2. *If $x \in A_0, y \in A_1$ and $z \in A_2$ then $x + y + z \neq 0$*

To get a good grasp on Definition 4.1, there are a few things to note about it:

- Taking $x = y$ in Ext1 shows that A_0 is disjoint from A_1 and A_2.
- Each pair $x, y \in \mathbb{F}_3^n$ determines a point $z \in \mathbb{F}_3^n$ such that $x + y + z = 0$, so in order to satisfy Ext1 there is a “forbidden” point in $A_1 \cup A_2$ for each pair of points $x, y \in A_0$. Therefore A_0 should be relatively small.
- $A_0 \cup (A_1 \cap A_2)$ is a cap:
  Taking $x, y \in A_0$ and $z \in A_1 \cap A_2$ in Ext1 and $x \in A_0$ and $y, z \in A_1 \cap A_2$ in Ext2 both gives us $x + y + z \neq 0$. Moreover A_0 and $A_1 \cap A_2$ contain no lines themselves since they are caps. This shows that $A_0 \cap (A_1 \cup A_2)$ contains no lines and is thus a cap.

Before we continue to defining admissible sets we introduce some notation.

Notation 2. *Let $*$ denote a non-zero element, i.e. $* \in \{1,2\}$.*

We refer to the i’th entries of a set of vectors as a row. For example, $\{x_i, y_i, z_i\}$ is the i’th row of the triple x, y, z. In the definition for admissible sets we will define a condition that asks every triple in the set to have a row that either has $\{s'_k, s''_k, s''_k\} = \{0, 1, 2\}$, or $\{s'_k, s''_k, s''_k\} = \{0, *\}$ where the row has two 0's and one $*$. For convenience, we define two sets of ordered triples:

Notation 3. *Let* N_{012} *denote the set of ordered triples containing every element of* $\mathbb{F}_3$ *exactly once:*

$$\mathrm{N}_{012} = \{(0, 1, 2), (0, 2, 1), (1, 0, 2), (1, 2, 0), (2, 0, 1), (2, 1, 0)\}$$

Notation 4. *Let* M_{00*} *denote the set of ordered triples containing two* 0*'s and one non-zero element:*

$$\mathrm{M}_{00*} = \{(*, 0, 0), (0, *, 0), (0, 0, *)\}$$

Now, we are ready to define an admissible set, which is a subset $S \subset \{0, 1, 2\}^m$ that satisfies the conditions in Definition 4.2, to be later referred to as 'Adm1' and 'Adm2'.

Definition 4.2. Admissible set
A set $S \subset \{0, 1, 2\}^m$ *is said to be* admissible *if the following two conditions hold:*

1. *For all distinct* $s, s' \in S$*, there are coordinates* i *and* j *such that* $s_i = 0 \neq s'_i$ *and* $s_j \neq 0 = s'_j$*.*
2. *For all distinct* $s, s', s'' \in S$*, there is a coordinate* k *such that* $(s'_k, s''_k, s''_k) \in \mathrm{N}_{012} \cup \mathrm{M}_{00*}$*.*

There are some special types of admissible sets, such as *recursive* or *constant weight* sets. In Section 4.3 these types will be defined and in Chapter 6 some explicit constructions and examples of these admissible sets are given.

Now that we have defined extendable collections of caps and admissible sets, we need to define an operation that links these concepts:

Let A_0, A_1, A_2 be sets in $\mathbb{F}_3^n$. An element $s = (s_1, \ldots, s_m)$ is applied to these sets by the following operation:

$$s(A_0, A_1, A_2) = A_{s_1} \times \cdots \times A_{s_m} \subseteq \mathbb{F}_3^{nm} \tag{4.1}$$

Now, using this operation we can finally define the extended product construction:

Theorem 4.3. The extended product construction.
If $A_0, A_1, A_2 \subset \mathbb{F}_3^n$ *is an extendable collection of caps and* $S \subset \{0, 1, 2\}^m$ *is an admissible set, and we take the union of all the elements of* S *applied to the extendable collection, as in equation (4.2), then the result* $S(A_0, A_1, A_2)$ *is a cap in* $\mathbb{F}_3^{nm}$*.*

$$S(A_0, A_1, A_2) = \bigcup_{s \in S} s(A_0, A_1, A_2) \tag{4.2}$$

A condensed version of the proof is given here, but for a more detailed proof the reader is referred to Tyrrell [9].

Proof. It needs to be checked that for all distinct $x, y, z \in S(A_0, A_1, A_2)$ we must have $x + y + z \neq 0$. There are three cases where x, y, z could come from:

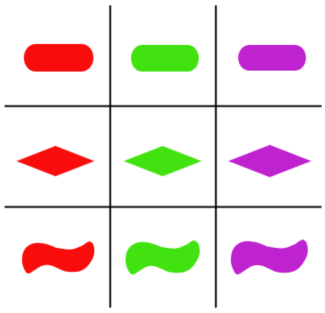
1. x, y, z all come from the same vector s.
   Since each $s(A_0, A_1, A_2)$ is a cap by the direct product construction, no lines would be formed.
2. x and y come from s and z comes from s' where s and s' are distinct vectors.
   By Adm1 there is a coordinate where $s_j = 0 \neq s'_j$, thus we would find $x_{s_j}, y_{s_j} \in A_0$ and $z_{s'_j} \in A_1 \cup A_2$, so by Ext1 $x + y + z \neq 0$.
3. x, y and z all come from distinct vectors s, s' and s''.
   By Adm2 we find $(s_k, s'_k, s''_k) \in \mathrm{M}_{00*}$, and then the argument for case 2 again applies, or $(s_k, s'_k, s''_k) \in \mathrm{N}_{012}$, and then by Ext2 $x + y + z \neq 0$.

Thus since $S(A_0, A_1, A_2)$ contains no lines, it is a cap. $\square$

4.2. An example of the extended product construction

In this section an example is covered to demonstrate the extended product construction.

Notation 5. The elements of $\mathbb{F}_3^2$ in a grid.
We schematically represent the elements of $\mathbb{F}_3^2$ in a two-dimensional grid. The columns are numbered left to right and give the first coordinate. The rows are numbered top to bottom and give the second coordinate.



$\begin{bmatrix}0\\0\end{bmatrix}$	$\begin{bmatrix}1\\0\end{bmatrix}$	$\begin{bmatrix}2\\0\end{bmatrix}$
$\begin{bmatrix}0\\1\end{bmatrix}$	$\begin{bmatrix}1\\1\end{bmatrix}$	$\begin{bmatrix}2\\1\end{bmatrix}$
$\begin{bmatrix}0\\2\end{bmatrix}$	$\begin{bmatrix}1\\2\end{bmatrix}$	$\begin{bmatrix}2\\2\end{bmatrix}$

Figure 4.1: The cards of a two-dimensional SET game and their corresponding vectors represented in a grid.

Consider the collection of caps $A_0, A_1, A_2 \subset \mathbb{F}_3^2$:

$$A_0 = \left\{\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1\\0\end{bmatrix}\right\}, \ A_1 = \left\{\begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}0\\2\end{bmatrix}\right\}, \ A_2 = \left\{\begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}1\\2\end{bmatrix}\right\} \tag{4.3}$$

It is easy to check that this is an extendable collection in $\mathbb{F}_3^2$ by drawing the caps according to Notation 5:

$$\begin{array}{c|c|c} A_0 & A_0 & \times \\ \hline A_1 & A_2 & \\ \hline A_1 & A_2 & \end{array} \tag{4.4}$$

Clearly A_0 is disjoint from $A_1 \cup A_2$. If we take the two different points in A_0 we would form a line with $(2,0)^T$, so this is a 'forbidden point'. We see that the collection of caps satisfies Ext1. Note that there are also no lines to be made with $x \in A_0, y \in A_1$ and $z \in A_2$, so the collection satisfies Ext2.

Now, consider the following set:

$$S = \left\{\begin{bmatrix}0\\2\end{bmatrix}, \begin{bmatrix}1\\0\end{bmatrix}\right\}$$

This set is admissible: there are no distinct triples and it is trivial to see that the only distinct pair satisfies Adm1. Let us apply S to the extendable collection. We then find a new cap in dimension $nm = 4$, which is explicitly given in equation (4.5).

$$\begin{aligned} S(A_0, A_1, A_2) &= (A_0 \times A_2) \cup (A_1 \times A_0) \\ &= \left\{\begin{bmatrix}0\\0\\1\\1\end{bmatrix}, \begin{bmatrix}0\\0\\1\\2\end{bmatrix}, \begin{bmatrix}1\\0\\1\\1\end{bmatrix}, \begin{bmatrix}1\\0\\1\\2\end{bmatrix}, \begin{bmatrix}0\\1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\1\\0\end{bmatrix}, \begin{bmatrix}0\\2\\0\\0\end{bmatrix}, \begin{bmatrix}0\\2\\1\\0\end{bmatrix}\right\} \end{aligned} \tag{4.5}$$

It can be manually checked that any triple in $S(A_0, A_1, A_2)$ does not add up to 0, since it has a coordinate i where $\{x_i, y_i, z_i\} = \{0,1\}, \{0,2\}$ or $\{1,2\}$. This means that any triple has a 'feature' where the three are not all different nor all the same, thus $S(A_0, A_1, A_2)$ is a 4-cap. It is also possible to run this check by a computer (see Appendix B.1).

4.3. Recursive and constant weight admissible sets

In this section recursive and constant weight admissible sets are defined and their useful results are stated. This section is again based on [9].

Recall the definition of admissible sets (Definition 4.2): a set $S \subset \{0,1,2\}^m$ is said to be admissible if all distinct pairs satisfy Adm1, i.e. we find $s_i = 0 \neq s'_i$ and $s_j \neq 0 = s'_j$, and if all distinct triples satisfy Adm2, i.e. we find $(s_k, s'_k, s''_k) \in \mathrm{N}_{012} \cup \mathrm{M}_{00*}$. An admissible set is called recursive if all distinct pairs also satisfy one of the two conditions in Definition 4.4, to be later referred to as 'Rec1' and 'Rec2'.

Definition 4.4. Recursively admissible set
S is a recursively admissible set *if S is an admissible set (Definition 4.2), $|S| \geq 2$, and for all distinct pairs $s, s' \in S$ at least one of the following holds:*

1. *There are coordinates i and j such that $\{s_i, s'_i\} = \{0,1\}$ and $\{s_j, s'_j\} = \{0,2\}$.*
2. *There is a coordinate k such that $s_k = s'_k = 0$.*

Note that each distinct pair in S needs to satisfy Rec1 *or* Rec2, thus it is sufficient if one of the two conditions holds for each pair. Moreover, not all pairs need to satisfy the same condition: some pairs may satisfy Rec1 while others may satisfy Rec2.

Using a recursively admissible set in the extended product construction yields a nice result, stated in Lemma 4.5.

Lemma 4.5. *If A_0, A_1, A_2 is an extendable collection of caps, and $S \subseteq \{0,1,2\}^m$ is a recursively admissible set, then $(S(A_0, A_1, A_2), A_1^m, A_2^m)$ is an extendable collection of caps.*

This result means that by applying an m-dimensional recursively admissible set to an extendable collection of caps in $\mathbb{F}_3^n$, we find a new extendable collection of caps in $\mathbb{F}_3^{nm}$, to which we can again apply an admissible set.

Let the weight of a vector be the number of non-zero entries it has. If a set of vectors all have the same weight, it is said to be constant weight, as is defined in Definition 4.6.

Definition 4.6. Constant weight admissible set
$S \subseteq \{0,1,2\}^m$ is a constant weight admissible set *if it is an admissible set consisting of vectors all of the same weight w. An admissible set cannot have two distinct elements with the same support, so $|S| \leq \binom{m}{w}$. We write:*

- *$S = I(m,w)$ if the set is full size: $|S| = \binom{m}{w}$.*
- *$S = \tilde{I}(m,w)$ if in addition S is recursively admissible.*
- *$S = A(m,w)$ if S is admissible and constant weight, but not full size: $|S| < \binom{m}{w}$.*

When constant weight admissible sets are applied to an extendable collection we find the nice property that for each $s \in S \subset \{0,1,2\}^m$ with weight w we have that $s(A_0, A_1, A_2)$ consists of $m - w$ "blocks" of A_0 and w "blocks" of A_1 or A_2. That means that if $|A_1| = |A_2|$, there is an explicit formula for the size of $S(A_0, A_1, A_2)$, presented in Lemma 4.7.

Lemma 4.7. *If we extend an extendable collection of caps A_0, A_1, A_2 by $S = I(m,w) \subseteq \{0,1,2\}^m$, where $|A_1| = |A_2|$, then*

$$|S(A_0, A_1, A_2)| = \binom{m}{w} |A_0|^{m-w} |A_1|^w$$

5

Asymptotic lower bounds

In this chapter an overview is given of how the asymptotic lower bound on the size of cap sets has little by little been improved using the extended product construction. The best asymptotic lower bound yet equals 2.2203 *and has been found with admissible sets that were discovered with the help of a Large Language Model.*

5.1. The Hill cap extendable collection

Recall the 6-cap of size 112 from Section 3.2:

- D are the 80 vectors in $\mathbb{F}_3^6$ of weight 3, with their supports given by the blocks of a 2-$(6, 3, 2)$ block design (see Table 3.1).
- R are the 32 vectors in $\mathbb{F}_3^6$ of weight 6.
- $\bar{D}$ are the 80 vectors in $\mathbb{F}_3^6$ of weight 3, with their supports *not* given in D.
- $D \cup R$ and $\bar{D} \cup R$ are caps of size 112.

Also recall from Section 3.2 that U is a cap, where U are the 12 vectors in $\mathbb{F}_3^6$ of weight 1. It turns out that the three mentioned caps form an extendable collection together:

Lemma 5.1. *[10] Let* $A_0 = U$, $A_1 = D \cup R$, *and* $A_2 = \bar{D} \cup R$. *Then* $A_0, A_1, A_2 \subset \mathbb{F}_3^6$ *is an extendable collection with* $|A_0| = 12$ *and* $|A_1| = |A_2| = 112$.

Proof. (from [9])
Ext1: Let $x, y \in A_0$ and $z \in A_1 \cup A_2$. Then since all vectors in A_0 have weight 1, $x + y$ has weight 0, 1 or 2. But all vectors in $A_1 \cup A_2$ have weight ≥ 3, so $x + y + z \neq 0$.

Ext2: By checking that $D + \bar{D}$, $R + R$, $D + R$, and $\bar{D} + R$ do not contain any weight 1 vectors, it follows that $A_1 + A_2 = \mathbb{F}_3^6 \setminus A_0$. Suppose there are $x \in A_0$, $y \in A_1$, and $z \in A_2$ with $x + y + z = 0$. Then $y + z = 2x \in A_0$, but also $y + z \in A_1 + A_2 = \mathbb{F}_3^6 \setminus A_0$, so we find a contradiction. We conclude that no such x, y, z exist.

We conclude that A_0, A_1, A_2 with $A_0 = U$, $A_1 = D \cup R$, and $A_2 = \bar{D} \cup R$ is an extendable collection in $\mathbb{F}_3^6$. $\square$

Realize that it is quite special that the extendable collection from Lemma 5.1 exists, since A_1 and A_2 are of maximum size in dimension 6. Both Tyrrell and Edel used this extendable collection as starting position in their constructions that improved the asymptotic lower bound.

5.2. Lower bounds by the extended product construction

The lower bounds found by Tyrrell [9], based on the ideas of Edel [10], are obtained by applying the extended product construction twice, using constant weight admissible sets. For the construction, the following result is needed:

For each dimension $k \geq 2$ it is possible to construct a recursively admissible set with constant weight $k-1$, as is stated in Lemma 5.2. The set is constructed by taking the k vectors with a 0 in exactly one position and letting all the entries before the 0 be 1 and all the entries after the 0 be 2 in each vector. If we take for example dimension $k = 4$, the result is the set as shown in expression (5.1).

$$\tilde{I}(4,3) = \left\{ \begin{bmatrix} 0 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \right\} \tag{5.1}$$

Lemma 5.2. *For any $k \geq 2$ there exists a recursively admissible set $\tilde{I}(k, k-1)$.*

For the proof that $\tilde{I}(k, k-1)$ is recursively admissible the reader is referred to [9].

Now let us turn to the construction: in general, we start with some extendable collection $A_0, A_1, A_2 \subset \mathbb{F}_3^n$ with $|A_1| = |A_2|$ and we obtain a new cap in a higher dimension using the extended product construction by the following two steps:

1. We extend the collection with $S = \tilde{I}(k, k-1)$, which we know exists by Lemma 5.2. Then we obtain a cap $B = S(A_0, A_1, A_2) \subset \mathbb{F}_3^{nk}$. Since S is constant weight and $|A_1| = |A_2|$, by Lemma 4.7 we know the size of B:
$$|B| = k|A_0||A_1|^{k-1} \tag{5.2}$$
Since S is recursively admissible, we obtain a new extendable collection $(B, A_1^k, A_2^k) \subset \mathbb{F}_3^{kn}$, where $B = S(A_0, A_1, A_2)$.

2. We extend the resulting collection with $T = I(m, w)$. Then we obtain a cap $\tilde{B} = T(B, A_1^k, A_2^k) \subset \mathbb{F}_3^{knm}$. Since T is constant weight and $|A_1^k| = |A_2^k|$, again by Lemma 4.7 we know the size of $\tilde{B}$:
$$\begin{aligned} |\tilde{B}| &= |T||B|^{m-w}|A_1^k|^w \\ &= \binom{m}{w}(k|A_0||A_1|^{k-1})^{m-w}|A_1|^{kw} \\ &= \binom{m}{w}(k|A_0|)^{m-w}|A_1|^{km-(m-w)} \end{aligned} \tag{5.3}$$

Now, suppose we start with the extendable collection $A_0, A_1, A_2 \subset \mathbb{F}_3^6$ with $|A_0| = 12$ and $|A_1| = |A_2| = 112$ from Section 5.1 to obtain a cap $\tilde{B} \subset \mathbb{F}_3^{6km}$. Filling in the sizes for A_0, A_1, and A_2 in equation (5.3) gives:

$$|\tilde{B}| = \binom{m}{w}(k \cdot 12)^{m-w} \cdot 112^{km-(m-w)} \tag{5.4}$$

The cap $\tilde{B} \subset \mathbb{F}_3^{6km}$ gives an asymptotic lower bound of $|\tilde{B}|^{\frac{1}{6km}}$ on the size of cap sets. Since the goal is to find a lower bound as high as possible, the next step is to find suitable $S = \tilde{I}(k, k-1)$ and $T = (m, w)$ such that $|\tilde{B}|^{\frac{1}{6km}}$ is as large as possible. By Lemma 5.2, we know that $S = \tilde{I}(k, k-1)$ exists for any $k \geq 2$, thus we start with a $T = I(m, w)$ and then optimize k such that $|\tilde{B}|^{\frac{1}{6km}}$ is as large as possible. The following constant weight admissible sets are found with a computer search by Tyrrell:

Lemma 5.3. *[9] There exist admissible sets $I(11,7)$, $I(11,6)$ and $I(10,6)$*[1].

For each option for $T = I(m, w)$ from Lemma 5.3 we use equation (5.4) to find the best $S = I(k, k-1)$ and the resulting lower bound:

[1]The sets can be found on Tyrrell's webpage http://fredtyrrell.com/cap-sets

- $T = I(10,6)$ gives $|\tilde{B}| = \binom{10}{6}(k \cdot 12)^4 \cdot 112^{10k-4}$.
  Then $|\tilde{B}|^{\frac{1}{60k}}$ has an extremum at $k = 6.66$, so we choose $S = \tilde{I}(7,6)$.
  By applying $S = \tilde{I}(7,6)$ and $T = I(10,6)$ we find a lower bound $|\tilde{B}|^{\frac{1}{420}} = 2.21756$.
- $T = I(11,6)$ gives $|\tilde{B}| = \binom{11}{6}(k \cdot 12)^5 \cdot 112^{11k-5}$.
  Then $|\tilde{B}|^{\frac{1}{66k}}$ has an extremum at $k = 7.44$, so we choose $S = \tilde{I}(7,6)$.
  By applying $S = \tilde{I}(7,6)$ and $T = I(11,6)$ we find a lower bound $|\tilde{B}|^{\frac{1}{462}} = 2.21795$.
- $T = I(11,7)$ gives $|\tilde{B}| = \binom{11}{7}(k \cdot 12)^4 \cdot 112^{11k-4}$.
  Then $|\tilde{B}|^{\frac{1}{66k}}$ has an extremum at $k = 5.95$, so we choose $S = \tilde{I}(6,5)$.
  By applying $S = \tilde{I}(6,5)$ and $T = I(11,7)$ we find a lower bound $|\tilde{B}|^{\frac{1}{396}} = 2.21798$.

The best new lower bound found by Tyrrell in [9] is $(2.21798)^n$. Edel used essentially the same method with $S = \tilde{I}(8,7)$ and $T = (10,5)$ to find the lower bound $(2.21739)^n$ in [10].

5.3. Using meta-admissible sets

In [9], Tyrrell improved the asymptotic lower bound found in the previous section even further, by extending Edel's methods. Tyrrell mimicks the extended product construction for caps to find large admissible sets. To do this, extendable collections of admissible sets need to be defined. Then a similar construction to the extended product construction (Theorem 4.3) yields a new admissible set. To prove this we are going to need the following proposition, stated without proof:

Proposition 5.4. *If S and T are admissible sets, then so is their direct product $S \times T$.*

A collection of admissible sets is called meta-extendable if the admissible sets satisfy the three conditions in Definition 5.5, to be later referred to as 'Meta-Ext1', 'Meta-Ext2', and 'Meta-Ext3'.

Definition 5.5. Meta-Extendable.
A collection of admissible sets $(S_0, S_1, S_2) \subset \{0,1,2\}^m$ is said to be meta-extendable *if:*

1. *For any $s \in S_0$ and $s' \in S_1 \cup S_2$ the weight of s is less than the weight of s'.*
2. *If $x, y \in S_0$ and $z \in S_1 \cup S_2$, then there is a coordinate k such that $(x_k, y_k, z_k) \in \mathrm{N}_{012} \cup \mathrm{M}_{00*}$.*
3. *If $x \in S_0$, $y \in S_1$ and $z \in S_2$, then there is a coordinate k such that $(x_k, y_k, z_k) \in \mathrm{N}_{012} \cup \mathrm{M}_{00*}$.*

Now, we can apply the extended product construction in the same way as for caps to an extendable collection of admissible sets, to find an admissible set again in a higher dimension:

Theorem 5.6. *If $(S_0, S_1, S_2) \subset \{0,1,2\}^m$ is a meta-extendable collection of admissible sets and $T \subset \{0,1,2\}^r$ is an admissible set, then*

$$T(S_0, S_1, S_2) = \bigcup_{t \in T} (S_{t_1} \times \cdots \times S_{t_r})$$

is an admissible set.

For a detailed proof the reader is referred to [9], but here a condensed version is given:

Proof. For the pairwise condition of admissible sets (Adm1) it needs to be checked that any pair $x, y \in T(S_0, S_1, S_2)$ has coordinates i and j such that $x_i = 0 \neq y_i$ and $x_j \neq 0 = y_j$. If x and y come from the same vector t then we are done since the direct product $S_{t_1} \times \cdots \times S_{t_r}$ is admissible. If x and y come from distinct t and t', then since T is admissible we find $t_k = 0 \neq t'_k$ and $t_l \neq 0 = t'_l$, so we find a block from S_0 in x at the same 'height' as a block from $S_1 \cup S_2$ in y and vice versa, as in (5.5):

$$x = \begin{bmatrix} s_1 \in S_{t_1} \\ \vdots \\ s_k \in S_0 \\ \vdots \\ s_l \in S_* \\ \vdots \\ s_r \in S_{t_r} \end{bmatrix} \qquad y = \begin{bmatrix} s'_1 \in S_{t'_1} \\ \vdots \\ s'_k \in S_* \\ \vdots \\ s'_l \in S_0 \\ \vdots \\ s'_r \in S_{t'_r} \end{bmatrix} \tag{5.5}$$

By Meta-Ext1 all vectors from S_0 have more zeros than any vector from $S_1 \cup S_2$, so we find $x_i = 0 \neq y_i$ in the blocks s_k and s'_k, and $x_j \neq 0 = y_j$ in the blocks s_l and s'_l.

For the triples condition of admissible sets (Adm2) it needs to be checked that any triple $x, y, z \in T(S_0, S_1, S_2)$ has a coordinate k such that $(x_k, y_k, z_k) \in \mathrm{N}_{012} \cup \mathrm{M}_{00*}$.

- If x, y, z come from the same t we are done.
- If x and y come from t and z comes from t', then since T is admissible there is a k such that $t_k = 0 \neq t'_k$, so $x_k, z_k \in S_0$ and $z_k \in S_1 \cup S_2$. By Meta-Ext2 we find a row $(x_k, y_k, z_k) \in \mathrm{N}_{012} \cup \mathrm{M}_{00*}$.
- If x, y, and z come from distinct vectors t, t', and t'', then since T is admissible there is a k where on of the following two cases holds:
  1. $(x_k, y_k, z_k) \in \mathrm{M}_{00*}$, so two of x, y, z are in S_0 and one is in $S_1 \cup S_2$.
  2. $(x_k, y_k, z_k) \in \mathrm{N}_{012}$, so x, y, z are from three different sets in (S_0, S_1, S_2).

  In case 1 by Meta-Ext2 and in case 2 by Meta-Ext3 we find a row in $\mathrm{N}_{012} \cup \mathrm{M}_{00*}$.

Thus any pair satisfies Adm1 and any triple satisfies Adm2 in $T(S_0, S_1, S_2)$. □

Recall the admissible set $I(11, 7)$ from Lemma 5.3. Let $S_1 = I(11, 7)$ and let S_2 be equal to $I(11, 7)$ with all 1's and 2's swapped. Then by a computer search, an admissible set S_0 of size $|S_0| = 37$ can be found such that (S_0, S_1, S_2) is meta-extendable[2]. Now let us apply $T = \tilde{I}(142, 141)$, which exists by Lemma 5.2, to this meta-extendable collection of admissible sets. We then find an admissible set $\tilde{T}$:

- $\tilde{T}$ has dimension $mk = 11 \cdot 142 = 1562$
- $|\tilde{T}| = |T||S_0||S_1|^{141} = 142 \cdot 37 \cdot \binom{11}{7}^{141}$
- Each element $x \in S_{t_1} \times \cdots \times S_{t_{142}}$ contains 141 blocks from $S_1 \cup S_2$ and one block from S_0, so the weight of each vector is $141 \cdot 7 + 3 = 990$.

If we now use $S = \tilde{I}(6, 5)$ and $\tilde{T} \subset \{0, 1, 2\}^{1562}$ to extend the extendable collection $A_0, A_1, A_2 \subset \mathbb{F}_3^6$ from Section 5.1 using the construction from Section 5.2, we find a cap $\tilde{B}$ in dimension $6 \cdot 6 \cdot 1562 = 56232$ of size:

$$\begin{aligned} |\tilde{B}| &= |\tilde{T}|(|S| \cdot |A_1|^5 \cdot |A_0|)^{1562-990}(|A_1|^6)^{990} \\ &= 142 \cdot 37 \cdot \binom{11}{7}^{141} \cdot 6^{572} \cdot 12^{572} \cdot 112^{8800} \end{aligned} \tag{5.6}$$

Since $|\tilde{B}|^{\frac{1}{56232}} = 2.21802$, Tyrrell has found a slight improvement compared to the lower bound of 2.21798 from the previous section.

5.4. Using a large language model

A team of researchers of Google used artificial intelligence to find a breakthrough in the cap set problem in [11]. The researchers developed FunSearch (short for searching in the function space): an interaction between a pretrained large language model (LLM) and an evaluator. Initially a specification of the problem is fed to the LLM, which consists of an 'evaluate' function and an initial program to solve.

[2] The set S_0 can be found on Tyrrell's webpage `http://fredtyrrell.com/cap-sets`

FunSearch works best if the initial program has the form of a skeleton, so that FunSearch is only used to evolve the crucial part. For example, to construct caps as large as possible the researchers gave a specification consisting of the following functions:

- A priority function:
  This is the crucial part to evolve by FunSearch. The function maps the elements of $\{0,1,2\}^d$ to a real number, scoring all the candidate elements.
- A solve function:
  The solve function sorts all the elements of $\{0,1,2\}^d$ by the priority function. Then a cap is built by considering the elements one by one, from highest priority to lowest priority. An element is only added to the cap if it is allowed (it does not make a line with elements already in the cap), otherwise it is skipped.
- An evaluate function:
  The evaluate function checks if the resulting set is a cap and if it is, returns the size of the set. This way biggest caps get highest scores.

Note that the solve function calls the priority function that the LLM still has to come up with.

The LLM initially uses the specification to generate solutions for the priority function. These solutions are evaluated and then stored in the program database. Next, each iteration a sample of best scoring programs is taken from the database to create the prompt for the LLM. The LLM combines and extends the best scoring solutions to create a better priority function. At any time the best-scoring program can be taken as 'the solution' of the problem.

Using the LLM 'Codey', the researchers found a cap of size 512 in dimension 8 (the priority function that results in this cap can be found in Figure 4b of [11]). By the doubling of the projective cap[3] of size 248 in $\mathbb{F}_3^7$ found in [12], the previously largest known cap in $\mathbb{F}_3^8$ has size 496. Thus the cap in $\mathbb{F}_3^8$ of size 512 is a significant improvement of the lower bound on a_8.

The researchers used the same strategy to create constant weight admissible sets, where a priority function rates elements of $\{0,1,2\}^m$ to iteratively grow admissible sets. Starting from a trivial constant function, FunSearch finds one that provides us with a full size $I(12,7)$. The discovery of $I(12,7)$ already improves the asymptotic lower bound from 2.21802 to 2.21844.

Now, since FunSearch provides us not the exact solution (here is 'an' admissible set), but *how* to obtain the solution, by studying the priority function the researches found that it treats the coordinates of elements of $\{0,1,2\}^m$ in a highly symmetric way. Now the researchers could push the boundaries of what admissible sets can be constructed by searching directly for symmetric admissible sets, allowing higher dimensions and weights. This led the researchers to the discovery of a full size $I(15,10)$ and an admissible set $A(24,17)$ of size 237984. These discoveries yield new asymptotic lower bounds of 2.21948 and 2.22023 respectively.

To summarize, thanks to FunSearch new admissible sets were found, with better lower bounds as a result:

- The full size admissible set $I(12,7)$.
  $T = I(12,7)$ together with $S = \tilde{I}(7,6)$ results in a lower bound of 2.2184.
- The full size admissible set $I(15,10)$.
  $T = I(15,10)$ together with $S = \tilde{I}(5,4)$ results in a lower bound of 2.2195.
- The admissible set $A(24,17)$ of size 237984.
  $T = A(24,17)$ together with $S = \tilde{I}(4,3)$ results in a lower bound of 2.2202.

[3] We have not discussed projective caps in this report. Note that a projective cap in $\mathbb{F}_3^7$ is not the same as an affine 7-cap (how we defined caps in Definition 2.3). Therefore the projective cap of size 248 does not improve the best known lower bound of 236 in dimension 7. However, a doubling of a projective d-cap of size $|C|$ yields an $(d+1)$-cap of size $2|C|$ by the doubling construction.

5.5. Conclusion

In Table 5.1 the improvements of the asymptotic lower bound on the size of caps from this chapter are summarized. Each row has a better lower bound than the previous row. The first row is due to Edel. The middle four rows are found by Tyrrell using Edel's methods. The last of these rows uses a constant weight admissible set $A(1562, 990)$ of size $142 \cdot 37 \cdot \binom{11}{7}^{141}$, which is due to Tyrrell's extension of Edel's methods with meta-extendable admissible sets. The last three rows are due to the researchers of Google, who discovered new admissible sets thanks to FunSearch. The constant weight admissible set $A(24, 17)$ in the last row has size 237984.

Dimension	Recursively Admissible S	Admissible T	Asymptotic Lower Bound
480	$\tilde{I}(8,7)$	$I(10,5)$	2.21738
420	$\tilde{I}(7,6)$	$I(10,6)$	2.21756
462	$\tilde{I}(7,6)$	$I(11,6)$	2.21795
396	$\tilde{I}(6,5)$	$I(10,6)$	2.21798
56232	$\tilde{I}(6,5)$	$A(1562,990)$	2.21802
504	$\tilde{I}(7,6)$	$I(12,7)$	2.21844
450	$\tilde{I}(5,4)$	$I(15,10)$	2.21948
576	$\tilde{I}(4,3)$	$A(24,17)$	2.22023

Table 5.1: Lower bound improvements from the trivial bound 2 until the best known bound 2.22023.

In all constructions discussed in this chapter the starting point is the extendable collection in $\mathbb{F}_3^6$ based on the Hill cap with $|A_0| = 12$ and $|A_1| = |A_2| = 112$ from Section 5.1. The size of the resulting caps can be computed with the formula in (5.4), with k from $S = \tilde{I}(k, k-1)$ in the second column of Table 5.1 and m and w from $T = I(m, w)$ (or $A(m, w)$) in the third column of Table 5.1.

6

Admissible sets and extendable collections

This chapter expands on admissible sets and extendable collections. We classify extendable collections and admissible sets in low dimensions and we give explicit constructions for some recursively admissible sets of constant weight.

6.1. Constructions for admissible sets

Conjecture 6.1. Tyrrell's conjecture.
For any $m > w > 0$ there exists an admissible set $I(m, w)$ consisting of $\binom{m}{w}$ vectors.

If Tyrrell's conjecture turns out to be true, then this would imply a lower bound of 2.233 by using $S = I(m, \frac{28m}{31})$ with large m to obtain a cap in $\mathbb{F}_3^{6m}$ of size 124^m [9]. The new bound 2.233 would improve the current best known bound 2.2202 from Section 5.4. It is therefore valuable to invent constructions for constant weight recursively admissible sets that exist in any dimension, for example the recursively admissible set $\tilde{I}(k, k-1)$ from Section 5.2.

According to Tyrrell the constant weight admissible sets $\tilde{I}(k, 0)$, $\tilde{I}(k, 1)$, $\tilde{I}(k, 2)$, $\tilde{I}(k, 3)$, $\tilde{I}(k, k-1)$, and $\tilde{I}(k, k)$ are known to exist. Of course, $\tilde{I}(k, 0)$, $\tilde{I}(k, 1)$ and $\tilde{I}(k, k)$ are quite trivial and we have already seen $\tilde{I}(k, k-1)$. Tyrrell states that "we can prove the existence of admissible sets of weight 2 and 3 for all k, via a similar construction". However, no constructions and proofs are given in [9]. This section gives an overview of these constant weight admissible sets, besides $\tilde{I}(k, k-1)$, including constructions and proofs.

The cases $\tilde{I}(k, 0)$ and $\tilde{I}(k, k)$ are trivial. Since $\binom{k}{0} = \binom{k}{k} = 1$, these sets must contain exactly one vector and thus automatically satisfy Adm1, Adm2, Rec1, and Rec2, since the sets do not contain pairs or triples. Explicit examples of these constant weight recursively admissible sets are $\tilde{I}(k, k) = \{\vec{0}\}$ and $\tilde{I}(k, k) = \{\vec{1}\}$.

Another quite trivial case is the existence of $\tilde{I}(k, 1)$, stated in Lemma 6.2.

Lemma 6.2. *For any $k \geq 2$ there exists a recursively admissible set $\tilde{I}(k, 1)$.*

Proof. If $k = 2$, then $\tilde{I}(k, 1) = \tilde{I}(k, k-1)$, which exists by Lemma 5.2.

For $k \geq 3$ let $S = \{e_1, \dots, e_k\}$, where e_i denotes the vector $(0, \dots, 1, \dots, 0)^T$ with the 1 in the i'th position. Then S is recursively admissible with $|S| = k$:

Since each pair of vectors has their non-zero entries in two different positions, each pair satisfies Adm1. Moreover, since each triple of vectors has their non-zero entries in three different positions, we can find a position k such that the k'th row of the triple is in M_{00*} (with non-zero entry 1), so the triple satisfies

Adm2. Lastly, each pair of vectors has $k-2$ coordinates where both entries equal 0, thus for $k \geq 3$ each pair satisfies Rec1. □

Now we turn to the more complicated cases $\tilde{I}(k,2)$ and $\tilde{I}(k,3)$, stated in Lemma 6.3 and Lemma 6.4.

Lemma 6.3. *For any $k \geq 3$ there exists a recursively admissible set $\tilde{I}(k,2)$.*

Proof. If $k = 3$, then $\tilde{I}(k,2) = \tilde{I}(k,k-1)$, which we know exists by Lemma 5.2. For $k \geq 4$, let S consist of the $\binom{k}{2}$ vectors with zeros in exactly $k-2$ positions, where no two vectors have the same support. For each vector, let the first non-zero entry be 1 and the second non-zero entry be 2. Then S is recursively admissible.

We prove that $S = \tilde{I}(k,2)$ is recursively admissible by using induction with base case $k = 4$.

If $k = 4$, the resulting set is as shown in expression (6.1).

$$\tilde{I}(4,2) = \left\{ \begin{bmatrix}1\\2\\0\\0\end{bmatrix}, \begin{bmatrix}1\\0\\2\\0\end{bmatrix}, \begin{bmatrix}1\\0\\0\\2\end{bmatrix}, \begin{bmatrix}0\\1\\2\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\\2\end{bmatrix}, \begin{bmatrix}0\\0\\1\\2\end{bmatrix} \right\} \tag{6.1}$$

It can be manually checked that this set satisfies Adm1 and Adm2, and that each pair satisfies either Rec1 or Rec2. In Appendix A.1 a code can be found to run this check by a computer. We continue with the ascertainment that $\tilde{I}(4,2)$ is recursively admissible.

$\tilde{I}(k,k-2)$ can be written in a structural way: start with $s_1 = 1$ and $s_i = 2$ for $i = 2,\dots,k$ to create the first $k-1$ columns, then move on to $s_1 = 0$, $s_2 = 1$ and $s_i = 2$ for $i = 3,\dots,k$ to create the next $k-2$ columns, and continue like this. Note that $\tilde{I}(4,2)$ is written like this in expression (6.1). If we write $\tilde{I}(5,2)$ in the same way, and we present the elements $s \in \tilde{I}(5,2)$ as columns of a matrix, then the resulting set is as shown in matrix (6.2).

$$\begin{array}{cccc|cccccc}
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
2 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 2 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 2 & 0 & 0 & 2 & 0 & 2 & 0 & 1 \\
0 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 2 & 2
\end{array} \tag{6.2}$$

The key observation is that the lower right block in the matrix is equal to $\tilde{I}(4,2)$. In general, for $k \geq 5$, the elements can be written as the columns in matrix (6.3):

$$\begin{array}{cccc|c}
1 & 1 & \dots & 1 & 0 \quad \dots \quad 0 \\
\hline
2 & 0 & \dots & 0 & \\
0 & 2 & \dots & 0 & \tilde{I}(k-1,2) \\
\vdots & & \ddots & \vdots & \\
0 & 0 & \dots & 2 &
\end{array} \tag{6.3}$$

The upper left box is a row consisting of only 1's. The upper right box is a row consisting of only 0's. The lower left box is an $(k-1)\times(k-1)$ diagonal matrix with 2's on the diagonal. The lower right box is equal to $\tilde{I}(k-1,2)$. Now we can prove that $\tilde{I}(k,2)$ is recursively admissible for all $k \geq 4$ by induction, using as base case that $\tilde{I}(4,2)$ is recursively admissible. Let us split the elements of $\tilde{I}(k,2)$ into two sets by the vertical bar in matrix (6.3): the elements to the left are in S_1 and the elements to the right are in S_2.

Let $k \geq 5$.

- All pairs $s, s' \in \tilde{I}(k,2)$ have weight 2, so there are at least $k-4$ coordinates i where $s_i = s'_i = 0$. Since $k \geq 5$ we find at least one such i. Thus all pairs satisfy Rec2.

- Adm1
  - Let $s, s' \in S_1$. Then there exist indices i and j such that $s_i = 2$ and $s'_i = 0$ and $s_j = 0$ and $s'_j = 2$.
  - Let $s, s' \in S_2$. Since $\tilde{I}(k-1, 2)$ is admissible, the pair satisfies Adm1.
  - Let $s \in S_1, s' \in S_2$. Then $s_1 = 1 \neq 0 = s'_1$. Moreover, after deletion of the first coordinate of s and s', the new s has weight 1 and the new s' has weight 2, so there is an index $j \geq 2$ such that $s_j = 0 \neq s'_j$.

  Thus all pairs satisfy Adm1.
- Adm2
  - Let $s, s', s'' \in S_1$. Then there is an index i such that $(s_i, s'_i, s''_i) \in \mathrm{M}_{00*}$ with non-zero entry 2.
  - Let $s, s', s'' \in S_2$. Since $\tilde{I}(k-1, 2)$ is admissible, the triple satisfies Adm2.
  - Let $s, s' \in S_1, s'' \in S_2$. The pair s, s' has an i such that $s_i = 2$ and $s'_i = 0$ and a j such that $s_j = 0$ and $s'_j = 2$. For $k \geq 5$ there are at least 2 coordinates where s and s' both have zeros. There are 2 cases:
    1. s'' has at least one of its non-zero entries in a coordinate where s and s' have zeros, so we find a row in M_{00*}.
    2. s'' has none of its non-zero entries in a coordinate where s and s' have zeros, so $\{s''_i, s''_j\} = \{1, 2\}$ and we find a row in N_{012}.
  - Let $s \in S_1, s', s'' \in S_2$. Then $(s_1, s'_1, s''_1) = (1, 0, 0) \in \mathrm{M}_{00*}$.

  Thus all triples satisfy Adm2.

We conclude that $\tilde{I}(k, 2)$ is recursively admissible. □

Lemma 6.4. *For any $k \geq 4$ there exists an admissible set $I(k, 3)$ and for any $k \geq 6$ there exists a recursively admissible set $\tilde{I}(k, 3)$.*

Proof. Let S consist of the $\binom{k}{3}$ vectors with zeros in exactly $k-3$ positions, where no two vectors have the same support. For each vector, let the first non-zero entry be 2, the second non-zero entry be 1, and the third non-zero entry be 2. Then S is admissible for $k \geq 4$:

Since all vectors have different support, for any pair $s, s' \in S$ we find i, j such that $x_i = 0 \neq y_i$ and $x_j \neq 0 = y_j$, so Adm1 is satisfied.

Let $s, s', s'' \in S$. For Adm2 we need the triple to have a coordinate i such that $(s_i, s'_i, s''_i) \in \mathrm{N}_{012} \cup \mathrm{M}_{00*}$. We say that an element of the support of a vector is *covered* if one of the other vectors has an element of their support in the same position. For example, in Figure 6.1 the second and third non-zero elements of s are covered by s'' and s' respectively.

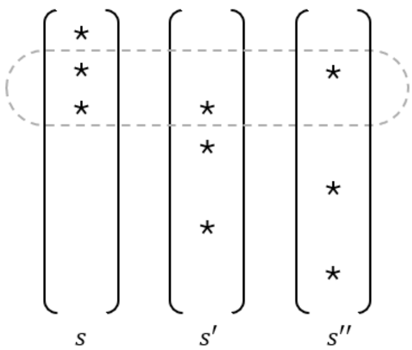


Figure 6.1: The second and third element of the support of s are covered.

If the supports of the three vectors do not all cover each other, then the triple satisfies Adm2, since we find an i where $(s_i, s_i', s_i'') \in \mathrm{M}_{00*}$. Figure 6.2 demonstrates this.

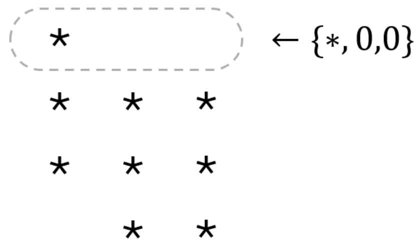


Figure 6.2: At least one of s, s' and s'' has a support that is not entirely covered.

We are left with the cases where all three supports are covered by each other. We can assume that the only triples that could possibly form a problem have supports that are aligned in the following way, where the third support element of z is in *one* of the places marked with '·':

s	s'	s''
×		×
×	×	·
×	×	·
	×	×

Why is this 'the only' shape where all three supports overlap?

- *The supports of s and s' need to overlap in at least two positions, since otherwise the support of s'' would have to cover four or more support elements of s and s'.*
- *No two vectors in S have the same support, so two vectors have overlapping support in at most two positions.*
- *Two support elements of s'' must cover the uncovered support elements of s and s'.*
- *The third support element of s'' may only be placed at the marks '·' since otherwise it would be uncovered.*

We see that if all three supports are covered by each other there are at most four rows with non-zero entries: the rest of the rows have only zeros. Thus we only need to check if S is recursively admissible for $k = 4$:

$$S = \tilde{I}(4,3) = \left\{ \begin{bmatrix} 0 \\ 2 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 0 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \\ 2 \\ 0 \end{bmatrix} \right\}$$

It can easily be verified that any triple in this set contains a row in N_{012}. We conclude that S satisfies Adm2.

For $k \geq 6$, S is also recursive. All vectors have weight 3, so any pair of vectors $s, s' \in S$ has at least $k-6$ coordinates i where $x_i = y_i = 0$. So for $k > 6$, the pairs in S satisfy Rec2. If $k = 6$ and there is a pair $s, s' \in S$ with no i such that $x_i = y_i = 0$, the supports of s and s' are disjoint, so we must find i, j such that $\{s_i, s_i'\} = \{0, 1\}$ and $\{s_j, s_j'\} = \{0, 2\}$, and S satisfies Rec1.

We conclude that S is admissible for $k \geq 4$ and recursively admissible for $k \geq 6$. □

6.2. Potential construction for admissible $I(k, k-2)$

In this section we discuss an attempt to make a construction for an admissible set $I(k, k-2)$ that works for any dimension k. That is, we try to build a k-dimensional admissible set that consists of $\binom{k}{k-2}$ vectors, where each vector has exactly two zeros.

The basic idea of the structure is using blocks of $\tilde{I}(k, k-1)$, which exists by Lemma 5.2. For example, to build an admissible set $I(5,3)$ consisting of 10 vectors, we would fix the following structure:

$$\begin{array}{|cccc|ccc|cc|c|}
\hline
0&0&0&0&*&*&*&*&*&*\\
\hline
&&&&0&0&0&*&*&*\\
\cline{5-7}
&&&&&&&0&0&2\\
\cline{8-9}
&\tilde{I}(4,3)&&&&\tilde{I}(3,2)&&0&1&0\\
&&&&&&&2&0&0\\
\hline
\end{array} \tag{6.4}$$

The non-zero elements, marked $*$, in (6.4) still have to be determined. In Appendix A.4 a code can be found that checks all possibilities for these non-zero elements. Note that in the last vector, one non-zero entry already has been fixed to be 2. This is because when this non-zero entry equals 1 no admissible sets are found. For the remaining nine non-zero elements, there are 256 ways of filling these in that result in admissible sets $I(5,3)$. None of these sets are recursively admissible.

Now, let us try to use these results to build an admissible $I(6,4)$ of the same structure, by first considering *one* of the 256 admissible sets $I(5,3)$ that were found in the previous step: take all non-zero elements in (6.4) equal to 2. That leaves us with the following structure for $I(6,4)$:

$$\begin{array}{|ccccc|cccc|ccc|cc|c|}
\hline
0&0&0&0&0&*&*&*&*&*&*&*&*&*&*\\
\hline
&&&&&0&0&0&0&2&2&2&2&2&2\\
\cline{6-9}
&&&&&&&&&0&0&0&2&2&2\\
\cline{10-12}
&&&&&&&&&&&&0&0&2\\
\cline{13-14}
&&\tilde{I}(5,4)&&&&\tilde{I}(4,3)&&&&\tilde{I}(3,2)&&0&1&0\\
&&&&&&&&&&&&2&0&0\\
\hline
\end{array} \tag{6.5}$$

The first row in (6.5) still has ten non-zero entries left to be determined. That means that there are $2^{10} = 1024$ possibilities to consider. The code in Appendix A.4 checks all options and finds 512 admissible sets $I(6,4)$, containing the fixed admissible set $I(5,3)$.

Now, to consider *all* possible admissible sets we fix the following structure for $I(6,4)$:

$$\begin{array}{|ccccc|cccc|ccc|cc|c|}
\hline
0&0&0&0&0&*&*&*&*&*&*&*&*&*&*\\
\hline
&&&&&0&0&0&0&*&*&*&*&*&*\\
\cline{6-9}
&&&&&&&&&0&0&0&*&*&*\\
\cline{10-12}
&&&&&&&&&&&&0&0&2\\
\cline{13-14}
&&\tilde{I}(5,4)&&&&\tilde{I}(4,3)&&&&\tilde{I}(3,2)&&0&1&0\\
&&&&&&&&&&&&2&0&0\\
\hline
\end{array} \tag{6.6}$$

Since it is only necessary to consider fillings of the non-zero entries in (6.4) that result in admissible sets $I(5,3)$, and we found 256 of such fillings, one would have to check $256 \cdot 1024 = 262144$ options. The code in Appendix A.4 does this in a reasonable time and counts 131072 admissible sets $I(6,5)$ with the fixed structure of (6.6).

To continue to finding admissible sets $I(7,5)$, one would have to check $131072 \cdot 2^{15}$ (approximately 4 billion!) options. We see that as we move on to higher dimensions, the number of options to consider quickly explodes. This is why we end the experiment here.

From this experiment, no conclusions can be drawn yet about whether it is possible to construct an admissible set $I(k, k-2)$ using blocks of $\tilde{I}(k, k-1)$. Thus far we did not find a pattern that seems to work in every dimension k. However, neither did we find a reason to rule out the option that it might work. The problem of finding an admissible $I(k, k-2)$ remains unresolved.

6.3. Admissible sets in two dimensions

In this section we list all admissible sets in two dimensions. To construct an admissible set in two dimensions, we pick from the nine elements in $\{0,1,2\}^2$. In Appendix A.3 a code can be found that, given a size $|S|$, checks for all the possibilities of picking $|S|$ cards from the deck if the picked cards form an admissible set or a recursively admissible set. Obviously, any set of size 1 satisfies Adm1 and

Adm2, since there are no pairs and triples, thus we only consider sizes $|S| \geq 2$. In two dimensions we find two admissible sets and two recursively admissible sets:

$$\left\{\begin{bmatrix}0\\1\end{bmatrix},\begin{bmatrix}2\\0\end{bmatrix}\right\} \text{ and } \left\{\begin{bmatrix}0\\2\end{bmatrix},\begin{bmatrix}1\\0\end{bmatrix}\right\} \text{ are recursively admissible}$$

$$\left\{\begin{bmatrix}0\\1\end{bmatrix},\begin{bmatrix}1\\0\end{bmatrix}\right\} \text{ and } \left\{\begin{bmatrix}0\\2\end{bmatrix},\begin{bmatrix}2\\0\end{bmatrix}\right\} \text{ are admissible}$$

There are some operations that can be applied to an admissible set that preserve the conditions for admissible sets:

- Swapping all 1's and 2's.
- Swapping rows, i.e. switching the i'th entry with the j'th entry in every element of the set.

If one admissible set can be turned into another admissible set by a sequence of these operations, then we call these sets equivalent. Thus the two recursively admissible sets are equivalent to each other and the two admissible sets as well. That means that there is essentially *one* admissible set and *one* recursively admissible set in two dimensions.

6.4. Admissible sets in three dimensions

In this section we list all admissible sets in three dimensions. To construct an admissible set in three dimensions, we pick from the 27 elements in $\{0,1,2\}^3$. Given a size $|S|$, the code in Appendix A.3 can check if subsets of $\{0,1,2\}^3$ of this size are (recursively) admissible sets. Again we only consider sizes $|S| \geq 2$. We find the following:

- $|S| = 2$
  There are 54 recursively admissible sets and 30 admissible sets.
- $|S| = 3$
  There are 16 recursively admissible sets and 48 admissible sets. All the sets are constant weight.
- $|S| = 4$
  No (recursively) admissible sets of size 4 are found. Since a subset of any admissible set again has to be admissible, there are also no (recursively) admissible sets of sizes larger than 4.

The larger the admissible set we use in the extended product construction, the larger the resulting cap will be. Thus, in order to find large caps, it is most interesting to look at the admissible sets of size 3. There are 16 recursively admissible sets. Recall however that some of these might be equivalent to each other.

Example 1. *The following sets are equivalent. The first '=' is by swapping rows and the second '=' is by swapping* 1*'s and* 2*'s.*

$$\left\{\begin{bmatrix}0\\1\\1\end{bmatrix},\begin{bmatrix}2\\2\\0\end{bmatrix},\begin{bmatrix}2\\0\\1\end{bmatrix}\right\} = \left\{\begin{bmatrix}0\\1\\1\end{bmatrix},\begin{bmatrix}2\\0\\2\end{bmatrix},\begin{bmatrix}2\\1\\0\end{bmatrix}\right\} = \left\{\begin{bmatrix}0\\2\\2\end{bmatrix},\begin{bmatrix}1\\0\\1\end{bmatrix},\begin{bmatrix}1\\2\\0\end{bmatrix}\right\}$$

That splits the 16 recursively admissible sets into four equivalence classes, with the following representatives:

Admissible sets with constant weight 1:

- $\left\{\begin{bmatrix}0\\0\\1\end{bmatrix},\begin{bmatrix}0\\1\\0\end{bmatrix},\begin{bmatrix}1\\0\\0\end{bmatrix}\right\}$

  A set with every non-zero entry the same element.
- $\left\{\begin{bmatrix}0\\0\\1\end{bmatrix},\begin{bmatrix}0\\1\\0\end{bmatrix},\begin{bmatrix}2\\0\\0\end{bmatrix}\right\}$

A set where two elements have the same non-zero entry and the third element has a different non-zero entry.

Admissible sets with constant weight 2:

- $\left\{\begin{bmatrix}0\\2\\2\end{bmatrix},\begin{bmatrix}1\\0\\2\end{bmatrix},\begin{bmatrix}1\\1\\0\end{bmatrix}\right\}$

  A set where one element has two 1's, one element has two 2's and one element has a 1 and a 2. Note that this is $\tilde{I}(k,k-1)$ with $k=3$ from Lemma 5.2.

- $\left\{\begin{bmatrix}0\\1\\2\end{bmatrix},\begin{bmatrix}2\\0\\1\end{bmatrix},\begin{bmatrix}1\\2\\0\end{bmatrix}\right\}$

  A set where every element has a 1 and a 2.

In conclusion, there are four different 3-dimensional recursively admissible sets of size 3.

6.5. Extendable collections

In this section we try to find extendable collections in low dimensions and we treat symmetries in extendable collections.

Recall that an extendable collection is a collection of caps $A_0, A_1, A_2 \subset \mathbb{F}_3^n$, that satisfy the conditions Extendable 1 and Extendable 2 (Definition 4.1):

- Extendable 1: *If* $x, y \in A_0$ *and* $z \in A_1 \cup A_2$ *then* $x+y+z \neq 0$
- Extendable 2: *If* $x \in A_0, y \in A_1$ *and* $z \in A_2$ *then* $x+y+z \neq 0$

The following results are directly implied by the definition of extendable collections (see also the remarks after Definition 4.1):

- A_0 is disjoint from $A_1 \cup A_2$
- For each distinct pair of points $x, y \in A_0$ there is a point z that is not allowed in $A_1 \cup A_2$
- $A_0 \cup (A_1 \cap A_2)$ is a cap

Though it is not a requirement in Definition 4.1, in the following sections we try to find extendable collection where none of the caps are empty. Thus we assume $|A_0|, |A_1|, |A_2| \geq 1$. Furthermore, we try to find extendable collections where $|A_1| = |A_2|$, so that we can use Lemma 4.7. It is however also possible to have the code in Appendix B.2 and Appendix B.3 find extendable collections with $|A_1| \neq |A_2|$.

One-dimensional extendable collections

In this section we show that there is no useful extendable collection in $\mathbb{F}_3$. Playing a 1-dimensional SET game can be imagined as playing SET with only one feature, say shape, while the other features are omitted (as in Figure 3.1 in Section 3.1).

Let $A_0, A_1, A_2 \subset \mathbb{F}_3$ be an extendable collection. Note that the three elements of $\mathbb{F}_3$ form a line, so A_0 does not contain all three cards: $|A_0| \neq 3$. This leaves two cases for the size of A_0:

1. If $|A_0| = 2$, then A_0 contains a pair which determines a 'forbidden' card in $A_1 \cup A_2$ in order to satisfy Ext2. Since A_0 should also be disjoint from $A_1 \cup A_2$, there are no cards left to put in $A_1 \cup A_2$.
2. If $|A_0| = 1$, then there are two cards left to put in A_1 and A_2. If we put the two cards in the two different sets, then Ext2 is not satisfied, but if we put both cards in the same set, then one set is empty. Thus only one card can be used and it is used double, such that $A_1 = A_2$.

We conclude that the only extendable collection of non-empty caps in $\mathbb{F}_3$ has $|A_0| = |A_1| = |A_2| = 1$ and $A_1 = A_2$. However, extendable collections turn out to *not* be of added value when $A_1 = A_2$.

Lemma 6.5. *Using an extendable collection $A_0, A_1, A_2 \subset \mathbb{F}_3^d$ with $A_1 = A_2$ in the extended product construction does not have advantage compared to the direct product construction.*

Proof. Note that if $A_1 = A_2$, then $(A_1 \cap A_2) = (A_1 \cup A_2)$, so since $A_0 \cup (A_1 \cap A_2)$ is a cap, $A_0 \cup A_1 \cup A_2$ is a cap. But $s(A_0, A_1, A_2) = A_{s_1} \times \ldots \times A_{s_m} \subset (A_0 \cup A_1 \cup A_2) \times \ldots \times (A_0 \cup A_1 \cup A_2)$ for any $s \in \{0, 1, 2\}^m$, so that means that the cap obtained by applying the extended product construction with an admissible set will never be better than taking the direct product of $(A_0 \cup A_1 \cup A_2)$. ☐

Therefore it can be assumed that $A_1 \neq A_2$ for an extendable collection $A_0, A_1, A_2 \subset \mathbb{F}_3^d$ and we conclude that there is no (useful) extendable collection in $\mathbb{F}_3$.

Symmetries in extendable collections

Before we turn to extendable collections in $\mathbb{F}_3^2$, we will look at symmetries in extendable collections.

Recall the elementary row operations from Section 2.3: switching rows, scalar multiplication of rows, and adding (multiples of) rows to rows. These row operations as well as translations preserve Ext1 and Ext2. We see that we can define equivalent extendable collections similarly to how we have defined caps of the same type: if we can take one extendable collection in $\mathbb{F}_3^d$ to another by using the transformation $A\vec{v} + \vec{b}$, then we call these extendable collections equivalent. Here, A is an invertible $d \times d$-matrix and $\vec{b}$ represents a translation. A can be written as a product of elementary matrices.

Example 2. Equivalence in 2-dimensional extendable collections.
Recall that the elements of $\mathbb{F}_3^2$ can be schematically represented in a 2-dimensional grid using Notation 5.

In $\mathbb{F}_3^2$ there are four elementary matrices. These are listed below, followed by an explanation how the operations affect the grid.

1. $\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x+y \\ y \end{bmatrix}$

   In the grid the second row shifts one place to the right and the third row shifts two places to the right (or equivalently one place to the left). The first row (where $y = 0$) remains the same.

   In terms of the vector $(x, y)^T$ this is a row sum, where the second row is added to the first.

2. $\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ x+y \end{bmatrix}$

   This shifts the second column one place downwards and the third column two places downwards (or equivalently one place upward). The first column (where $x = 0$) remains unshifted.

3. $\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -x \\ y \end{bmatrix}$

   This swaps the second and third column of the grid.

   In terms of the vector $(x, y)^T$ it is a scalar multiplication, with scalar $2 = -1$.

4. $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ -y \end{bmatrix}$

   This swaps the second and third row of the grid.

Extendable collections in $\mathbb{F}_3^2$

In this section we turn to extendable collections in $\mathbb{F}_3^2$. A 2-dimensional SET game can be imagined as playing SET with two features, say shape and colour.

In Appendix B.1 a code can be found with functions for checking if sets are caps, and if collections of (cap) sets are extendable. In Appendix B.2 a program is written that uses these functions to find all extendable collections in $\mathbb{F}_3^2$, given $|A_0| = 1, 2$ or 3. The program uses a prefixed A_0 and a ‘deck’ of cards that are left as potential elements of $A_1 \cup A_2$. When the program runs it asks you to input sizes for A_0, A_1 and A_2 and then finds all extendable collections.

Suppose we start with $|A_0| = 1$.
We start with $A_0 = \vec{0}$ and the 'deck' of possible cards for $A_1 \cup A_2$ is in this case $\mathbb{F}_3^2 \backslash \{\vec{0}\}$. The maximum size of a cap in two dimensions is 4 so for $|A_1| = |A_2|$ it is possible to choose sizes 1, 2, 3, and 4.

- There are 24 extendable collections with $|A_0| = 1$ and $|A_1| = |A_2| = 1$. Note that this can easily be deduced without any code, since $\vec{0}$ is contained in exactly 4 lines, so there are $\binom{8}{2} - 4 = 24$ extendable collections.
- There are 198 extendable collections with $|A_0| = 1$ and $|A_1| = |A_2| = 2$.
- There are 192 extendable collections with $|A_0| = 1$ and $|A_1| = |A_2| = 3$.
- There are 3 extendable collections with $|A_0| = 1$ and $|A_1| = |A_2| = 4$.

The larger the sets are of the extendable collection we use in the extended product construction, the larger the resulting cap will be. Since in the list above the size of A_0 is fixed, the most useful result is the fact that there exist 3 extendable collections with $|A_0| = 1$ and $|A_1| = |A_2| = 4$. These turn out to all be equivalent to each other (by repeatedly using the operation that shifts the second row one place and the third row two places to the right) and one representative is shown in Figure 6.3.
Now, suppose we start with $|A_0| = 2$.

A_0	A_1	A_1
A_1	A_2	A_2
A_1	A_2	A_2

Figure 6.3: Extendable collection in $\mathbb{F}_3^2$ with $|A_0| = 1$ and $|A_1| = |A_2| = 4$

Since for any two points we can apply an affine transformation to obtain a favorable starting position, we can choose and fix the two cards of A_0. We leave these two cards plus the 'forbidden card' they determine out of the deck of possible cards for $A_1 \cup A_2$. We find the following:

- There are 9 extendable collections with $|A_0| = 2$ and $|A_1| = |A_2| = 1$.
- There are 9 extendable collections with $|A_0| = 2$ and $|A_1| = |A_2| = 2$.
- There are *no* extendable collections with $|A_0| = 2$ and $|A_1| = |A_2| = 3$ or 4.

Again, it is most useful to look at extendable collections with the sizes of A_1 and A_2 as big as possible, so we turn to the extendable collections with $|A_1| = |A_2| = 2$, which are represented in the two-dimensional grids in Figure 6.4.

Now, let us consider which extendable collections are equivalent to each other:

- $(1) \sim (4)$, $(2) \sim (5)$, and $(3) \sim (6)$ by swapping the second and third row.
- $(1) \sim (2)$ by swapping the first and second column. This can be done by shifting all columns one place to the right, then swapping the second and third column, and then shifting all columns two places more to the right (or equivalently one place back to the left).
- $(1) \sim (3)$ by shifting the second row one place (and the third row two places) to the right and then swapping the first and second column using the same trick as before.

We conclude that (1), (2), (3), (4), (5), and (6) are all equivalent to each other.

- $(7) \sim (8)$ by shifting the second row one place and the third row two places to the right.
- $(8) \sim (9)$ by shifting the second row one place and the third row two places to the right again.

We conclude that (7), (8), and (9) are equivalent to each other.

If $|A_0| = 3$, we can formally prove that an extendable collection $A_0, A_1, A_2 \subset \mathbb{F}_3^2$ with $|A_1| = |A_2| = 3$ cannot exist.

(1)

A_0	A_0	$\times$
A_1/A_2	A_1	A_2

(2)

A_0	A_0	$\times$
A_1	A_1/A_2	A_2

(3)

A_0	A_0	$\times$
A_1	A_2	A_1/A_2

(4)

A_0	A_0	$\times$
A_1/A_2	A_1	A_2

(5)

A_0	A_0	$\times$
A_1	A_1/A_2	A_2

(6)

A_0	A_0	$\times$
A_1	A_2	A_1/A_2

(7)

A_0	A_0	$\times$
A_1	A_2	
A_1	A_2	

(8)

A_0	A_0	$\times$
	A_1	A_2
A_2		A_1

(9)

A_0	A_0	$\times$
A_1		A_2
	A_2	A_1

Figure 6.4: Extendable collections in $\mathbb{F}_3^2$ with $|A_1| = |A_2| = 2$

Lemma 6.6. *There exists no extendable collection of caps A_0, A_1, A_2 in $\mathbb{F}_3^2$ with $|A_0| = |A_1| = |A_2| = 3$*

Proof. Assume to a contradiction that there exists an extendable collection $A_0, A_1, A_2 \subset \mathbb{F}_3^2$ with $|A_0| = |A_1| = |A_2| = 3$. For each pair $x, y \in A_0$ there is a z such that $x + y + z = 0$, so $z \notin A_1 \cup A_2$ in order to satisfy Extendable 1. Then $|A_1 \cup A_2| \leq 3$, since $\{0, 1, 2\}^2$ contains 9 elements in total, the 3 elements of A_0 are not in $A_1 \cup A_2$, and the 3 pairs of A_0 determine 3 forbidden points in $A_1 \cup A_2$. Then:

$$3 \geq |A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2| = 6 - |A_1 \cap A_2| \implies |A_1 \cap A_2| \geq 3$$

But since $A_0 \cup (A_1 \cap A_2)$ is a cap we find a cap in $\mathbb{F}_3^2$ of size at least 6, which is impossible. □

Now, this does not rule out the possibilities that there exist extendable collections with $|A_1| = |A_2| = 1$ or 2. However, when $|A_0| = 3$, the program in Appendix B.2 checks all options and finds no extendable collections at all. We conclude the following from this section:

Proposition 6.7. *Except for symmetry, the only extendable collections of caps A_0, A_1, A_2 in $\mathbb{F}_3^2$ of sizes:*

- $|A_0| = 1$ *and* $|A_1| = |A_2| = 4$
- $|A_0| = |A_1| = |A_2| = 2$ *and* $|A_1 \cap A_2| = 1$
- $|A_0| = |A_1| = |A_2| = 2$ *and* $A_1 \cap A_2 = \emptyset$

Extendable collections in $\mathbb{F}_3^3$

In this section we try to find extendable collections in $\mathbb{F}_3^3 = \{0, 1, 2\}^3$. This can be imagined as playing a SET game with three features, say shape, colour, and shading. It is again useful to use the notation introduced in the previous section, only in order to show three dimensions in a two-dimensional grid we place three grids next to each other: the first numbered 0, the second numbered 1 and the third numbered 2. The element $(2, 0, 1)^T$ for example corresponds to the third grid (the grid on the right), the first column and the second row. In Figure 6.5 a maximum cap in $\mathbb{F}_3^3$ is represented this way.

$\times$		$\times$
$\times$		$\times$

	$\times$	

	$\times$	
$\times$		$\times$
	$\times$	

Figure 6.5: A maximum cap in $\mathbb{F}_3^3$.

In Appendix B.3 a code is written that finds an extendable collection in $\mathbb{F}_3^3$, given A_0, a 'deck' of possible cards for $A_1 \cup A_2$, and some size for $|A_1| = |A_2|$. For choosing a cap A_0 to start with we will make a subset of the maximum cap in Figure 6.5. Recall that each pair of points in A_0 determines a ‘forbidden’ point in $A_1 \cup A_2$. Thus, if we want $|A_0| \geq 4$, it is a good starting position to choose either the four points in the first or the four points in the third grid, since then two pairs of points rule out the same point, leaving one card extra in the potential deck for $A_1 \cup A_2$.

Suppose that $|A_0| = 4$, where $A_0 = \{(0,0,0)^T, (0,2,0)^T, (0,0,2)^T, (0,2,2)^T\}$ (the four crosses in the first grid of Figure 6.5). Then the code in Appendix B.3 finds extendable collections for sizes $|A_1| = |A_2| = 1$ up to 5. For $|A_1| = |A_2| \geq 6$ no extendable collections are found. It has to be mentioned that running the code with $|A_1| = |A_2| = 6$ is very time consuming.

Suppose that $|A_0| = 5$. To find suitable extendable collections we start with

$$A_0 = \left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\},$$

the crosses in the first and second grid in Figure 6.5. The four elements in the first grid rule out all the other elements in the first grid. The element $(1,1,1)^T$ in the second grid paired up with each each element in the first grid rules out the corner points of the third grid. This leaves a potential deck of 8 cards in the second grid plus 5 cards in the third grid for $A_1 \cup A_2$. The code in Appendix B.3 asks for an input size $|A_1| = |A_2|$, then makes two subsets of the potential deck of 13 cards with this size, and stops once it finds that A_0 together with the two subsets form an extendable collection. For sizes $|A_1| = |A_2| = 1$ up to 4 extendable collections are found. There are no extendable collections with $|A_0| = 5$ and $|A_1| = |A_2| \geq 5$.

6.6. Experimenting with the extended product construction

Up to (and including) dimension 6, the maximum size of a cap is known. One may wonder if the extended product construction can be used to find some of the already known maximum caps in lower dimensions. Is it for example possible to apply a 2-dimensional admissible set to an extendable collection in $\mathbb{F}_3^2$ to find a cap in $\mathbb{F}_3^4$ of the maximum size 20, or at least a size close to 20? In this section we experiment with applying the extended product construction to try to find (close to) maximum caps in lower dimensions.

Trying to Find a Cap in $\mathbb{F}_3^4$

Let us first try to use the extended product construction to find a cap in $\mathbb{F}_3^4$ of a size 20. The extended product construction finds a cap in $\mathbb{F}_3^{nm}$, where n is the dimension of the extendable collection we start with, and m is the dimension of the admissible set we use to extend the collection. Since $4 = 2 \cdot 2$, we are looking for an extendable collection and an admissible set that are both two-dimensional. Of course also $4 = 1 \cdot 4$, but $\mathbb{F}_3^1$ has no (useful) extendable collections (see Section 6.5) and applying a one-dimensional admissible set is useless as this amounts to just taking one of the caps A_0, A_1 or A_2.

Now, let $A_0, A_1, A_2 \subset \mathbb{F}_3^2$ be an extendable collection. In Section 6.3 we found that the only two-dimensional admissible sets have constant weight 1. Thus in any case we apply $S = I(2,1)$ and then the resulting cap has size $2|A_0||A_1|$. In Section 6.5 we found that the extendable collections with $|A_0| = 1$ and $|A_1| = |A_2| = 4$ and the extendable collections with $|A_0| = |A_1| = |A_2| = 2$ were the best options. In both cases the result is a 4-cap of size 8, which is way smaller than 20.

Trying to Find a Cap in $\mathbb{F}_3^6$

Let us now try to use the extended product construction to find a cap in $\mathbb{F}_3^6$. The best direct product construction we can make to obtain a cap in $\mathbb{F}_3^6$ is by taking the direct product of a maximum cap in $\mathbb{F}_3^1$ and a maximum cap in $\mathbb{F}_3^5$, resulting in a cap of size $2 \cdot 45 = 90$. Here, we use caps from different dimensions, while the extended product construction always combines caps in the same dimension. If we restrict ourselves to taking the direct product of caps with equal dimension, then the best we can do is taking the direct product of maximum caps in $\mathbb{F}_3^3$, resulting in a cap of size $9 \cdot 9 = 81$. Thus, we want to use the extended product construction to find a cap of size at least 81, but ideally even at least 90.

Since $6 = 2 \cdot 3$ there are two ways to use the extended product construction:

1. Starting with a 2-dimensional extendable collection and applying a 3-dimensional admissible set: Let $A_0, A_1, A_2 \subset \mathbb{F}_3^2$ be an extendable collection. In Section 6.4 we found admissible sets $I(3,2)$ and $I(3,1)$, so $B = S(A_0, A_1, A_2) \subset \mathbb{F}_3^6$ has size either $3|A_0||A_1|^2$ or $3|A_0|^2|A_1|$. But again the best extendable collections we can find in $\mathbb{F}_3^2$ have $|A_0| = 1$ and $|A_1| = |A_2| = 4$, or $|A_0| = |A_1| = |A_2| = 2$, thus the best result we find is $|B| = 3 \cdot 1 \cdot 4^2 = 48$, which is way smaller than 81.

2. Starting with a 3-dimensional extendable collection and applying a 2-dimensional admissible set: Let $A_0, A_1, A_2 \subset \mathbb{F}_3^3$ be an extendable collection. In Section 6.3 we found that all two-dimensional admissible sets have size 2 and constant weight 1. Then $B = S(A_0, A_1, A_2) \subset \mathbb{F}_3^6$ has size $2|A_0||A_1|$. In Section 6.5 we found that the best extendable collections have $|A_0| = 4$ and $|A_1| = |A_2| = 5$ or $|A_0| = 5$ and $|A_1| = |A_2| = 4$. In both cases $|B| = 40$, which is again not merely close enough to the maximum cap size in $\mathbb{F}_3^6$.

Trying to Find a Cap in $\mathbb{F}_3^8$

Lastly, let us try to use the extended product construction to find a cap in $\mathbb{F}_3^8$. Using the direct product construction to obtain a cap in $\mathbb{F}_3^8$, while restricting ourselves to taking caps with equal dimension, results in either taking two maximum caps in $\mathbb{F}_3^4$ or taking four maximum caps in $\mathbb{F}_3^2$. This results in caps of sizes $20^2 = 400$ and $4^4 = 256$, respectively. Since $8 = 2 \cdot 2 \cdot 2$ we will start with an extendable collection in $\mathbb{F}_3^2$ and apply $\tilde{I}(2,1)$ twice.

In Section 6.5 the best extendable collections in $\mathbb{F}_3^2$ we found were:

- A_0, A_1, A_2 with $|A_0| = 1$ and $|A_1| = |A_2| = 4$, which yields a cap in $\mathbb{F}_3^8$ of size 256.
- A_0, A_1, A_2 with $|A_0| = 2$ and $|A_1| = |A_2| = 2$, which yields a cap in $\mathbb{F}_3^8$ of size 64.
- It was proved that there are no extendable collections in $\mathbb{F}_3^2$ with $|A_0| = 3$.

Now, suppose we do not restrict ourselves to taking an extendable collection with $|A_1| = |A_2|$. Then the general size of the cap $\tilde{B}$ we obtain is $|A_0|(|A_1| + |A_2|)(|A_1|^2 + |A_2|^2)$. If $|A_0| = 1$ we can never do better than $|A_1| = 4$ and $|A_2| = 4$, so the best cap remains one of size 256. If $|A_0| = 2$, we could try $\{|A_1|, |A_2|\} = \{2,3\}$, $\{2,4\}$, or $\{3,4\}$ to improve the size of the cap obtained. However, there are no extendable collections of these sizes in $\mathbb{F}_3^2$ (see the code in Appendix B.2). Thus the best result is the cap of size 256, which is equal to the size of the cap obtained by taking the direct product of four maximum caps in $\mathbb{F}_3^2$. This is the closest result in our experiments, but it is still a lot smaller than the best we can do with the direct product construction by taking two maximum caps in $\mathbb{F}_3^4$.

In conclusion, using the extended product construction to find caps in $\mathbb{F}_3^4$, $\mathbb{F}_3^6$, or $\mathbb{F}_3^8$ does *not* lead to better results than the direct product construction.

7

Conclusion and discussion

This thesis gives a comprehensive overview of what is known about constructions for the cap set problem. In dimensions 1 through 6 there exist explicit constructions of caps that are proved to be of maximum size, from dimension 7 up we know some constructions for caps, but none are proved to be of maximum size yet. To compare caps in different dimensions we use the solidity of caps ($\sqrt[d]{|C|}$). To obtain information about the maximum cap size in high dimensions we consider which number the solidity of caps approaches as the dimension grows larger: the asymptotic solidity. The cap set problem then amounts to finding asymptotic bounds on the size of caps. This thesis mainly focused on improvements of the asymptotic lower bound by means of the extended product construction (EPC).

The EPC, due to Edel, extends extendable collections of cap sets by constant weight (recursively) admissible sets. Tyrrell enhanced the methods of Edel by mimicking the extended product construction to find large admissible sets and found the best lower bound for the time. Researchers of Google then found a better lower bound by cause of the discovery of new constant weight admissible sets with the help of a large language model. This lower bound is currently the best known lower bound. This result combined with the upper bound found by Ellenberg en Gijswijt, leads to the following conclusion about the current best bounds on the asymptotic solidity c:

$$2.2202 \leq c \leq 2.2756$$

If Tyrrell's conjecture, which states the existence of constant weight admissible sets for all weights smaller than the dimension, is true, the asymptotic lower bound would be improved to 2.2331. The EPC makes use of the existence of a recursively admissible set of constant weight $k-1$ in any dimension k, which has been proven. In this thesis we in addition proved that there exist constant weight admissible sets of weight $w = 0, 1, 2, 3, k$ for all dimensions larger than w. All of these sets are also recursive, with exception of the admissible set of weight 3, which is recursive for $k \geq 6$.

How useful the EPC may be to construct caps in high dimensions, it turns out not to be useful to construct caps in low dimensions. In this thesis we have established all extendable collections in one, two, and three dimensions. Starting from these we deduced that it is not possible to construct maximum caps in dimensions 4, 6, and 8.

Considering the gap between the lower and upper bound on the asymptotic solidity, the cap set problem has not yet been solved. As large language models evolve, it might be possible to obtain new breakthroughs in constructing large caps and admissible sets by creating better priority functions. Moreover, as stepping stones to proving Tyrrell's conjecture, it would be useful to further research constructions of constant weight admissible sets that exist in any dimension. Another suggestion for further research is to continue this thesis' attempt to construct an admissible set $I(k, k-2)$ that works for any $k \geq 3$, where either the idea of using blocks of $\tilde{I}(k, k-1)$ could be continued or one may come up with another construction.

Bibliography

[1] M. Falco. Marsha jean falco - the creative genius behind set.

[2] J.A. Grochow. New applications of the polynomial method: The cap set conjecture and beyond. *The american mathematical society*, 56:29–64, 2019.

[3] B.L. Davis and D. Maclagan. The card game set. *The Mathematical Intelligencer*, 25:33–40, 2003.

[4] G. Pellegrino. The maximal order of the spherical cap in S(4,3). *Matematiche*, 25:149–157, 1971.

[5] Y. Edel, S. Ferret, I. Landjev, and L. Storme. The classification of the largest caps in AG(5,3). *Journal of Combinatorial Theory, Series A*, 99:95–110, 2002.

[6] A. Potechin. Maximal caps in AG(6,3). *Designs, Codes and Cryptography*, 46:243–259, 2008.

[7] J. Ellenberg and D. Gijswijt. On large subsets of $\mathbb{F}_q^n$ with no three-term arithmetic progression. *Annals of Mathematics*, 185:339–343, 2017.

[8] E. Croot, V.F. Lev, and Pach. P.P. Progression-free sets in $\mathbb{Z}_4^n$ are exponentially small. *Annals of Mathematics*, 185:331–337, 2017.

[9] F. Tyrrell. New lower bounds for cap sets. *Discrete Analysis*, 2023(20), 2023.

[10] Y. Edel. Extensions of generalized product caps. *Designs, Codes and Cryptography*, 31:5–14, 2004.

[11] B. Romera-Paredes, M. Barekatain, A. Novikov, and et al. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2024.

[12] Y. Edel and J. Bierbrauer. Large caps in small spaces. *Designs, Codes and Cryptography*, 23:197–212, 2001.

A

Finding Admissible Sets

A.1. Settings and Functions

```
import math
from itertools import *
from sys import *
from collections import *

########## IS ADMISSIBLE (function) ##########
# INPUT:   the dimension m
#          a subset of {0,1,2}^m, given as a list of lists
# OUTPUT:  returns True if S subset of {0,1,2}^m is admissible and False if it is not

def is_admissible(S,m):
    admissible_set = False

    # check Adm1:
    naughty_pair=[]
    for x,y in permutations(S,2):
        for i in range(m):
            if x[i]==0 and y[i] != 0:
                break
        else:
            naughty_pair.append([x,y])

    # check Adm2:
    naughty_triple=[]
    for x,y,z in combinations(S,3):
        for i in range(m):
            coordinates = [x[i],y[i],z[i]]
            count0 = coordinates.count(0)
            count1 = coordinates.count(1)
            count2 = coordinates.count(2)
            if count0 == 1 and count1 == 1 and count2 == 1:
                break
            if coordinates.count(0) == 2:
                break
        else:
            naughty_triple.append([x,y,z])
    if not naughty_pair and not naughty_triple:
        admissible_set = True
    return admissible_set

########## IS RECURSIVE (function) ##########
# INPUT:   the dimension m
#          a subset of {0,1,2}^m, given as a list of lists
# OUTPUT:  returns True if S subset of {0,1,2}^m is recursive and False if it is not

def is_recursive(S,m):
    naughty_list = []
```

```

    for x,y in combinations(S,2):
        found_zero_zero = False
        found_zero_one = False
        found_zero_two = False

        for i in range(m):
            coordinates = {x[i],y[i]}
            if coordinates == {0,0}:
                found_zero_zero = True
                break
            if coordinates == {0,1}:
                found_zero_one = True
            if coordinates == {0,2}:
                found_zero_two = True
            if (found_zero_one == True) and (found_zero_two == True):
                break
        else:
            naughty_list.append([x,y])
    return not naughty_list

########## IS CONSTANT WEIGHT (function) ##########
# INPUT:   a subset of {0,1,2}^m, given as a list of lists
# OUTPUT:  returns True if S subset of {0,1,2}^m is constant weight and False if it is not

def is_constant_weight(S):
    constant = False
    counts = set()
    for vector in S:
        count = 0
        for i in vector:
            if i == 0:
                count += 1
        counts.add(count)
    if len(counts)==1:
        constant = True
    return constant
```

A.2. Admissible Sets $\tilde{I}(k,2)$ and $\tilde{I}(k,3)$

```
########## BUILD I(m,2) (function) ##########
# INPUT:   the dimension m
# OUTPUT:  builds a recursively admissible set I(m,2)

def build_set(m):
    S = []
    for i in range(m):
        element = [0] * m
        element[i] = 1
        for j in range(i + 1, m):
            new_element = element.copy()
            new_element[j] = 2
            S.append(new_element)
    return S

########## BUILD I(m,3) (function) ##########
# INPUT:   the dimension m
# OUTPUT:  builds a recursively admissible set I(m,3)

def build_set2(m):
    S = []
    for i in range(m):
        element = [0]*m
        element[i] = 2
        for j in range(i+1,m):
            new_element = element.copy()
            new_element[j] = 1
            for k in range(j+1,m):
                new_new_element = new_element.copy()
                new_new_element[k] = 2
```

```
                S.append(new_new_element)
    return S
```

```
########## CHECK I(m,2) AND I(m,3) ARE RECURSIVELY ADMISSIBLE AND CONSTANT WEIGHT (program)
    ##########

m = int(input('Dimension: '))
i = int(input('Do you want to construct I(m,w) with weight 2 or 3? '))
if i == 2:
    S = build_set(m)
if i == 3:
    S = build_set2(m)
print('S =',S)
print('|S| =',len(S))

# check if S is admissible:
if is_admissible(S,m):
    print('S is an admissible set')

# check if S is recursive:
if is_recursive(S,m):
    print('S is recursive')

# check weight of vectors:
if is_constant_weight(S):
    print('S is constant weight')
```

A.3. Admissible Sets in Two and Three Dimensions

```
########## FIND ALL (RECURSIVELY) ADMISSIBLE SETS IN TWO DIMENSIONS (program) ##########

deck = [[0, 0],[0, 1],[0, 2],[1, 0],[1, 1],[1, 2],[2, 0],[2, 1],[2, 2]]
i = int(input('|S|='))
for S in combinations(deck,i):
    if is_admissible(S,2) and is_recursive(S,2):
        print(S,'is recursively admissible')
        continue
    if is_admissible(S,2):
        print(S,'is admissible')
```

```
########## FIND ALL (RECURSIVELY) ADMISSIBLE SETS IN THREE DIMENSIONS (program) ##########

deck = [[0, 0, 0], [0, 0, 1], [0, 0, 2], [0, 1, 0], [0, 1, 1], [0, 1, 2], [0, 2, 0], [0, 2,
    1], [0, 2, 2], [1, 0, 0], [1, 0, 1], [1, 0, 2], [1, 1, 0], [1, 1, 1], [1, 1, 2], [1, 2,
    0], [1, 2, 1], [1, 2, 2], [2, 0, 0], [2, 0, 1], [2, 0, 2], [2, 1, 0], [2, 1, 1], [2, 1,
    2], [2, 2, 0], [2, 2, 1], [2, 2, 2]]

i = int(input('|S|='))
count = 0
count2 = 0
for S in combinations(deck,i):
        if is_admissible(S,3) and is_recursive(S,3):
            count += 1
            print(S,'is recursively admissible')
            continue
        if is_admissible(S,3):
            count2 += 1
            print(S,'is admissible')
print('There are',count,'recursively admissible sets in 3 dimensions (some might be
    symmetrical) of size |S|=',i,'.')
print('There are',count2,'admissible sets in 3 dimensions (some might be symmetrical) of size
     |S|=',i,'.')
```

A.4. Attempt to Construct $I(k, k-2)$

```
########## GENERATE NON-ZERO ROW (function) ##########
# INPUT:   the length of the row n
# OUTPUT:  generates a set {1,2}^n

def generate_set(n):
    elements = [1, 2]
    result = list(itertools.product(elements, repeat=n))
    return result

########## FIND ADMISSIBLE I(5,3) WITH FIXED STRUCTURE USING I(k,k-1) BLOCKS (program)
    ##########

candidate_S = [[0,0,2,2,2],[0,1,0,2,2],[0,1,1,0,2],[0,1,1,1,0],
[None,0,0,2,2],[None,0,1,0,2],[None,0,1,1,0],
[None,None,0,0,2],[None,None,0,1,0],
[None,None,2,0,0]]

options_row1 = generate_set(6)
options_row2 = generate_set(3)

adm_sets_dim5 = []

# for each row option:
for row1 in options_row1:
    S = candidate_S.copy()
    for i in range(4,10):
        for j in range(6):
            S[i][0] = row1[j]
    for row2 in options_row2:
        S2 = S.copy()
        for k in range(7,10):
            for l in range(3):
                S2[k][1] = row2[l]
        if is_admissible(S2,5):
            adm_sets_dim5.append(S2)
print('There are',len(adm_sets_dim5),'admissible sets I(5,3) with the fixed structure.')
```

```
########## FIND ADMISSIBLE I(6,4) WITH FIXED STRUCTURE AND CONTAINING FIXED I(5,3) (program)
    ##########

options_row0 = generate_set(10)
first_five = [[0, 0, 2, 2, 2, 2], [0, 1, 0, 2, 2, 2], [0, 1, 1, 0, 2, 2], [0, 1, 1, 1, 0, 2],
     [0, 1, 1, 1, 1, 0]]

adm_set = [[0, 0, 2, 2, 2], [0, 1, 0, 2, 2], [0, 1, 1, 0, 2], [0, 1, 1, 1, 0], [2, 0, 0, 2,
    2],
           [2, 0, 1, 0, 2], [2, 0, 1, 1, 0], [2, 2, 0, 0, 2], [2, 2, 0, 1, 0], [2, 2, 2, 0,
    0]]

adm_sets_dim6 = []
for m in range(len(options_row0)):
    row0 = options_row0[m]
    build_S3 = []
    for k in range(10):
        lists = adm_set.copy()
        S3_k = lists[k]
        new_element = [row0[k]]+S3_k
        build_S3.append(new_element)
    test_set = first_five + build_S3
    if is_admissible(test_set,6):
        adm_sets_dim6.append(test_set)

print('There are',len(adm_sets_dim6),'admissible sets I(6,4) with the fixed-fixed structure.'
    )
```

```
########## FIND ALL ADMISSIBLE I(6,4) WITH FIXED STRUCTURE (program) ##########

options_row0 = generate_set(10)
```

```
first_five = [[0, 0, 2, 2, 2, 2], [0, 1, 0, 2, 2, 2], [0, 1, 1, 0, 2, 2], [0, 1, 1, 1, 0, 2],
      [0, 1, 1, 1, 1, 0]]

adm_sets_dim6 = []
for adm_set in adm_sets_dim5:
    for m in range(len(options_row0)):
        row0 = options_row0[m]
        build_S3 = []
        for k in range(10):
            lists = adm_set.copy()
            S3_k = lists[k]
            new_element = [row0[k]]+S3_k
            build_S3.append(new_element)
        test_set = first_five + build_S3
        if is_admissible(test_set,6):
            adm_sets_dim6.append(test_set)

print('There are',len(adm_sets_dim6),'admissible sets I(6,4) with the fixed structure.')
```

B Extendable Collections

B.1. Settings and Functions

```
import math
from itertools import *
from sys import *
from collections import *

########## IS CAP SET (function) ##########
# INPUT:   the dimension n
#          a subset of {0,1,2}^n, given as a list of lists
# OUTPUT:  returns True if the subset is a cap set and False if it is not

def is_capset(checkset,n):
    lines = []
    for x,y,z in combinations(checkset,3):
        for j in range(n):
            coordinates = {x[j], y[j], z[j]}
            if len(coordinates) == 2:
                break
        else:
            lines.append([x,y,z])
    return not lines

########## IS NOT A LINE (function) ##########
# INPUT:   the dimension n
#          a triple of elements of {0,1,2}^n, given as a list of lists
# OUTPUT:  returns True if the triple is not a line (i.e. x+y+z is not equal to 0)

def is_not_a_line(triple,n):
    x = triple[0]
    y = triple[1]
    z = triple[2]
    not_a_line = False
    for j in range(n):
        coordinates = {x[j],y[j],z[j]}
        if len(coordinates) == 2:
            not_a_line = True
            break
    return not_a_line

########## IS EXTENDABLE COLLECTION (function) ##########
# INPUT:   the dimension n
#          a list of three subsets of {0,1,2}^n, each a list of lists
#          i = 1 if it needs to be checked if the subsets are cap sets (and i = 0 if we want to skip this step)
# OUTPUT:  returns True if the three sets form an extendable collection

def extendable_check(collection,n,i):
    # check cap sets
```

```
    if i == 1:
        for candidate in collection:
            if not is_capset(candidate,n):
                return False

    A0 = collection[0]
    A1 = collection[1]
    A2 = collection[2]
    extendable = False

    # check Ext1:
    naughty_1 = []
    for x,y in combinations(A0,2):
            for z1 in A1:
                if not is_not_a_line([x,y,z1],n):
                    naughty_1.append([x,y,z1])
            for z2 in A2:
                if not is_not_a_line([x,y,z2],n):
                    naughty_1.append([x,y,z2])

    # check Ext2:
    naughty_2 = []
    for x in A0:
        for y in A1:
            for z in A2:
                if not is_not_a_line([x,y,z],n):
                    naughty_2.append([x,y,z])

    if not naughty_1 and not naughty_2:
        extendable = True
    return extendable

########## FINDS AN EXTENDABLE COLLECTION (function) ##########
# INPUT:   the dimension n
#          A0 (as a list of lists)
#          a deck of potential elements for A1 U A2 (as a list of lists)
#          the desired sizes for A1 and A2
# OUTPUT:  returns the first extendable collection (A0,A1,A2) it finds of the desired sizes
#          for A1 and A2

def find_extendable_collection(A0,deck,n,sizeA1,sizeA2):
    for A1 in combinations(deck,sizeA1):
        if not is_capset(A1,n):
            continue
        for A2 in combinations(deck,sizeA2):
            if not is_capset(A2,n):
                continue
            if A2 == A1:
                continue
            if extendable_check([A0,A1,A2],n,0):
                return [A0,A1,A2]
```

B.2. Finding Extendable Collections in Two Dimensions

```
########## EXTENDABLE COLLECTION IN {0,1,2}^2 WITH |A0| = 1, 2 or 3 (program) ##########
n = 2

print('Choose |A0| = 1, 2 or 3:')
sizeA0 = int(input('|A0| = '))

# |A0|=1
if sizeA0 == 1:
    A0 = [[0,0]]
    deck = [[1,0],[2,0],[0,1],[1,1],[2,1],[0,2],[1,2],[2,2]]

# |A0|=2
if sizeA0 == 2:
    A0 = [[0,0],[1,0]]
    deck = [[0,1],[1,1],[2,1],[0,2],[1,2],[2,2]]

```

```
# |A0|=3
if sizeA0 == 3:
    A0 = [[0,0],[1,0],[0,1]]
    deck = [[1,1],[2,1],[1,2]]

print('Choose sizes for A1 and A2:')
sizeA1 = int(input('|A1| = '))
sizeA2 = int(input('|A2| = '))

collections = []
count = 0

for A1 in combinations(deck,sizeA1):
    if not is_capset(A1,n):
        continue
    for A2 in combinations(deck,sizeA2):
        if not is_capset(A2,n):
            continue
        if A2 == A1:
            continue
        if extendable_check([A0,A1,A2],n,0):
            if [A2,A1] in collections:
                continue
            count+=1
            collections.append([A1,A2])
print('There are',count,'extendable collections with |A0| =',sizeA0,', |A1| =',sizeA1,'and |A2| =',sizeA2)

## uncomment to print the found extendable collections:
# for lst in collections:
#     print(lst)
```

B.3. Finding Extendable Collections in Three Dimensions

```
########## EXTENDABLE COLLECTION IN {0,1,2}^3 WITH |A0| = 4 or 5 (program) ##########
n = 3

print('Choose |A0| = 4 or 5:')
sizeA0 = int(input('|A0| = '))

if sizeA0 == 4:
    A0 = [[0,0,0],[0,2,0],[0,0,2],[0,2,2]]
    deck = [[1,0,0],[1,0,1],[1,0,2],[1,1,0],[1,1,1],[1,1,2],[1,2,0],[1,2,1],[1,2,2],
            [2,0,0],[2,0,1],[2,0,2],[2,1,0],[2,1,1],[2,1,2],[2,2,0],[2,2,1],[2,2,2]]
if sizeA0 == 5:
    A0 = [[0,0,0],[0,2,0],[0,0,2],[0,2,2],[1,1,1]]
    deck = [[1,0,0],[1,0,1],[1,0,2],[1,1,0],[1,1,2],[1,2,0],[1,2,1],[1,2,2],
            [2,0,1],[2,1,0],[2,1,1],[2,1,2],[2,2,1]]

print('Choose sizes for A1 and A2:')
sizeA1 = int(input('|A1| = '))
sizeA2 = int(input('|A2| = '))

print(find_extendable_collection(A0,deck,n,sizeA1,sizeA2))
```