University of Twente

The Rectangle Covering Bound on the Extension Complexity of Small Cut Polytopes

Master Thesis
Applied Mathematics

K.W. Fokkema

Daily Supervisor:
Dr. M. Walter (University of Twente)

Graduation Committee:
Prof.dr. M.J. Uetz (University of Twente)
Dr. M. Walter (University of Twente)
Prof.dr. V. Kaibel (Otto von Guericke University Magdeburg)
Prof.dr. S. Weltge (Technical University of Munich)
Prof.dr. N.V. Litvak (University of Twente)

April 11, 2021

Preface

This report was written at the end of a master's project about investigating lower bounds on the extension complexity of small cut polytopes. Because of the COVID-19 pandemic, all contact with supervisors and committee members was digital, and the report was written from home. I want to thank everyone who helped me finish this challenging task. This mainly includes M. Walter as supervisor and the friends I made at and around the university who kept in touch.

Table of Contents

1 Introduction

In this report we investigate lower bounds for the extension complexity of the cut polytope of size n. At this moment, the best known asymptotic lower bound was found in [1], and is equal to 1.5^{n-1}. However, the best known upper bound is equal to 2^{n-1}. Computing this number for small n can give an idea about where the true relationship lies in this gap. Furthermore, results of our computations might give ideas for a theoretical proof of a better lower or upper bound for larger values of n.

Cut polytopes have been widely studied. One application in which the cut polytope is used is in solving the max-cut problem, which is NP-hard [2]. They are also closely related to correlation polytopes, which are tightly connected to combinatorial problems in the foundations of quantum mechanics, and to the Ising spin model [3].

In chapter 2, we discuss the cut polytope and some of its properties. In chapter 3 we introduce the concept of extension complexity and techniques to find lower bounds for extension complexity. Then in chapter 4 the techniques from chapter 3 are applied to small cut polytopes. We also show when and why our approach fails. Finally, in chapter 5 we reflect on our results and make suggestions for further research.

2 Cut Polytope

Let $G = (V, E)$ denote a finite, undirected simple graph with vertex set V and edge set E. A *cut* of G associated by the set of vertices $W \subseteq V$ is defined as $\delta(W) := \{\{u, v\} \in E \mid \{u, v\} \cap W = 1\}$. For each different cut $\delta(W)$, we define an *incidence* vector $\chi^{\delta(W)}$ of length $|E|$ such that

$$\chi_e^{\delta(W)} = \begin{cases} 1 & \text{if } e \in \delta(W) \\ 0 & \text{otherwise.} \end{cases}$$

The *cut polytope* $P(G)$ is defined as the convex hull of all such incidence vectors. [2] In this report, we restrict our view to cut polytopes of complete graphs, so let K_n denote the complete graph on n vertices and define $P^n = P(K_n)$.

There are some other geometric objects which are closely related to P^n. One of these is the *correlation polytope*, the convex hull of all the rank-1 binary symmetric matrices of size $n \times n$, which is linearly isomorphic to P^n [4]. Another one is the *cut cone* C^n, which is the cone defined by all facets of P^n that also contain the origin.

Since K_n has $d := n(n-1)$ edges (and P^n is full-dimensional), P^n is a d-dimensional polytope. Also, because all vertices of P^n lie on $\{0, 1\}^d$, P^n can be classified as a *0/1-polytope* [5]. Furthermore, there are 2^{n-1} different cuts of K_n, because each cut can be represented by two complementary subsets of vertices. This means that P^n has 2^{n-1} vertices.

2.1 Symmetry

For efficient computations, it is useful to know the symmetries of P^n. There are two types of operations which define an automorphism of P^n [6]: the *permutation operation* and the *switching operation*.

The permutation operation is defined by permuting the vertices of the underlying graph K_n. Using this operation, cuts can be mapped onto each other if and only if they have the same 'size', where the 'size' is defined as the amount of elements in the smallest of the two disjoint subsets defined by the cut:

Definition 2.1 (Size of a cut)**.**

$$\|\delta(W)\| := \min(|W|, |\overline{W}|).$$

The switching operation is defined as follows: a cut $\delta(W)$ can by switched by any cut $\delta(W')$ by taking the symmetric difference $\delta(W) \,\Delta\, \delta(W')$, which is also a cut that can be rewritten as $\delta((W \cup W') \setminus (W \cap W'))$. The switching operation is an automorphism for P^n that maps W' to the origin. Because each cut can be mapped onto the origin this way, this means that P^n 'looks the same' from the perspective of each cut.

It is useful to know the conditions that have to hold for a pair of cuts $(\delta(W_1), \delta(W_2))$ to be mapped onto another pair of cuts $(\delta(W_1'), \delta(W_2'))$.

Proposition 2.2. *There exists an automorphism for the cut polytope that maps* $\delta(W_1)$ *onto* $\delta(W_1')$ *and* $\delta(W_2)$ *onto* $\delta(W_2')$ *if and only if* $\|\delta(W_1) \,\Delta\, \delta(W_2)\| = \|\delta(W_1') \,\Delta\, \delta(W_2')\|$

Proof. Using the switching operation, the first vertex of each pair of cuts can be mapped onto the origin, so that the pairs become $(\delta(\emptyset), \delta(W_1) \,\Delta\, \delta(W_2))$ and $(\delta(\emptyset), \delta(W_1') \,\Delta\, \delta(W_2'))$. Because the permutation operation maps $\delta(\emptyset)$ onto itself, these two pairs can be mapped onto each other using the permutation operation if and only if $\|\delta(W_1) \,\Delta\, \delta(W_2)\| = \|\delta(W_1') \,\Delta\, \delta(W_2')\|$. □

This result motivates the following definition:

Definition 2.3 (Distance between cuts)**.** *We define the distance between cuts* $\delta(W_1)$ *and* $\delta(W_2)$ *to be* $\|\delta(W_1) \,\Delta\, \delta(W_2)\|$.

This definition allows the following interpretation: a pair of cuts can be mapped onto another pair of cuts if the distance between each pair is equal. This distance has the intuitive interpretation of being the least amount of vertices that 'need to change sides' to change one cut to another cut.

Finally, we can also use the distance between cuts to check whether a collection of more than 2 cuts can be mapped onto another collection of the same size: define for each collection a complete weighted subgraph, in which the vertices represent cuts and the weights of the edges represent the distance between the two vertices it is adjacent to. If there exists an isomorphism from one collection of cuts onto another collection of cuts, then there also exists an isomorphism between their weighted graphs.

2.2 Facets

We look at P^n as a function of the variable $\mathbf{x} \in \mathbb{R}^d$. Any face of P^n can then be represented by an inequality $\mathbf{a} \cdot \mathbf{x} \leq \beta$, where $\mathbf{a} \in \mathbb{R}^d$ and $\beta \in \mathbb{R}_{\geq 0}$. The face of P^n that corresponds to this equality is given by $\{\mathbf{x} \in P^n \,|\, \mathbf{a} \cdot \mathbf{x} = \beta\}$. The vertices of P^n that are contained in a facet are called its *roots*. Because P^n is d-dimensional, a face is called a facet when it has a dimension equal to $d-1$. Furthermore, because all vertices of P^n are integer, any facet can be represented by a pair $(\mathbf{a}, \beta)$ such that $\mathbf{a} \in \mathbb{Z}^n$ and $\beta \in \mathbb{N}$ [7].

If $(\mathbf{a}, \beta)$ corresponds to a face and $\mathbf{a} \neq \mathbf{0}$ then β is completely determined by $\mathbf{a}$. Because each element of $\mathbf{x}$ corresponds to an edge in K_n, any facet that is represented by $\mathbf{a}$ can also be represented by a weighted version of K_n, where the edge weights are given by the corresponding elements in $\mathbf{a}$.

In general, finding the facets of P^n for all n is an impossible task, unless NP = co-NP [3]. Furthermore, unless NP = co-NP even determining whether a given pair $(\mathbf{a}, \beta)$ defines a facet of the correlation polytope is NP-hard. Nevertheless, for small values of n all facets of P^n can be enumerated and classified [8]. Here, two facets are considered to be from the same class if and only if there exists a combination of permutation and switching operations such that the facets can be mapped onto each other. In Table 1, an overview is given of the amount of facet-classes and facets up to $n = 9$, and in Table 2 an overview is given for all the facet classes up to $n = 7$, which is based on the SMAPO database [9]. It can be seen that the total amount of facets increases quite rapidly with n. It seems that it might be exponential in the number of vertices of P^n, which is equal to 2^{n-1} and thereby also exponential in n. Ziegler suggested cut polytopes (along with random 0/1 polytopes) as a candidate for the purpose of proving that the number of facets of a 0/1-polytope

Table 1: Number of facet classes and number of facets of P^n for small n[9]. Values marked with (*) are conjectured.

n	# classes	# facets	$\log_2$(# facets)
3	1	4	2.000
4	1	16	4.000
5	2	56	5.807
6	3	368	8.524
7	11	116,764	16.833
8	147*	217,093,472*	27.694*
9	164,506*	12,246,651,158,320*	43.477*

can be exponential in terms of its dimension (which is polynomial in terms of the number of vertices for the cut polytope), but by our knowledge there is no literature showing this for cut polytopes yet.[5]

One type of facet classes that is particularly interesting because of its simple structure is the *hypermetric facets*. Any hypermetric facet is defined by some $\mathbf{b} \in \mathbb{Z}^n$. In Table 2, these facet classes are denoted by $\text{Hyp}_n(\mathbf{b})$. In terms of the representation of a weighted graph, $\mathbf{b}$ gives weights for the vertices of K_n. The weight of each edge (and thereby each entry of $\mathbf{a}$), is subsequently given by the product of the weights of its two adjacent vertices. Therefore, hypermetric facets correspond to an inequality for some β of the following form:

$$\sum_{1 \leq i < j \leq n} \mathbf{b}_i \mathbf{b}_j \mathbf{x}_{ij} \leq \beta. \tag{2.1}$$

Remark 2.4. *Because* $\mathbf{b}$ *and* $-\mathbf{b}$ *correspond to the same facet we can make the assumption while indexing that* $\sum_i \mathbf{b}_i \geq 0$ *to prevent counting duplicates. For hypermetric facets, we always have that* $\sum_i \mathbf{b}_i$ *is odd, which is why assuming* $\sum_i \mathbf{b}_i \geq 0$ *prevents counting any duplicate facets.*

A nice property of hypermetric facets is how the left-hand side (2.1) can be rewritten for $\mathbf{x} = \chi^{\delta(W)}$:

$$\sum_{1 \leq i < j \leq n} \mathbf{b}_i \mathbf{b}_j \chi_{ij}^{\delta(W)} = \sum_{e_{ij} \in \delta(W)} \mathbf{b}_i \mathbf{b}_j = \left(\sum_{v \in W} \mathbf{b}_v \right) \left(\sum_{v \notin W} \mathbf{b}_v \right) = \mathbf{b}(W)\mathbf{b}(\overline{W}). \tag{2.2}$$

The amount of roots a facet can therefore be counted by counting the amount of cuts $\delta(W)$ for which $\mathbf{b}(W)\mathbf{b}(\overline{W})$ is maximal, which will be true when $\mathbf{b}(W)$ is as close to $\frac{1}{2}\sum_i \mathbf{b}_i$ as possible.

When $\mathbf{b} \in \{-1, 0, 1\}^n$, the corresponding facet is called a *pure* hypermetric facet. These facets can be classified further by looking at the amount of nonzeros in $\mathbf{b}$. If there are k such nonzeros, the facet is called *k-gonal* (in general this terminology is used when $\sum_i |\mathbf{b}_i| = k$). Here, k is any odd integer such that $3 \leq k \leq n$. The pure hypermetric facets and some of their properties are further described in section 4.4.

Table 2: Facet types of P^n for small n [9]. Facet names are taken from [8]. The '# roots' column indicates the number of roots in a facet in the corresponding class and the '# facets' column indicates the number of facets in the corresponding class.

n	class name	# roots	# facets
3	Hyp_3(1,1,-1)	3	4
4	Hyp_4(1,1,-1,0)	6	16
5	Hyp_5(1,1,-1,0,0)	12	40
5	Hyp_5(1,1,1,-1,-1)	10	16
6	Hyp_6(1,1,-1,0,0,0)	24	80
6	Hyp_6(1,1,1,-1,-1,0)	20	96
6	Hyp_6(2,1,1,-1,-1,-1)	15	192
7	Hyp_7(1,1,-1,0,0,0,0)	48	140
7	Hyp_7(1,1,1,-1,-1,0,0)	40	336
7	Hyp_7(1,1,1,1,-1,-1,-1)	35	64
7	Hyp_7(2,1,1,-1,-1,-1,0)	30	1344
7	Hyp_7(2,2,1,-1,-1,-1,-1)	26	1344
7	Hyp_7(3,1,1,-1,-1,-1,-1)	21	448
7	Cyc_7(1,1,1,1,1,-1,-1)	21	16128
7	Cyc_7(2,2,1,1,-1,-1,-1)	21	26880
7	Cyc_7(3,2,2,-1,-1,-1,-1)	21	6720
7	Par_7	21	23040
7	Gr_7	21	40320

3 Bounds for Extension Complexity

3.1 Extension Complexity

The *extension complexity* $\mathrm{xc}(P)$ of a polytope P is the minimal number of facets of a polytope P' such that P is a projection of P'. In terms of linear programs, this can also be viewed as the minimal number of inequalities that are necessary to define the feasible region of a linear program. Yannakakis used this connection between polytopes and linear programs to formulate the extension complexity in terms of linear programming [10]. To formulate this connection, first some more standard terminology needs to be introduced.

A *slack matrix* S^P of a polytope P is a matrix in which the rows are indexed by facets of the polytope and columns are indexed by vertices of the polytope. Each element S^P_{vf} indicates the amount of slack vertex v has with respect to the inequality that defines facet f. Intuitively, this slack can be understood as the distance between v and the hyperplane containing f. By definition, the slack matrix is a nonnegative matrix.

The *nonnegative rank* $\mathrm{rk}_+(\,\cdot\,)$ of a nonnegative matrix A is the smallest number of nonnegative rank-1 matrices that sum to A.

Yannakakis showed in [10] that

$$\mathrm{xc}(P) = \mathrm{rk}_+(S^P). \tag{3.1}$$

3.2 Rectangle Covering Bound

Computing the extension complexity by directly computing (3.1) can be very difficult, as finding the nonnegative rank of a matrix is NP-hard [11]. To solve this problem, Yannakakis also introduced a lower bound for the nonnegative rank by considering the *support* of the matrix.

The support of a real matrix A is defined as $\mathrm{supp}(A) = \{(i,j) \mid A_{ij} \neq 0\}$. The lower bound Yannakakis found follows from the fact that the support of a nonnegative matrix is equal to the union of the support of nonnegative rank-1 matrices that sum up to it. Define a *rectangle* of a real matrix A to be the support of a rank-1 matrix that has a support that is contained within the support of A. We will let $\mathrm{rects}(A)$ denote the set of rectangles of A. A *rectangle cover* of A is a set of rectangles such that their union is $\mathrm{supp}(A)$, and the *rectangle covering number* $\mathrm{rc}(\,\cdot\,)$ of A is the minimal amount of rectangles needed to define a rectangle cover for A. The problem of finding such a minimal rectangle cover is also known in literature as *Boolean Matrix Factorization* [12].

The lower bound shown by Yannakakis in [10] is called the *rectangle covering bound*, and is given by

$$\mathrm{rc}(S^P) \leq \mathrm{rk}_+(S^P) = \mathrm{xc}(P). \tag{3.2}$$

The remaining part of this section tries to clarify some of the properties of the rectangle covering number.

Definition 3.1. *Call a rectangle* inclusion-wise maximal *if it is not strictly contained in any other rectangle, and let* $\mathrm{rects}^*(A)$ *denote the set of inclusion-wise maximal rectangles of* A.

Proposition 3.2. *Only inclusion-wise maximal rectangles need to be considered for the purpose of computing the rectangle covering number.*

Proof. Suppose the rectangle cover that contains the smallest amount of rectangles contains a rectangle that is not inclusion-wise maximal. Replacing that rectangle by another rectangle that contains it also yields a valid rectangle cover with the same number of rectangles. Iteratively applying this procedure yields a rectangle cover with the same number of rectangles containing only inclusion-wise maximal rectangles. □

Proposition 3.3. *Let A be a real matrix.* $\mathrm{rc}(A)$ *is invariant under permutations of rows and columns of A.*

Proof. Trivial □

Proposition 3.4. *Let A be a real matrix and let A' be any submatrix of A. Then* $\mathrm{rc}(A') \leq \mathrm{rc}(A)$.

Proof. This follows directly from the fact that any rectangle cover of A also defines a rectangle cover of A'. □

Remark 3.5. *Let A, B be real matrices.* $\mathrm{supp}(A) \subseteq \mathrm{supp}(B)$ *does not imply that* $\mathrm{rc}(A) \leq \mathrm{rc}(B)$

Proof. Counterexample:

$$A = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, B = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

yields $\mathrm{supp}(A) \subseteq \mathrm{supp}(B)$, but $3 = \mathrm{rc}(A) > \mathrm{rc}(B) = 2$. □

Remark 3.6. $\mathrm{rc}(S^P)$ *is not always equal to* $\mathrm{xc}(P)$

Proof. The extension complexity of the matching polytope grows exponentially, but the corresponding rectangle covering number grows polynomially [13]. □

Proposition 3.7. *Let A be a real matrix and let $\mathbf{r}$ be a row vector with the same width as A. If* $\mathrm{supp}(\mathbf{r})$ *is the union of the support of some rows in A, then* $\mathrm{rc}(A) = \mathrm{rc}\left(\begin{pmatrix} A \\ \mathbf{r} \end{pmatrix}\right)$

Proof. Each rectangle cover for A can be adjusted to also be a rectangle cover for $\begin{pmatrix} A \\ \mathbf{r} \end{pmatrix}$ in the following way: Every rectangle that contains a row A_{i*} such that $\mathrm{supp}(A_{i*}) \subseteq \mathrm{supp}(\mathbf{r})$, is extended to also contain elements from the row $\mathbf{r}$. By the construction of $\mathbf{r}$, this new rectangle cover also covers all elements from the new row. Therefore, $\mathrm{rc}(A) \geq \mathrm{rc}\left(\begin{pmatrix} A \\ \mathbf{r} \end{pmatrix}\right)$. Because of Proposition 3.4, $\mathrm{rc}(A) \leq \mathrm{rc}\left(\begin{pmatrix} A \\ \mathbf{r} \end{pmatrix}\right)$, which means $\mathrm{rc}(A) = \mathrm{rc}\left(\begin{pmatrix} A \\ \mathbf{r} \end{pmatrix}\right)$ □

Proposition 3.7 has a useful intuitive meaning in the context of slack matrices, because the row corresponding to a face that is not a facet has the same support as the union of the rows corresponding to the facets in which the face is contained. This is expressed in the following corollary:

Corollary 3.8. *Adding rows to S^P that correspond to faces of P that are not facets does not affect* $\mathrm{rc}(S^P)$.

Proposition 3.9. *Computing the rectangle covering number is NP-hard*

Proposition 3.9 is proven in [14].

3.3 Bounds for the Rectangle Covering Number

Because computing the rectangle covering number is NP-hard (Proposition 3.9), it is useful to look for lower and upper bounds for the rectangle covering number. Because the rectangle covering number can be used as a lower bound for extension complexity by applying (3.2), any lower bound for the rectangle covering number leads to a lower bound for extension complexity. On the other hand, upper bounds for the rectangle covering number do not give upper bounds for the extension complexity, because of Remark 3.6.

After the introduction of some notation, some known upper and lower bounds for the rectangle covering number will be listed [11].

Let $\langle \cdot , \cdot \rangle$ denote the *Frobenius inner product* of two matrices. For real matrices, this is the sum of the elementwise product of the two matrices. Furthermore, for any set of pairs of indices $\{(i_1, j_1), (i_2, j_2), ...\}$ let $\chi(\cdot)$ denote the binary matrix such that $\chi(\mathcal{S})_{ij} = 1 \Leftrightarrow (i, j) \in \mathcal{S}$. To simplify notation, the size of this binary matrix will follow from the context. Some examples: $\chi(\mathrm{supp}(A))$ denotes a binary matrix of the same size as A such that $\chi(\mathrm{supp}(A))_{ij} = 0 \Leftrightarrow A_{ij} = 0$. Also, if $\mathcal{R}$ is a rectangle of A, then $\chi(\mathcal{R})$ denotes a binary matrix of the same size as A such that $\chi(\mathcal{R})_{ij} = 1 \Leftrightarrow (i, j) \in \mathcal{R}$.

Proposition 3.10. *Any rectangle cover defines an upper bound for the rectangle covering number.*

Proof. Trivial. □

Proposition 3.11. *The number of unique rows and the number of unique columns of a matrix are upper bounds for its rectangle covering number.*

Proof. We can construct a rectangle cover that has the required amount of rectangles by letting the rectangles be single rows or columns of the matrix. □

Proposition 3.12 (Fooling Set Bound)**.** *Let A be a real matrix and let $\mathcal{F} \subseteq \mathrm{supp}(A)$. If*

$$\max_{\mathcal{R} \in \mathrm{rects}(A)} |\mathcal{R} \cap \mathcal{F}| = 1,$$

then

$$\mathrm{rc}(A) \geq |\mathcal{F}|. \tag{3.3}$$

Proof. This follows directly from the fact that every element in $\mathcal{F}$ needs to be contained in at least 1 rectangle in the rectangle cover. □

Proposition 3.13 (Generalized Fooling Set Bound). *Let A be a real matrix and let $\mathcal{F} \subseteq \operatorname{supp}(A)$. Then*

$$\operatorname{rc}(A) \geq \frac{|\mathcal{F}|}{\max_{\mathcal{R} \in \operatorname{rects}(A)} |\mathcal{R} \cap \mathcal{F}|}. \tag{3.4}$$

Proof. This follows from the fact that every element in $\mathcal{F}$ needs to be contained in at least 1 rectangle in the rectangle cover. □

We could not find the following bound in any literature, but it is a natural generalization of the generalized fooling set bound, so we call it the *weighted fooling set bound*. It is shown in section 3.4 that this bound is equivalent to the fractional rectangle covering bound, which is well known and also explained in section 3.4 [11].

Proposition 3.14 (Weighted Fooling Set Bound). *Let A be a real matrix and let W be a nonnegative real matrix of the same size as A. Then*

$$\operatorname{rc}(A) \geq \frac{\langle W, \chi(\operatorname{supp}(A))\rangle}{\max_{\mathcal{R} \in \operatorname{rects}(A)} \langle W, \chi(\mathcal{R})\rangle}. \tag{3.5}$$

Proof. Call $\langle W, \chi(\mathcal{R})\rangle$ the weight of rectangle $\mathcal{R}$. Because each element of $\operatorname{supp}(A)$ needs to be contained in at least one rectangle of a rectangle cover and because W is nonnegative, the sum of the weights of the rectangles in a rectangle cover must be at least $\langle W, \chi(\operatorname{supp}(A))\rangle$. Combining this with the maximum weight of a rectangle, which is given by $\max\{\langle W, \chi(\mathcal{R})\rangle \mid \mathcal{R} \text{ rectangle of } A\}$, gives the lower bound for the number of rectangles in a rectangle cover. □

It can be easily seen that the generalized fooling set bound (Proposition 3.13) is a generalization of the fooling set bound (Proposition 3.12), and also that the weighted fooling set bound (Proposition 3.14) is a generalization of both of those bounds.

Remark 3.15. *In Propositions 3.12, 3.13 and 3.14, we can safely replace every instance of* $\max_{\mathcal{R} \in \operatorname{rects}(A)}$ *by* $\max_{\mathcal{R} \in \operatorname{rects}^*(A)}$*, because the arguments are monotonous in the size of the rectangle.*

3.4 Integer Linear Programming Formulation

It turns out that the weighted fooling set bound (Proposition 3.14) can be better understood by looking at (integer) linear programming formulations of finding the rectangle number, which are investigated in this section.

Proposition 3.16 (Rectangle Cover ILP). *The following integer linear program models the the problem of finding the rectangle covering number of a real matrix A:*

$$\textbf{minimize} \quad \sum_{\mathcal{R}} x_{\mathcal{R}} \tag{3.6a}$$

$$\textbf{subject to} \quad \sum_{\mathcal{R}:(i,j)\in\mathcal{R}} x_{\mathcal{R}} \geq 1 \qquad \forall (i,j) \in \operatorname{supp}(A) \tag{3.6b}$$

$$x_{\mathcal{R}} \in \mathbb{Z}_{\geq 0} \qquad \forall \mathcal{R} \in \operatorname{rects}^*(A) \tag{3.6c}$$

Proof (Sketch). The variables $x_{\mathcal{R}}$ model whether rectangle $\mathcal{R}$ is contained in the rectangle cover. Because of Proposition 3.2, any optimal solution can be assumed without loss of generality to only contain inclusion-wise maximal rectangles, so we can restrict the rectangles to be in $\operatorname{rects}^*(A)$. Note that the optimal values of $x_{\mathcal{R}}$ will always be 0 or 1, so $x_{\mathcal{R}} = 1 \Leftrightarrow \mathcal{R}$ is contained in the rectangle cover. The constraints in (3.6b) enforce that each element of $\operatorname{supp}(A)$ is contained in at least 1 rectangle in the rectangle cover and the objective function (3.6a) minimizes the number of rectangles in the rectangle cover. □

The linear programming relaxation of (3.6) very straightforwardly replaces equations (3.6c) by their continuous version:

$$\textbf{minimize} \quad \sum_{\mathcal{R}} x_{\mathcal{R}} \tag{3.7a}$$

$$\textbf{subject to} \quad \sum_{\mathcal{R}:(i,j)\in\mathcal{R}} x_{\mathcal{R}} \geq 1 \qquad \forall (i,j) \in \operatorname{supp}(A) \tag{3.7b}$$

$$x_{\mathcal{R}} \geq 0 \qquad \forall \mathcal{R} \in \operatorname{rects}^*(A) \tag{3.7c}$$

This problem is very similar to finding a rectangle cover, except that fractional rectangles are allowed. For that reason, the solution to (3.7) is called the *fractional rectangle covering number*, denoted $\operatorname{frc}(\,\cdot\,)$ [11]. Because (3.7) is a relaxation of (3.6),

$$\operatorname{frc}(A) \leq \operatorname{rc}(A). \tag{3.8}$$

Proposition 3.17. *The dual of the linear programming relaxation of (3.6) is given by the following linear program:*

$$\textbf{maximize} \quad \sum_{(i,j)\in\operatorname{supp}(A)} w_{(i,j)} \tag{3.9a}$$

$$\textbf{subject to} \quad \sum_{(i,j)\in\mathcal{R}} w_{(i,j)} \leq 1 \qquad \forall \mathcal{R} \in \operatorname{rects}^*(A) \tag{3.9b}$$

$$w_{(i,j)} \geq 0, \qquad \forall (i,j) \in \operatorname{supp}(A) \tag{3.9c}$$

By inspection it turns out that finding a solution to (3.9) is equivalent to finding a weighted fooling set bound (Proposition 3.14), with $W_{ij} = w_{(i,j)}$. To see this, note that the weighted fooling set bound remains the same if W is scaled by some real number, so it is possible scale W in such a way that

$$\max_{\mathcal{R}\in \text{rects}^*(A)} \langle W, \chi(\mathcal{R})\rangle = 1,$$

which is precisely modelled by equations (3.9b). Furthermore, the fact that W is a nonnegative matrix is modelled by equations (3.9c). Finally, strong duality implies that the optimal solution for (3.9) is equal to the optimal solution of (3.7), which is the fractional rectangle covering number, which is a lower bound for the rectangle covering number as shown in (3.8).

The fact that the best weighted fooling bounds are given by the optimal solution to a linear program helps in showing how the task of finding a matrix W can be simplified for a matrix A with many automorphisms in the form of permutations of rows and columns.

Theorem 3.18. *There exists a nonnegative matrix W that defines an optimal weighted fooling set bound and satisfies $W_{ij} = W_{i'j'}$ for any pair $(i, j), (i', j')$ for which there exists a permutation of rows and columns that is an automorphism for A that maps A_{ij} onto $A_{i'j'}$.*

Proof. Let W' be any nonnegative real matrix that defines an optimal weighted fooling set bound and is a solution to (3.9). All permutations of rows and columns that are automorphisms for A define a permutation of W' that also defines an optimal weighted fooling set. These permutations of W' are all solutions to the linear program (3.9), so therefore the average W of all these permutations of W' is also a solution to (3.9) and defines an optimal weighted fooling set. Therefore, we can assume that W satisfies the property that $W_{ij} = W_{i',j'}$ if there exists a permutation of rows and columns that is an automorphism for A that maps A_{ij} onto $A_{i'j'}$. ☐

3.5 Hyperplane Separation Bound

Besides the rectangle covering bound, another closely related lower bound is known for the nonnegative rank of a matrix, called the *hyperplane separation bound*, denoted hsb(·) [13]. It is defined as follows:

Proposition 3.19 (Hyperplane Separation Bound)**.** *Let A be a nonnegative real matrix and let W be a real matrix of the same size as A. Then*

$$\text{rk}^+(A) \geq \frac{\langle W, \frac{A}{\|A\|_\infty}\rangle}{\max\limits_{\mathcal{R}\in \text{rects}(A)} \langle W, \chi(\mathcal{R})\rangle}. \tag{3.10}$$

Notice the similarity to the weighted fooling set bound (3.14). In [13] this bound is used to show that the extension complexity of the matching polytope is exponential. It is also shown there that the rectangle covering bound is polynomial for the matching polytope. Therefore, at least in some cases the hyperplane separation bound is stronger than the rectangle covering bound. In comparison to the weighted fooling set bound, the hyperplane separation bound can be stronger because W is not restricted to be nonnegative. However, when the optimal W for the hyperplane separation bound is already nonnegative, the weighted fooling set bound will be a factor $\|A\|_\infty$ stronger. This might not be a problem for showing that the extension complexity of a polytope is exponential when $\|A\|_\infty$ can be shown to be polynomial at worst.

In [13], the rectangles that are considered for the hyperplane separation bound are allowed to have a support that is not a subset of the support of A. However, for the value of the hyperplane separation bound this makes no difference. To see that this is true, consider some W^* that maximises the right hand side of (3.10). All elements of W^* that correspond to a 0 in A do not appear in the numerator, so they will minimize the denominator. This can be done by sending the value of these elements to negative infinity, which means any rectangle that maximizes the denominator will not contain any of these elements and will therefore be contained within the support of A.

In contrast to bounds for the rectangle covering number, the rectangles in (3.10) cannot be assumed to be inclusion-wise maximal, because when W contains negative elements, the rectangle that maximises the denominator of the right hand side of (3.10) might not be inclusion-wise maximal. This fact makes the hyperplane separation bound harder to compute than the rectangle covering bound. Therefore, it is useful to know any limits where using (lower bounds for the) rectangle covering bound yields equally good results as using the hyperplane separation bound.

The following theorem shows that one case in which this happens is when the hyperplane separation bound is equal to the amount of columns in A. For S^P, this is the case in which the hyperplane separation bound on the extension complexity is equal to the number of vertices of the polytope.

Theorem 3.20. *Let* $A \in \mathbb{R}^{f\times v}_{\geq 0}$*. If* $\operatorname{hsb}(A) = \operatorname{rk}^+(A) = v$*, then also* $\operatorname{frc}(A) = v$*.*

Proof. Let W be the matrix that makes the right hand side of (3.10) equal to v. In the following, we only consider the part of W corresponding to the support of A, because all rectangles we consider are also in the support of A. Decompose $W := W^+ - W^-$ such that $\langle W^+, W^-\rangle = 0$ and $W^+, W^- \in \mathbb{R}^{f\times v}_{\geq 0}$.

Substituting what we have into (3.10) gives

$$v \cdot \max_{\mathcal{R}\in \operatorname{rects}(A)} \langle W, \chi(\mathcal{R})\rangle = \langle W, \frac{A}{\|A\|_\infty}\rangle. \tag{3.11}$$

We will construct a collection of rectangles that satisfy this equation: consider the rectangles that consist of single columns and are contained in $\operatorname{supp}(W^+)$:

$$\mathcal{R}_j := \{(i,j) \mid (i,j) \in \operatorname{supp}(W^+)\}.$$

Summing $\langle W^+, \chi(\mathcal{R}_j)\rangle$ for all $1 \leq j \leq v$ gives

$$\sum_{j=1}^{v} \langle W^+, \chi(\mathcal{R}_j)\rangle = \langle W^+, \operatorname{supp}(A)\rangle,$$

because $\operatorname{supp}(W^+) \subseteq \operatorname{supp}(A)$. We show that both sides of this equation are equal to (3.11) as follows:

$$v \cdot \max_{\mathcal{R}\in \text{rects}(A)} \langle W, \chi(\mathcal{R})\rangle \geq \sum_{j=1}^{v} \langle W, \chi(\mathcal{R}_j)\rangle = \sum_{j=1}^{v} \langle W^+, \chi(\mathcal{R}_j)\rangle$$
$$= \langle W^+, \text{supp}(A)\rangle \geq \langle W^+, \frac{A}{\|A\|_\infty}\rangle \geq \langle W, \frac{A}{\|A\|_\infty}\rangle.$$

The equality on the first line holds because $\bigcup_j \text{supp}(\mathcal{R}_j) = \text{supp}(W^+)$ We conclude that all inequalities must in fact be equalities. Because of the last inequality, this implies that $\langle W^-, A\rangle = 0$. This means that W is nonnegative, in which case $\text{frc}(A) \geq \text{hsb}(A) = v$. Because we know that $v \geq \text{rk}^+(A) \geq \text{frc}(A)$, we conclude that $\text{hsb}(A) = \text{frc}(A) = v$. □

Remark 3.21. *A remarkable detail of the above proof is the fact that $\langle W^+, \text{supp}(A)\rangle = \langle W^+, \frac{A}{\|A\|_\infty}\rangle$ for an optimal hyperplane separation bound equal to v. This would mean that we only have to consider elements of A that are equal to $\|A\|_\infty$. However, this does not make much sense intuitively. For one thing, if we have a matrix with a high nonnegative rank, we would not expect that making one entry of the matrix very large changes much about a good bound for this rnonnegative rank. Furthermore, we might have a slack matrix with nonnegative rank equal to v in which the matrix entries with smaller value are hard to cover with rectangles. This is exactly the situation we will encounter in this report. This raises the question if the hyperplane separation bound could be improved in this limit. The main problem is the factor $\|A\|_\infty$. Maybe this normalization factor could be made smaller by masking a part of the matrix and only considering the difficulty for rectangles to cover the other parts of the matrix. Furthermore, scaling rows and columns of a matrix can make the hyperplane separation bound significantly worse, but it does not change the nonnegative rank at all. Therefore, it would be a nice property to investigate if there exists a stronger bound that does not change when scaling rows and columns.*

4 Rectangle covering bound of the Cut Polytope

4.1 Introduction

In this chapter, we will try to compute the rectangle covering number of S^{P^n} for small values of n. Currently, the best known lower bound for general n by our knowledge is $\mathrm{rc}(S^{P^n}) \geq 1.5^{n-1}$[1] and a trivial upper bound is given by 2^{n-1}. Therefore, it is an open question where in this gap the true relation for the extension complexity of the cut polytope lies.

The main result of this chapter is the following:

Theorem 4.1. *For* $3 \leq n \leq 8$, $\mathrm{xc}(P^n) = \mathrm{rc}(S^{P^n}) = 2^{n-1}$

To get to this result, in section 4.2 algorithms are described to enumerate the inclusion-wise maximal rectangles of a matrix. In section 4.4, we introduce the submatrix on which we will use our bounding techniques. Then our theoretical and computational results are shown in sections 4.5 and 4.6 respectively. In section 4.7 we show how and why our approach fails for larger values of n.

4.2 Enumerating Rectangles

First we introduce Algorithm 1, which computes the set of all inclusion-wise maximal rectangles of a matrix A, given the inclusion-wise maximal rectangles of the submatrix that excludes the last row of A. We assume we have access to the procedures $\mathrm{cols}(\mathcal{R})$ and $\mathrm{rows}(\mathcal{R})$, which will respectively give the set of rows and columns in the rectangle. We will assume that these procedures run in output-linear time. This makes sense because a rectangle can be stored compactly in terms of its rows and columns.

Algorithm 1 Inclusion-wise maximal rectangles iteration step

Input: Matrix $A \in \mathbb{R}^{f \times v}$, the set $\mathcal{Z}'$ of inclusion-wise maximal rectangles of the submatrix of A that excludes the last row.
Output: The set $\mathcal{Z}$ of inclusion-wise maximal rectangles of A
$\mathcal{Z} = \emptyset$
for $\mathcal{R}' \in \mathcal{Z}'$ **do**
  if $A_{f,j} \neq 0$ for all $j \in \mathrm{cols}(\mathcal{R}')$ **then**
    $\mathcal{R} := \mathcal{R}' \cup (\{m\} \times \mathrm{cols}(\mathcal{R}'))$
    add $\mathcal{R}$ to $\mathcal{Z}$
  else
    add $\mathcal{R}'$ to $\mathcal{Z}$
    $J := \mathrm{cols}(\mathcal{R}') \cap \{j \mid M_{f,j} \neq 0\}$
    if $J \neq \emptyset$ **then**
      $I := \{i \mid i \notin \mathrm{rows}(\mathcal{R}') \text{ and } A_{i,j} \neq 0\ \forall j \in J\}$
      if $I = \emptyset$ **then**
        $\mathcal{R} := (\mathrm{rows}(\mathcal{R}') \cup \{f\}) \times J$
        add $\mathcal{R}$ to $\mathcal{Z}$
      end if
    end if
  end if
end for

Note that for Algorithm 1 to work, the (possibly empty) rectangle that consists of all columns also needs to be included in $\mathcal{Z}'$. However, the algorithm does not yield the rectangle that consists of all rows if it is empty. These are small implementation details we will not worry about. We show the correctness of Algorithm 1 by proving the following theorem:

Theorem 4.2. *Let $A \in \mathbb{R}^{f \times v}$ and let A' denote the submatrix of A of size $f-1 \times v$ that excludes the last row of A. Furthermore, let $\mathcal{Z}$ and $\mathcal{Z}'$ denote the set of all inclusion-wise maximal rectangles of A and A' respectively. For any $\mathcal{R} \in \mathcal{Z}$ exactly one of the following cases holds:*

- $\mathcal{R} \in \mathcal{Z}'$
- $\text{rows}(\mathcal{R}) = \text{rows}(\mathcal{R}') \cup \{f\}$ *and* $\text{cols}(\mathcal{R}) = \text{cols}(\mathcal{R}') \cap \{j \mid M_{f,j} \neq 0\}$ *for a unique rectangle* $\mathcal{R}' \in \mathcal{Z}'$

Proof. First note that both cases cannot hold, as all inclusion-wise maximal rectangles that satisfy the first case do not contain row f, but all inclusion-wise maximal rectangles that satisfy the second case do.

Assume for the sake of contradiction that there is an inclusion-wise maximal rectangle $\mathcal{R} \in \mathcal{Z}$ that does not satisfy any of the cases. Since it does not satisfy the first case, $f \in \text{rows}(\mathcal{R})$. We will now construct $\mathcal{R}'$ such that the second case holds.

Let $I := \text{rows}(R) \backslash \{f\}$ and $J := \text{cols}(\mathcal{R}) \cup \{j \mid M_{i,j} \neq 0 \ \forall i \in I\}$. Now, $\mathcal{R}' := I \times J$ is a rectangle. It is also inclusion-wise maximal: we cannot add another row to $\mathcal{R}'$ because $\mathcal{R}$ was inclusion-wise maximal and we cannot add another column to $\mathcal{R}'$ by the construction of J.

Now we want to show that $\mathcal{R}$ satisfies the second case for $\mathcal{R}'$. From the construction of J it follows that $\text{rows}(\mathcal{R}) = \text{rows}(\mathcal{R}') \cup \{f\}$. By construction of J, $\text{cols}(\mathcal{R}) \subseteq \text{cols}(\mathcal{R}')$, and because $\mathcal{R}$ is inclusion-wise maximal the part $\text{cols}(\mathcal{R}) = \text{cols}(\mathcal{R}') \cap \{j \mid M_{f,j} \neq 0\}$ must also be true.

Finally we show that $\mathcal{R}'$ is unique. It is trivial that $I = \text{rows}(\mathcal{R}')$ is determined uniquely by $\mathcal{R}$. Furthermore, $J = \text{cols}(\mathcal{R}')$ is also unique because it is uniquely determined by I and the fact that $\mathcal{R}'$ is an inclusion-wise maximal rectangle. This means that the second case holds, which is a contradiction. Since at least one of the two cases holds and both cannot hold at the same time, we conclude that any $\mathcal{R} \in \mathcal{Z}$ satisfies exactly one of the two cases. □

Now we can easily use Algorithm 1 to write a recursive algorithm that computes the inclusion-wise rectangles of a matrix from scratch. The base case is the only (empty) rectangle for a matrix without any rows. The result is Algorithm 2.

Algorithms 1 and 2 can be easily changed to iterate over rows instead of columns. We can approximate the running time by counting how many rectangles are considered in the for-loop of Algorithm 1. An upper bound is the total amount of inclusion-wise maximal rectangles of A times $\min\{f, v\}$. This suggests that the computation time is smallest by choosing to iterate over the smallest dimension of A. A more realistic computation time is given by the assumption that the amount of rectangles in a submatrix is exponential in the smallest dimension of the submatrix, which gives the same result in the limit of large matrices. These estimates for the computation time are especially useful for very ‘rectangular’ matrices that have one dimension that is much larger than the other dimension.

Algorithm 2 Recursive inclusion-wise maximal rectangles

```
Input: Matrix A ∈ ℝ^{f×v}
Output: The set 𝒵 of inclusion-wise maximal rectangles of A
if f = 0 then
    ℛ := {∅} × {1, 2, ..., v}
    𝒵 = {ℛ}
else
    Let A′be A without its last row
    Obtain the set 𝒵′ of inclusion-wise maximal rectangles of A′ by recursion
    Use Algorithm 1 to obtain 𝒵 from Z′
end if
```

When the goal is to iterate over all the rectangles instead of listing them, Algorithm 1 can also be adapted to generate the rectangles one by one because each rectangle that is found only depends on at most one rectangle of the previous submatrix. This is useful when storage space for the rectangles is limited.

Now, we will look at how to deal with the symmetry in our matrix. This will be very useful when computing the fractional rectangle covering number using the weighted fooling set bound, because of Theorem 3.18. Using this theorem, we can assume that we can assign 'similar matrix entries' the same weight, because we can construct a symmetrical fractional rectangle covering. This motivates the following definition:

Definition 4.3. *We will consider two rectangles* symmetrical *if there is a permutation of rows and columns of A that is an automorphism for A and maps the rectangles onto each other. Otherwise , we will call the rectangles* non-symmetrical*.*

The notion of symmetrical rectangles is useful when we want to find certain properties of all the rectangles of a matrix that do not change for such a mapping. In that case, we only need to iterate over non-symmetrical rectangles. Therefore, we will introduce variations of Algorithm 2 to find all non-symmetrical inclusion-wise maximal rectangles in a (submatrix of a) slack matrix of the cut polytope.

The goal of these algorithms is to find at least one instance of each non-symmetrical inclusion-wise maximal rectangle, while minimizing computation time. In other words, we want break the symmetries of the cut polytope. However, breaking all symmetries might be more computationally intensive than allowing some duplicates of rectangles that are already found. To see this, we look at inclusion-wise maximal rectangles as fully described by their columns. These columns are vertices of the cut polytope, which are described by cuts of K_n. Therefore, to find out if two inclusion-wise maximal rectangles are symmetrical, we need to find out if there exists an isomorphism from one collection of cuts to another collection of cuts (see section 2.1). This is a special case of the graph isomorphism problem. There is no known polynomial time for the graph isomorphism problem, which explains why it might be advantageous to allow some symmetrical rectangles to reduce the computational complexity. The following algorithms are an attempt to reduce the amount of symmetrical rectangles that are obtained, but without too much (computational) effort.

First we use the symmetry of the switching operation (see section 2.1), which implies that columns are equivalent: any column of S^n can be mapped onto the first column of S^n (or a submatrix that has the same property). Because of this, we only have to look for inclusion-wise maximal rectangles containing the first column, so any row that has a 0 in the first column can be discarded. This idea is shown in Algorithm 3, which is a variation of Algorithm 2.

Algorithm 3 Inclusion-wise maximal rectangles for a matrix with equivalent columns

Input: Matrix $A \in \mathbb{R}^{f \times v}$ with equivalent columns
Output: A superset $\mathcal{Z}$ of all non-symmetrical inclusion-wise maximal rectangles of A
if v = 1 **then**
    $\mathcal{R} = \{(i, 1) \mid A_{i1} \neq 0\}$
    $\mathcal{Z} = \{\mathcal{R}\}$
else
    Let A' be A without its last column
    Obtain the set $\mathcal{Z}'$ of inclusion-wise maximal rectangles of A' by recursion
    Use a column-wise version of Algorithm 1 to obtain $\mathcal{Z}$ from Z'
end if

Finally, we will introduce Algorithm 4, which can be used in case we have more information about the symmetries of A. For this purpose, we introduce the notion of different *classes* of nonzeros of A:

Definition 4.4. *We define a class c for each nonzero elements of a matrix A. Two different nonzero elements belong to the same class if and only if there exists an automorphism for A which is a permutation of rows and columns of A that maps one nonzero onto the other.*

Assume we know the set of classes that nonzero entries of A belong to. For a class c and a rectangle $\mathcal{R}$ of the matrix A, there are two options: either the rectangle contains a matrix entry (i, j) that belongs to c, or it does not.

Now we can use the symmetry of the cut polytope from section 2.1. In the first case, we can assume this matrix entry is in the first column of the matrix (which means $j = 1$) because of the switching operation. Therefore, we can eliminate all rows of A that have a 0 in the first column and all columns of A that have a 0 in the i'th row.

In the second case, we can set all entries of A that belong to class c to 0, because this prevents any rectangle from containing an entry that belongs to such a class. This action does not introduce new rectangles, but can make rectangles of A inclusion-wise maximal that were not inclusion-wise maximal before. Because the extra zeros lead to more columns and rows being eliminated in further steps, this is generally a very good trade-off for large classes.

We have not yet considered the symmetry of the cut polytope that is given by the permutation operation. For that reason (but also in general) it is very likely that there exists a direct improvement of Algorithms 3 and 4, which gives fewer rectangles that are not inclusion-wise maximal or equivalent to other rectangles. For the purposes of the research in this report however, these algorithms did suffice, as the main problem for larger matrices is the large amount non-symmetrical inclusion-wise maximal rectangles for larger cut polytopes.

Algorithm 4 Inclusion-wise maximal rectangles for a matrix with equivalent entries

```
1: Input: Matrix A ∈ ℝ^{f×v} with entries belonging to classes in 𝒞
2: Output: A superset 𝒵 of all non-symmetrical inclusion-wise maximal rectangles of A
3: 𝒵 = ∅
4: for c ∈ 𝒞 do
5:     Pick an element (i, 1) that belongs to class c
6:     A' = (A without columns j where A_{i,j} = 0)
7:     Obtain the set 𝒵' of inclusion-wise maximal rectangles of A' by using Algorithm 3.
8:     𝒵 = 𝒵 ∪ 𝒵'
9:     Make all entries of A that belong to c equal to 0
10: end for
```

4.3 Direct Computation

Using Algorithm 2, we can list all inclusion-wise maximal rectangles of a given matrix. Using those rectangles we can compute the rectangle covering number of the matrix directly by solving the integer linear program (3.6) with a solver. Because the amount of inclusion-wise maximal rectangles of S^n grows fast (see Table 3), this is only a feasible way to find the rectangle covering number of S^n for small n. It can be deduced from Table 3 that finding the rectangle covering number of S^6 this way means solving an integer linear program with 417400 variables.

Table 3: Size and number of inclusion-wise maximal rectangles of S^n for small n. The number of rows follows from Table 1.

n	# columns	# rows	# inclusion-wise maximal rectangles
3	4	4	4
4	8	16	24
5	16	56	352
6	32	368	417400
7	64	116,764	?

The rectangle covering numbers of S^n for $3 \leq n \leq 6$ turn out to be exactly equal to 2^{n-1} when these computations are done, which is equal to the amount of columns of S^n and the upper bound for the extension complexity of P^n. This motivates looking at ower bounds for the rectangle covering number for higher values of n that can be calculated more easily. We will simplify the bound in 2 ways: we will restrict ourselves to the pure hypermetric submatrix of S^n, and we will look at the lower bounds for the rectangle covering number (described in section 3.3) for that matrix instead of the rectangle covering number itself.

4.4 Pure hypermetric facets

To find lower bounds for the rectangle covering number of S^n, a very useful strategy is to find lower bounds for a submatrix of S^n. This is especially true because the full description of the facets of the cut polytope (and thereby the rows of S^n) is only known for small n [8] [15]. Furthermore, the amount of rows in the submatrix can be much smaller, which makes computations easier.

In general, we are looking for a submatrix that has useful properties which we can use for our computations, like a high degree of symmetry. Furthermore, we are looking for a submatrix that can be expected to have a high rectangle covering number, otherwise the rectangle covering number of the submatrix might be lower than the rectangle number of the entire matrix. In [1], where the rectangle covering number of S^n is shown to be at least 1.5^{n-1} the submatrix of choice is the *unique disjointness matrix*. More information about how the unique disjointness matrix is connected to our work can be found in Appendix A. We will restrict ourselves instead to the submatrix consisting of pure hypermetric inequalities, which we will call H^n. These matrices have a lot of symmetry and can also be expected to have a high rectangle number, because pure hypermetric facets contain a lot of roots (see Table 1), which correspond to zeros in the slack matrix (and matrices with too few zeros generally have a low rectangle covering number [11]). Finally, the rows of these matrices can easily be obtained. More information about how we obtained these rows, but also rows of S^n in general, can be found in Appendix B.

First, some more properties of the pure hypermetric facets are discussed. Recall from section 2.2 that these facets are described by a vector $\mathbf{b} \in \{-1, 0, 1\}^n$, and that the facet is called k-gonal if the number of nonzeros in this vector $\mathbf{b}$ is equal to k, for any odd k such that $3 \leq k \leq n$. We will define $H^{n,k}$ to be the submatrix of H^n that only contains the rows corresponding to k-gonal facets. Because each row of H^n corresponds to a vector $\mathbf{b}$ and each column of H^n corresponds to a cut defined by a set of nodes W, we will index H^n and $H^{n,k}$ as $H^{n,k}_{\mathbf{b},W}$.

Remark 4.5. *Because W and $\overline{W}$ correspond to the same cut we can assume $\mathbf{b}(W) > \frac{1}{2}\sum_i \mathbf{b}_i$ to prevent counting duplicates. This is similar to Remark 2.4.*

Proposition 4.6. *$H^{n,k}$ is a matrix with $\binom{n}{k}2^{k-1}$ rows and 2^{n-1} columns.*

Proof. The amount of columns follows from the amount of vertices of the cut polytope. To count the number of rows, we count all vectors $\mathbf{b}$ that yield a different facet. There are $\binom{n}{k}$ ways to choose k nonzero elements in $\mathbf{b}$ and there are 2 options for each nonzero: 1 or -1. We assume $\sum_i \mathbf{b}_i > 0$ because of Remark 2.4, which gives a total of $\binom{n}{k}2^{k-1}$ k-gonal pure hypermetric facets. This number is consistent with Table 2. □

Because for each n the number of vectors $\mathbf{b} \in \{-1, 0, 1\}^n$ is equal to 3^n, the total amount of pure hypermetric facets is at most exponential in n. When compared to the values in Tables 1 and 2, this seems to be much smaller than the total number of facets of P^n.

We want to compute the entries of the slack matrix directly. From section 2.2 we know the inequality (2.1) in combination with the expression for hypermetric inequalities (2.2), which gives

$$\mathbf{b}(W)\mathbf{b}(\overline{W}) \leq \beta(\mathbf{b}). \tag{4.1}$$

For pure hypermetric facets we can work this out a bit more:

$$\mathbf{b}(W)\mathbf{b}(\overline{W}) = \frac{1}{4}\left((\mathbf{b}(W)+\mathbf{b}(\overline{W}))^2 - (\mathbf{b}(W)-\mathbf{b}(\overline{W}))^2\right) \tag{4.2a}$$

$$=\frac{1}{4}\left(\left(\sum_i \mathbf{b}_i\right)^2 - (\mathbf{b}(W)-\mathbf{b}(\overline{W}))^2\right) \tag{4.2b}$$

$$\leq\frac{1}{4}\left(\left(\sum_i \mathbf{b}_i\right)^2 - 1\right), \tag{4.2c}$$

where the last inequality holds because $\sum_i \mathbf{b}_i$ is odd. Since $\mathbf{b} \in \{-1,0,1\}^n$, we can always find some W for any $\mathbf{b}$ such that $(\mathbf{b}(W)-\mathbf{b}(\overline{W}))^2 = 1$, we conclude that

$$\beta(\mathbf{b}) = \frac{1}{4}\left(\left(\sum_i \mathbf{b}_i\right)^2 - 1\right). \tag{4.3}$$

Using (4.1) and (4.3) we can compute entries of the slack matrix $H^{n,k}$:

$$H^{n,k}_{\mathbf{b},W} = \beta(\mathbf{b}) - \mathbf{b}(W)\mathbf{b}(\overline{W}) = \frac{1}{4}\left((\mathbf{b}(W)-\mathbf{b}(\overline{W}))^2 - 1\right) \tag{4.4a}$$

$$= \frac{1}{4}\left(\left(2\mathbf{b}(W) - \sum_i \mathbf{b}_i\right)^2 - 1\right) \tag{4.4b}$$

$$= \left(\mathbf{b}(W) - \frac{1}{2}\sum_i \mathbf{b}_i + \frac{1}{2}\right)\left(\mathbf{b}(W) - \frac{1}{2}\sum_i \mathbf{b}_i - \frac{1}{2}\right). \tag{4.4c}$$

Equation (4.4c) gives the intuition that the value of $H^{n,k}_{\mathbf{b},W}$ is quadratic in the value of $\mathbf{b}(W)$. The zeros of $H^{n,k}$ are given by $(\mathbf{b}, W)$ such that $\mathbf{b}(W)$ and $\mathbf{b}(\overline{W})$ are as close together as possible.

Proposition 4.7. *Each column of $H^{n,k}$ contains $\binom{n}{k}\binom{k}{\frac{k-1}{2}}$ zeros.*

Proof. Because of the symmetry of the cut polytope, we know that every column must contain the same amount of zeros, so we will calculate the amount of zeros in the column indexed by the empty cut. Substituting $W = \emptyset$ and $H^{n,k}_{\mathbf{b},W}$ into (4.4b) gives $\sum_i \mathbf{b}_i = 1$ or $\sum_i \mathbf{b}_i = -1$, but we can ignore the latter because of Remark 2.4. Now we need to count the amount of different vectors $\mathbf{b}$ that contain k nonzeros, of which $\frac{k-1}{2}$ are negative and the rest are positive. This gives the result of $\binom{n}{k}\binom{k}{\frac{k-1}{2}}$ zeros per column. □

Proposition 4.8. *$H^{n,k}$ contains $\frac{k-1}{2}$ classes of nonzeros.*

Proof. We will investigate the conditions under which a cut W and a hypermetric facet $\mathbf{b}$, can be mapped onto another cut W' and another hypermetric facet $\mathbf{b}'$, using the symmetries of the cut polytope (section 2.1). Note that we can apply the switching operation to both pairs $(W, \mathbf{b})$ and $(W', \mathbf{b}')$ to obtain $(\emptyset, \tilde{\mathbf{b}})$ and $(\emptyset, \tilde{\mathbf{b}}')$. Next we can apply the permutation operation (which leaves the empty cut invariant) to map $\tilde{\mathbf{b}}$ onto $\tilde{\mathbf{b}}'$ if and only if $\tilde{\mathbf{b}}$ and $\tilde{\mathbf{b}}'$ contain the same number of 1's and -1's. Because both vectors contain k nonzeros, this condition is equivalent to $\sum_i \tilde{\mathbf{b}}_i = \sum_i \tilde{\mathbf{b}}'_i$. We assume both sums are positive because of Remark 2.4, and note that both must be odd because k is odd, which leaves $\frac{k-1}{2}$ possible values for the sums. Therefore, we can conclude that there are $\frac{k-1}{2}$ different classes of pairs $(W, \mathbf{b})$ and therefore $\frac{k-1}{2}$ classes of nonzeros of the matrix $H^{n,k}$. □

Remark 4.9. *If we inspect the number of different values that $H^{n,k}_{\mathbf{b},W}$ can take in (4.4), we can see that there are $\frac{k-1}{2}$ different possible values. Different values in $H^{n,k}$ must correspond to different classes of nonzeros, which leads to the conclusion that each class of nonzeros in $H^{n,k}$ is characterized by its own unique value in that matrix.*

4.5 Theoretical description for $H^{n,3}$

In this section, we will use the generalized fooling set (Proposition 3.13) to show that the rectangle covering number of $H^{n,3}$ is equal to 2^{n-1} when $3 \leq n \leq 6$. In this case, the generalized fooling set $\mathcal{F}$ will be equal to $\text{supp}(H^{n,3})$. To show the generalized fooling set bound, we need to calculate the maximum amount of nonzero elements in a rectangle.

Lemma 4.10. *Let $j \in \mathbb{Z}$ such that $0 \leq j \leq n-3$. Then*

$$\max_{\mathcal{R} \in \text{rects}(H^{n,3})} |\mathcal{R}| = \max_j \binom{n-j}{3} 2^j. \tag{4.5}$$

Proof. Assume without loss of generality that $\mathcal{R} := \arg\max_{\mathcal{R} \in \text{rects}(H^{n,3})} |\mathcal{R}|$ contains the column indexed by $\emptyset$. We assume $\sum_i \mathbf{b}_i > 0$ because of Remark 2.4, so $\sum_i \mathbf{b}_i = 1$ or $\sum_i \mathbf{b}_i = 3$. Let

$$\mathcal{B} := \{\mathbf{b} \,|\, \mathbf{b} \in \{-1, 0, 1\}^n,\ H^{n,3}_{\mathbf{b},\emptyset} \neq 0\}.$$

Using (4.4b), we simplify this to

$$\mathcal{B} = \{\mathbf{b} \,|\, \mathbf{b} \in \{0, 1\}^n,\ \sum_i \mathbf{b}_i = 3\}.$$

Given a subset $\mathcal{B}^1 \subseteq \mathcal{B}$, define

$$\mathcal{W} := \{W \,|\, H^{n,3}_{\mathbf{b},W} \neq 0 \ \forall \mathbf{b} \in \mathcal{B}^1\}.$$

Because of Remark 4.5 and (4.4), $\mathcal{W}$ only contains W satisfying $\mathbf{b}(W) = 3$. Now we see that $\mathcal{W}$ just consists of all cuts that contain all vertices of K_n that are labeled 1 by any $\mathbf{b} \in \mathcal{B}^1$. Let j be the number of vertices that are labeled 0 by all $\mathbf{b} \in \mathcal{B}^1$, then $|\mathcal{W}| = 2^j$. There are $n - j$ vertices left that are labeled 1 by at least some $\mathbf{b} \in \mathcal{B}^1$, which means that we can find $\binom{n-j}{3}$ solutions $\mathbf{b}$ for a given j.

This means that any inclusion-wise maximal rectangle of $H^{n,3}$ has size $\binom{n-j}{3} 2^j$ for some $j \in \{0, 1, ..., n-3\}$. From this, we derive (4.5). □

Table 4: The values of $\binom{n-j}{3}2^j$ for $0 \le j \le n-3$ and $3 \le n \le 7$.

n	j	$\binom{n-j}{3}2^j$	$\binom{n}{3}$
3	0	1	1
4	0	4	4
	1	2	
5	0	10	10
	1	8	
	2	4	
6	0	20	20
	1	20	
	2	16	
	3	8	
7	0	35	35
	1	40	
	2	40	
	3	32	
	4	16	

Theorem 4.11.

$$\mathrm{rc}(H^{n,3}) \ge \begin{cases} 2^{n-1}, & \text{if } 3 \le n \le 6 \\ \frac{8}{5}\binom{n}{3}, & \text{if } n \ge 6 \end{cases} \tag{4.6}$$

Proof. We use the generalized fooling set bound (3.4), where $A = H^{n,3}$ and $\mathcal{F} = \mathrm{supp}(H^{n,3})$. Using Propositions 4.6 and 4.7 we know that $|\mathcal{F}| = \binom{n}{3}2^{n-1}$ and from Lemma 4.10 we know that

$$\max_{\mathcal{R} \in \mathrm{rects}(A)} |\mathcal{R} \cap \mathcal{F}| = \max_j \binom{n-j}{3} 2^j,$$

so we have

$$\mathrm{rc}(H^{n,3}) \ge \frac{\binom{n}{3}2^{n-1}}{\max_j \binom{n-j}{3}2^j},$$

The values of $\binom{n-j}{3}2^j$ and $\binom{n}{3}$ are shown in Table 4 for illustration. For $3 \le n \le 6$, $\max_j \binom{n-j}{3}2^j = \binom{n}{3}$, which gives a rectangle covering bound of 2^{n-1}. For $n \ge 6$, $\max_j \binom{n-j}{3}2^j = 5 \cdot 2^{n-4}$, which gives a rectangle covering bound of $\frac{8}{5}\binom{n}{3}$. □

Corollary 4.12. *For* $3 \le n \le 6$,

$$\mathrm{xc}(P^n) = \mathrm{rc}(S^n) = \mathrm{rc}(H^{n,3}) = 2^{n-1}$$

4.6 Weighted fooling sets for pure hypermetric facets

In the previous section, we have shown that the extension complexity of P^n is equal to 2^{n-1} for $3 \leq n \leq 6$ by using the rectangle covering bound of $H^{n,k}$ for $k = 3$. For any fixed k, this cannot work for arbitrarily large n. This is because the amount of rows of $H^{n,k}$, which is an upper bound for $\mathrm{rc}(H^{n,k})$, grows polynomially (see (4.6)), but 2^{n-1} grows exponentially. Therefore in this section, we include more values for k, and we show that we can generalize the result of the previous section to compute that $\mathrm{xc}(P^n) = 2^{n-1}$ for $3 \leq n \leq 8$, as was given in Theorem 4.1. We will do this doing the weighted fooling set bound. However, in the next section we will show that this is as far as the weighted fooling set bound (or equivalently the fractional rectangle covering number) can lead us when applying it to the submatrices $H^{n,k}$.

We know that $H^{n,k}$ contains $\frac{k-1}{2}$ classes of nonzeros (Proposition 4.8 and that each of these classes has its own unique values in this matrix (Remark 4.9). Furthermore, we know that for the weighted fooling set bound we can assign the same weight to nonzeros of the same class (Theorem 3.18). One strategy to determine what the weights of the different classes should be to give the best weighted fooling set bound, is to iterate over all non-symmetrical rectangles in the matrix (using, for example, Algorithm 4) and count for each of these rectangles how many elements they contain from each class. In a simple case it might be possible to see this way that using some class as the generalized fooling set, the best bound can be obtained. In general, one could construct a linear program to find the best weights.

We implemented this algorithm to decide whether or not $\mathrm{frc}(H^{n,k}) = 2^{n-1}$ for $3 \leq n \leq 9$. The results can be seen in Table 5.

Table 5: The (bounds for the) fractional rectangle covering numbers of $H^{n,k}$, resulting from our computations.

n	$\mathrm{frc}(H^{n,3})$	$\mathrm{frc}(H^{n,5})$	$\mathrm{frc}(H^{n,7})$	$\mathrm{frc}(H^{n,9})$
3	4	-	-	-
4	8	-	-	-
5	16	16	-	-
6	32	32	-	-
7	56	64	32	-
8	<128	128	<128	-
9	<256	<256	<256	<256

We can see that for all $3 \leq n \leq 8$, there is at least one matrix $H^{n,k}$ that has a fractional rectangle covering number equal to 2^{n-1}, but not for $n = 9$. This does not in itself mean that $\mathrm{frc}(H^9) < 2^8$. To see this, consider a rectangle as defined by its columns. Rectangles that cover many elements of $H^{n,k}$ for a specific k might cover very little elements of $H^{n,k'}$ for some $k \neq k'$. However, in the next section we will show that we can find some rectangles that cover enough elements in each $H^{n,k}$ to conclude that $\mathrm{frc}(H^{n,k}) < 2^{n-1}$ for all $n \geq 9$.

In all computations, it turned out that the best bound was obtained by only assigning weight to the class corresponding to the smallest value in $H^{n,k}$. It turns out that nonzeros of higher value are too common in large rectangles. In fact, instead of the more general weighted fooling set bound, the general fooling set gives the same result for the set of all the smallest entries. The advantage of

Table 6: The number of rectangles given by Algorithm 4) for the matrix H^n, in comparison to the amount of rectangles of H^n and S^n without optimization.

n	# rectangles of S^n	# rectangles of H^n	# rectangles checked
3	4	4	1
4	24	24	2
5	352	352	16
6	417400	12536	173
7	?	2121108	9232
8	?	?	3204482

using the weighted fooling set is that we know that we cannot do any better by choosing another set. The fact that the smaller entries of the matrix seem harder to cover can also be seen in the next section.

The total amount of rectangles that we checked with our method is shown in Table 6. For $n = 9$, not all rectangles were checked, but the computation was stopped after 'large enough' rectangles were found. It can be seen that we drastically reduced the total amount of rectangles that we needed to check by using the rectangle enumeration algorithms we designed for symmetrical matrices in section 4.2. However, the total number of rectangles that are checked still increases very fast with n. This is partially because our algorithm does not remove all symmetrical rectangles. However, it is very likely that the total amount of rectangles that are not symmetrical grows very fast by itself, which makes the computations difficult for larger values of n.

4.7 Rectangle coverings of pure hypermetric facets

In this section, we will show limits to bounding the (fractional) rectangle covering number of S^n by the fractional rectangle covering number of H^n. To this end, we construct (fractional) rectangle coverings of H^n.

We look at rectangles in H^n as described by their columns, which are indexed by cuts of K_n. Our goal is to find large rectangles, such that we can cover the entire matrix with less than 2^{n-1} rectangles. We can use our knowledge from section 2.1 to conclude that each non-symmetrical rectangle is defined by the relative distances (as in Definition 2.3) between the cuts that correspond to columns of the rectangle.

We consider 3 different kinds of rectangles: a rectangle described by a single cut that we will call $\mathcal{R}^0$, a rectangle that is described by two cuts that are a distance 1 apart, which we will call $\mathcal{R}^p$, and finally a rectangle that is described by a single cut and all cuts that have distance at most 1 to that single cut, which we will call $\mathcal{R}^*$.

We want to count how many elements from each class of nonzeros in H^n each rectangle contains. From Remark 4.9 we know that there are $\frac{k-1}{2}$ classes of nonzeros in $H^{n,k}$, which correspond to a unique value in $H^{n,k}$. For simplicity, we will number the classes from 1 to $\frac{k-1}{2}$ and indicate this class number with c. In terms of the entries of the slack matrix, given in (4.4), we have the relation $|\mathbf{b}(W) - \mathbf{b}(\overline{W})| = 2c + 1$. Coincidentally, the zero-entries of the slack matrix correspond to $c = 0$.

For $c \geq 1$ we define the counting function

$$\phi(\mathcal{R}, n, k, c),$$

which gives the number of nonzeros of class c in the matrix $H^{n,k}$ that are contained in the rectangle $\mathcal{R}$.

Proposition 4.13. *The number of nonzeros of class c in the matrix H^n, k that are contained in the rectangles $\mathcal{R}^0$, $\mathcal{R}^0$ and $\mathcal{R}^*$ respectively are as follows:*

$$\phi(\mathcal{R}^0, n, k, c) = \binom{n}{k}\binom{k}{\frac{k-1}{2} - c} \tag{4.7a}$$

$$\phi(\mathcal{R}^p, n, k, c) = \begin{cases} 2\binom{n}{k}\binom{k}{\frac{k-1}{2} - c} - 2\binom{n-1}{k-1}\binom{k-1}{\frac{k-1}{2} - c} & \text{if } c = 1 \\ 2\binom{n}{k}\binom{k}{\frac{k-1}{2} - c} & \text{if } c \geq 2 \end{cases} \tag{4.7b}$$

$$\phi(\mathcal{R}^*, n, k, c) = \begin{cases} \frac{k-3}{2}\binom{n}{k}\binom{k}{\frac{k-1}{2} - c} & \text{if } c = 1 \\ \left(n + 1 - k + \frac{k-5}{2}\right)\binom{n}{k}\binom{k}{\frac{k-1}{2} - c} & \text{if } c = 2 \\ (n+1)\binom{n}{k}\binom{k}{\frac{k-1}{2} - c} & \text{if } c \geq 3 \end{cases} \tag{4.7c}$$

The derivation of the expressions in Proposition 4.13 can be found in Appendix C.

Theorem 4.14. *If $n > k + 3$, then* $\operatorname{frc}(H^{n,k}) < 2^{n-1}$.

Proof. We check under which conditions $\phi(\mathcal{R}^p, n, k, c) > \phi(\mathcal{R}^0, n, k, c)$. This is trivially true when $c \geq 2$. Solving $\phi(\mathcal{R}^p, n, k, 1) > \phi(\mathcal{R}^0, n, k, 1)$ yields $n > k + 3$. This means that when $n > k + 3$, a rectangle given by two cuts that are a distance 1 apart contains more elements from $M_j^{n,k}$ than a rectangle given by a single cut, for all values of c. Therefore, if we construct a fractional rectangle covering that consists of all the symmetries of $\mathcal{R}^p$, we can cover all elements using less than 2^{n-1} fractional rectangles. ☐

Theorem 4.15. *If $k > 5$, then* $\operatorname{frc}(H^{n,k}) < 2^{n-1}$

Proof. Similar to the proof of Theorem 4.14, we check under which conditions $\phi(\mathcal{R}^*, n, k, c) > \phi(\mathcal{R}^0, n, k, c)$. This relation is trivially true for $c \geq 2$ and $\phi(\mathcal{R}^*, n, k, 1) > \phi(\mathcal{R}^0, n, k, 1)$ gives $k > 5$, so for any $k > 5$ we can construct a fractional rectangle covering consisting of less than 2^{n-1} fractional rectangles. ☐

All matrices $H^{n,k}$ that can still have $\operatorname{frc}(H^{n,k}) = 2^{n-1}$ despite Theorems 4.14 and 4.15 are $H^{3,k}$ for $3 \leq k \leq 6$ and $H^{5,k}$ for $5 \leq k \leq 8$. For all of these matrices, we indeed have $\operatorname{frc}(H^{n,k}) = 2^{n-1}$, because of our computational results for these values in Table 5.

Next, we extend our results from the matrices $H^{n,k}$ to the matrices H^n by considering affine combinations of the rectangles $\mathcal{R}^p$ and $\mathcal{R}^*$.

Theorem 4.16. *For $n \geq 9$, $\operatorname{frc}(H^n) < 2^{n-1}$.*

Proof. If we can find an affine combination of $\mathcal{R}^p$ and $\mathcal{R}^*$ that covers more nonzero elements of each class than $\mathcal{R}^0$, we can construct a fractional rectangle covering of H^n using less than 2^{n-1} fractional rectangles, similar to our construction in Theorems 4.14 and 4.15.

It is easy to verify that

$$\lambda\phi(\mathcal{R}^p, n, k, c) + (1-\lambda)\phi(\mathcal{R}^*, n, k, 1) > \phi(\mathcal{R}^0, n, k, c)$$

is true for all allowed values of c and $n \geq 9$, for all $\frac{3}{4} < \lambda < \frac{6}{7}$. Therefore, we have found an appropriate affine combination and we conclude that $\operatorname{frc}(H^n) < 2^{n-1}$ for $n \geq 9$. ☐

Finally, we construct a rectangle covering of H^n for large n, to show that not only $\operatorname{frc}(H^n) < 2^{n-1}$ holds for large n, but also $\operatorname{rc}(H^n) < 2^{n-1}$.

Theorem 4.17. *For large enough n, $\operatorname{rc}(H^n) < 2^{n-1}$.*

Proof. It is easy to verify that for $n \geq k \geq 7$, $\phi(\mathcal{R}^*, n, k, c) \geq 2$ for all allowed values of c. Therefore the set of the 2^{n-1} symmetries of $\mathcal{R}^*$ contains each nonzero matrix element of H^n at least twice, except for the submatrices $H^{n,3}$ and $H^{n,5}$.

To make this set a rectangle covering, we need to add a rectangle covering of the submatrices $H^{n,3}$ and $H^{n,5}$. For simplicity, we will choose the rows of these submatrices as rectangles, which are $4\binom{n}{3} + 16\binom{n}{5}$ rectangles in total, by Proposition 4.6. Therefore, we have a rectangle covering of H^n of

$$2^{n-1} + 4\binom{n}{3} + 16\binom{n}{5}$$

rectangles.

We will remove some symmetries of $\mathcal{R}^*$ from our rectangle cover that do not overlap with each other. Because each element was covered at least twice, removing some non-overlapping rectangles still leaves a rectangle cover.

A simple way to choose non-overlapping symmetries of $\mathcal{R}^*$ is to group the vertices of K^n into groups of 3 (or more) vertices and only consider cuts that do not contain any edge between vertices of the same group. This way, the distance between the cuts is at least 3. Now we can remove any symmetries of $\mathcal{R}^*$ that have these cuts as their central cut. These rectangles do not overlap because $\mathcal{R}^*$ only covers matrix elements corresponding to cuts that have distance at most 1 to the central cut, so we are left with a rectangle covering.

The number of rectangles in this rectangle covering is

$$2^{n-1} - 2^{\lfloor \frac{n}{3} \rfloor - 1} + 4\binom{n}{3} + 16\binom{n}{5}.$$

Because $2^{\lfloor \frac{n}{3} \rfloor - 1}$ grows exponentially and $4\binom{n}{3} + 16\binom{n}{5}$ grows polynomially, we know that in the limit of n to infinity, this expression will be less than 2^{n-1}. Numerically, we can verify that this is true for all $n \geq 93$. Therefore, we have constructed a rectangle covering for H^n that has less than 2^{n-1} rectangles for $n \geq 93$ and we have $\operatorname{rc}(H^n) < 2^{n-1}$ for $n \geq 93$. ☐

We have chosen the rectangles $\mathcal{R}^p$ and $\mathcal{R}^*$ because they often turned out to contain many elements in the computations for section 4.6. Geometrically it makes sense to pick a rectangle from which the columns consist of a set of cuts which are close together on the cut polytope, because that maximizes the number of facet-defining inequalities of the cut polytope that have slack for all those cuts together. Therefore, these rectangles and generalizations are good candidates for constructing rectangle coverings for larger submatrices of S^n.

An example for a generalization of $\mathcal{R}^*$ is a rectangle that is described by a single cut and all cuts that have distance at most z to that single cut, where z is some positive number. Another example of a generalization of $\mathcal{R}^*$ would be a single cut and some cuts that have distance at most 1 to that single cut, such that all these cuts are equal for a subset of nodes of size z.

5 Conclusions and Recommendations

In this report, we have shown that the rectangle covering bound on the extension complexity of the cut polytope is equal to 2^{n-1} for n up to 8. We have done this using the fractional rectangle covering number of the slack matrix containing all pure hypermetric facets as a lower bound.

To do the computations necessary for this result, we have first explored the relation between the fractional rectangle covering number and fooling set bounds, in the form of the weighted fooling set bound. We have shown that the hyperplane separation bound is not stronger than the weighted fooling set bound in our limit. Furthermore, we have explored how to use symmetry when computing rectangle cover bounds to greatly reduce the amount of computational work.

Finally, we have used some insights given by the results of our computation to show that the fractional rectangle covering number of the pure hypermetric submatrix is lower than 2^{n-1} for $n \geq 9$, and also that the rectangle covering number of this submatrix is less than 2^{n-1} for large enough n, where $n \geq 93$ suffices.

Our result showing that the hyperplane separation bound is not stronger than the fractional rectangle covering bound when the extension complexity is equal to the number of columns showed some weaknesses of the hyperplane separation bound. The normalization factor in the hyperplane separation bound makes it weaker when scaling rows or columns of the matrix, while this does not make a difference for the nonnegative rank of the matrix. The normalization factor also leads to weaker bounds in matrices in which the smaller entries are hard to cover with rectangles. Therefore, it would be interesting to investigate whether the hyperplane separation bound could be improved to address these issues.

Further research would be needed to decide whether or not the extension complexity of the cut polytope is equal to 2^{n-1} for all n. To get more computational results, our approach for the pure hypermetric facets could perhaps be applied to a larger subset of faces or facets. The set of all hypermetric facets seems like an obvious candidate for this purpose. However, because of the greater amount of facets and the lower degree of symmetry, this is a computationally challenging task, and our algorithms would have to be improved to do computations beyond $n = 9$.

Another interesting idea would be to try to construct smaller rectangle coverings for a larger subsets of faces or facets like the hypermetric facets. This might lead to a method to show that the rectangle covering bound is not strong enough to show that the extension complexity of the cut polytope is equal to 2^{n-1}. The downside of this result would be that it still does not give any conclusions about the extension complexity.

Maybe the best chance to decide whether or not the extension complexity of the cut polytope is equal to 2^{n-1} is to theoretically prove that the fractional rectangle covering number of the slack matrix of all hypermetric facets is equal to 2^{n-1}. This is because the space of hypermetric facets (and faces) still has some nice properties and symmetries to be able to make a theoretical argument, given that the pure hypermetric facets are not enough.

A Unique Disjointness Matrix

In [1], the unique disjointness matrix to compute a rectangle covering bound of 1.5^{n-1} for the extension complexity of the cut polytope. This is another choice for a submatrix of the slack matrix of the cut polytope than we have made in the report. In order to see how this matrix compares to our results, we will interpret the unique disjointness matrix in terms of hypermetric inequalities, and also in terms of the cut polytope, as it is usually defined for the correlation polytope. We start with the definition of the facet defining inequalities for the unique disjointness matrix. For this purpose, we will index the columns of the slack matrix of the cut polytope with all subsets W of $\{1, 2, \ldots, n-1\}$. This indeed gives all possible cuts $\delta(W)$.

Proposition A.1. *For each $A \subseteq \{1, 2, \ldots, n-1\}$, there exists a face of the cut polytope such that the cut $\delta(W)$ is contained in the face if and only if $|W \cap A| = 1$.*

Proof. Fix A and consider the hypermetric inequality with characteristic vector $\mathbf{b}$, such that

$$\mathbf{b}_i = \begin{cases} 1 & \text{if } i \in A \\ 2 - |A| & \text{if } i = n \\ 0 & \text{otherwise.} \end{cases}$$

Because $\sum_i \mathbf{b}_i = 2$ we have $\mathbf{b}(W)\mathbf{b}(\overline{W}) = \mathbf{b}(W)(2 - \mathbf{b}(W)) \leq 1$ for any W. Furthermore, if $\mathbf{b}(W)\mathbf{b}(\overline{W}) = 1$ then we have $\mathbf{b}(W) = 1$. Because W does not contain node n, this means $|W \cap A| = 1$. Therefore, the hypermetric inequality $\mathbf{b}(W)\mathbf{b}(\overline{W}) \leq 1$ satisfies our property. □

This result means that the faces that define the unique disjointness matrix are hypermetric. They are not pure hypermetric faces, which means the results elsewhere in this report cannot be used to draw conclusions about the (fractional) rectangle covering number of the unique disjointess matrix. However, the following is not hard to see:

Proposition A.2. *The rectangle covering number of the unique disjointness matrix is less than 2^{n-1} for any $n \geq 3$*

Proof. In the unique disjointness matrix , the zero-entries correspond to $|W \cap A| = 1$. Therefore, the row indexed $A = \emptyset$ contain only nonzero entries, because there is no W that satisfies $|W \cap \emptyset| = 1$. Consider the submatrix U' of the unique disjointness matrix without this row. U' has $(2^{n-1} - 1)$ rows, so $\text{rc}(A') \leq 2^{n-1} - 1$. Furthermore, each column of U' has at least one nonzero entry: if $|W| \geq 2$ we can choose $A = W$ and if $|W| = 1$ we can pick A to be disjoint from W. Because each column of U' has at least one nonzero, we can extend the rectangle covering of U' to also include the row indexed by $A = \emptyset$, which means the rectangle covering number of the unique disjointness matrix is less than 2^{n-1}. □

Because the rectangle covering number of the unique disjointness matrix is less than 2^{n-1}, it makes sense to invetigate other submatrices of the slack matrix of the cut polytope, like the matrix defined by the pure hypermetric facets that we used in the report. Finally, because of our result in Theorem 3.20, we also know that the hyperplane separation bound on the unique disjointness matrix will be less than 2^{n-1}.

B Computing Slack matrices

To investigate the rectangle covering number of the cut polytope for small values of n, first we need to obtain its slack matrix. In this appendix, 3 methods are described for doing so:

1. Exhaustively computing all facets of the cut polytope with the C library lrslib [16]
2. Using the known classification of facets from SMAPO,[9] and computing all the symmetries of the facets by calculating the symmetry group of cut(n) using nauty.[17]
3. Listing all symmetries of a hypermetric inequality by listing all possible vectors **b** for that inequality.

Each method has his own advantages and disadvantages:

The first method can be used to quickly and independently calculate all facets for small values of n. The downside is that these facets are not classified. Furthermore, the total amount of facets quickly becomes too large for this method to be feasible above $n = 6$.

The second method is the most involved, but can give any class of facets that is known if we can list all symmetries (switching and permutation) of the cut polytope. We can list these symmetries as permutations of the vertices of the cut polytope. This can be done reasonably quickly up to $n = 7$ by using nauty, which can be used to compute generators for the symmetry group of the cut polytope. These generators can be applied recursively to the identity permutation to make a list of all the symmetries. Because nauty requires a vertex-colored graph as input, we need to construct such a graph from the cut polytope, such that any automorphism for the graph is an automorphism for the cut polytope. To do this, the cut polytope can be seen as an edge colored graph, where the colors encode the euclidian distances between the vertices. Subsequently this graph can be converted to a vertex colored graph using a technique described in the nauty user guide, which is shown in Figure 1.

The third method can be used to obtain rows of the slack matrix for only the hypermetric inequalities. This method works for any n as long as the required matrix is not too large. The idea is that we can iterate over all possible vectors **b**. Here we have to keep in mind the permutation and switching operations on the cut polytope. The permutation operation gives pemutations of **b** and the switching operation flips the signs of any set of elements of **b**. Since **b** and $-\mathbf{b}$ give the same inequality, we can assume $\sum_i \mathbf{b}_i \geq 0$ to find less duplicate rows. In fact, we can only find duplicate rows if $\sum_i \mathbf{b}_i = 0$, which can only happen for faces and not for facets, because for hypermetric facets the sum of **b** is always odd.[8]

Isomorphism of edge-coloured graphs. Isomorphism of two graphs, each with both vertices and edges coloured, is defined in the obvious way. An example of such a graph appears at the left of Figure 3.

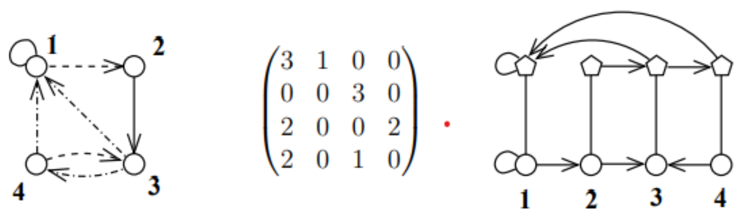

Figure 3: Graphs with coloured edges

In the center of the figure we identify the colours with the integers 1, 2, 3. At the right of the figure we show an equivalent vertex-coloured graph. In this case there are two layers, each with its own colour. Edges of colour 1 are represented as an edge in the first (lowest) layer, edges of colour 2 are represented as an edge in the second layer, and edges of colour 3 are represented as edges in both layers. It is now easy to see that the automorphism group of the new graph (precisely, its action on the first layer) is the automorphism group of the original graph. Moreover, the order in which a canonical labelling of the new graph labels the vertices of the first layer can be taken to be a canonical labelling of the original graph.

More generally, if the edge colours are integers in $\{1, 2, \ldots, 2^d - 1\}$, we make d layers, and the binary expansion of each colour number tells us which layers contain edges. The vertical threads (each corresponding to one vertex of the original graph) can be connected using either paths or cliques. If the original graph has n vertices and k colours, the new graph has $O(n \log k)$ vertices. This can be improved to $O(n\sqrt{\log k})$ vertices by also using edges that are not horizontal, but this needs care.

Figure 1: Technique to write an edge colored graph to a vertex colored graph with "the same" automorphisms.

C Special rectangles calculations

In this appendix, we show the derivation of the formulas given in Proposition 4.13. We will use the formula

$$|(\mathbf{b}(W) - \mathbf{b}(\overline{W})| = 2c + 1$$

for pure hypermetric facets that are defined by $\mathbf{b}$. These vectors $\mathbf{b}$ have $k \leq n$ nonzeros, which are all equal to 1 or -1. Furthermore, we have $1 \leq c \leq \frac{k-1}{2}$. We assume $\sum_i \mathbf{b}_i > 0$ because of Remark 2.4.

We start with $\phi(\mathcal{R}^0, n, k, c)$. Because $\mathcal{R}^0$ consists of a single column and all columns are symmetrical, we can pick the column $W = \emptyset$. This gives $\sum_i \mathbf{b}_i = 2c+1$. Because $\sum_i \mathbf{b}_i > 0$, $\mathbf{b}$ contains k nonzero elements, from which $\frac{k-1}{2} - c$ are equal to -1 (and the other nonzero elements are equal to 1). This gives $\binom{n}{k}\binom{k}{\frac{k-1}{2}-c}$ solutions for $\mathbf{b}$, which means

$$\phi(\mathcal{R}^0, n, k, c) = \binom{n}{k}\binom{k}{\frac{k-1}{2} - c}.$$

Now we consider $\phi(\mathcal{R}^p, n, k, c)$. $\mathcal{R}^p$ consists of 2 columns that are a distance 1 apart. Distance here means the minimal amount of vertices that need to change sides for the cuts to be equal. We pick the columns indexed by $W = \emptyset$ and $W = \{1\}$. Because of symmetry between these two columns, we only need to count the elements in one of the columns and multiply the result by 2. To do this, we consider the result for

$$\phi(\mathcal{R}^0, n, k, c)$$

and count all the vectors $\mathbf{b}$ that are a solution to $\sum_i \mathbf{b}_i = 2c + 1$ but give a zero in the second column, which gives the following system of equations:

$$\begin{cases} \sum_i \mathbf{b}_i = 2c + 1 \\ |\mathbf{b}_1 - \mathbf{b}(\{2, 3, \ldots, n\})| = 1 \end{cases}$$

This system can only be satisfied if $c = 1$ and $\mathbf{b}_1 = 1$. Now we use some combinatorics to find out that the number of vectors $\mathbf{b}$ that satisfy $\mathbf{b}_1$ is equal to $\binom{n-1}{k-1}\binom{k-1}{\frac{k-1}{2}-c}$. Combining all this gives

$$\phi(\mathcal{R}^p, n, k, c) = \begin{cases} 2\binom{n}{k}\binom{k}{\frac{k-1}{2} - c} - 2\binom{n-1}{k-1}\binom{k-1}{\frac{k-1}{2} - c} & \text{if } c = 1 \\ 2\binom{n}{k}\binom{k}{\frac{k-1}{2} - c} & \text{if } c \geq 2. \end{cases}$$

In the case $c = 1$, we could of course had substituted $c = 1$ into the formula, but not doing so stresses the similarities between the terms.

For $\phi(\mathcal{R}^p, n, k, c)$, we consider the column indexed by $W = \emptyset$ and all n cuts that differ in one element. First we will determine the set of vectors $\mathbf{b}$ that give nonzeros in the slack matrix for all these cuts, which means $|\mathbf{b}(W) - \mathbf{b}(W)| \geq 3$ for all W that satisfy $|W| \leq 1$. For any $\mathbf{b}$ that has $\sum_i \mathbf{b}_i = 3$, we can find a W such that $|\mathbf{b}(W) - \mathbf{b}(W)| = 1$ which is no good, so $\sum_i \mathbf{b}_i \geq 5$. For any

$\mathbf{b}$ that has $\sum_i \mathbf{b}_i = 5$, we can never find a W such that $|\mathbf{b}(W) - \mathbf{b}(W)| = 1$, so we conclude that our rectangle consists of exactly the rows indexed by $\mathbf{b}$ such that $\sum_i \mathbf{b}_i \geq 5$.

Given these rows, we calculate $\phi(\mathcal{R}^p, n, k, c)$ by counting how many instances of each class of nonzero occurs in the column indexed by $W = \emptyset$ and $W = \{1\}$. By symmetry, we can multiply the latter by n and add the former to get the correct total.

- In the column $W = \emptyset$, we have the same result as for $\phi(\mathcal{R}^0, n, k, c)$ except when $c = 1$, for which there are no elements.
- In the column $W = \{1\}$, we have $(\sum_i \mathbf{b}_i) - 2\mathbf{b}_1 = 2c + 1$. We need to count the amount of solutions that satisfy $\sum_i \mathbf{b}_i \geq 5$, which gives $2c + 1 + 2\mathbf{b}_1 \geq 5$. Therefore, for $c = 1$, we need to count all vectors $\mathbf{b}$ that satisfy $\sum_i \mathbf{b}_i \geq 5$ and $\mathbf{b}_1 = 1$, for $c = 2$ we need to count all vectors $\mathbf{b}$ that satisfy $\sum_i \mathbf{b}_i \geq 5$ and $\mathbf{b}_1 \geq 0$ and for $c \geq 3$ we have the same result as for $\phi(\mathcal{R}^0, n, k, c)$.

By adding everything and using combinatorial identities to simplify the expression, we obtain

$$\phi(\mathcal{R}^*, n, k, c) = \begin{cases} \dfrac{k-3}{2}\dbinom{n}{k}\dbinom{k}{\frac{k-1}{2} - c} & \text{if } c = 1 \\ \left(n + 1 - k + \dfrac{k-5}{2}\right)\dbinom{n}{k}\dbinom{k}{\frac{k-1}{2} - c} & \text{if } c = 2 \\ (n+1)\dbinom{n}{k}\dbinom{k}{\frac{k-1}{2} - c} & \text{if } c \geq 3. \end{cases}$$

D Iterative facet approach

We started the project which is described in this report with another idea to obtain a submatrix of a slack matrix that has a high rectangle covering number. This idea was to start with a submatrix that consists of a single row and subsequently iteratively add new rows that are likely to increase the rectangle covering number. Such a row could be found by computing a rectangle cover using the minimal amount of rectangles and then picking a row that could not be covered by these rectangles. Therefore, our algorithm was as follows:

Algorithm 6 Iterative rectangle covering bound

```
1: Output: A submatrix M of A that has the same rectangle cover number as A
2: Initialize M as an empty matrix
3: Initialize C as an empty set of rectangles of M.
4: while there exists a row r of A that is not covered by C do
5:     Add row r to the matrix M
6:     Compute the minimal cover C of M
7: end while
```

To compute a minimal rectangle cover in Algorithm 5 (line 6), we used an integer linear programming solver for the integer linear program described in (3.6). To find a row that is not covered by the existing rectangle cover, we used another integer linear program. This last integer linear program consisted of one part that determines a face-defining equality for the cut polytope, which included boolean variables that indicated which vertices of the cut polytope had slack with respect to the face-defining equality. The second part of this integer linear program enforced that the row of the slack matrix that corresponded to this inequality could be not covered by extending rectangles of the existing rectangle cover to this new row.

We used this approach up to $n = 6$, to find submatrices that have a rectangle covering number of 2^{n-1}. The main bottleneck was the integer linear program to find a minimal rectangle cover of each submatrix. The total amount of inclusion-wise maximal rectangles in these submatrices quickly became very large (exponential in the amount of rows of the matrix), which implied solving an integer linear program with millions or billions of variables. Because of the lack of knowledge about the structure of this submatrix, it was hard to optimize this procedure, which is why we abandoned it.

Bibliography

[1] Volker Kaibel and Stefan Weltge. A short proof that the extension complexity of the correlation polytope grows exponentially. *Discrete & Computational Geometry*, 53(2):397–401, Dec 2014.

[2] F. Barahona and A.R. Mahjoub. On the cut polytope. *Mathematical Programming*, 36(2):157–173, June 1986.

[3] Itamar Pitowsky. Correlation polytopes: Their geometry and complexity. *Math. Program.*, 50(3):395–414, April 1991.

[4] Samuel Fiorini, Serge Massar, Sebastian Pokutta, Hans Tiwary, and Ronald Wolf. Linear vs. semidefinite extended formulations: Exponential separation and strong lower bounds. *Proceedings of the Annual ACM Symposium on Theory of Computing*, 11 2011.

[5] Günter M. Ziegler. *Lectures on 0/1-Polytopes*, pages 1–41. Birkhäuser Basel, Basel, 2000.

[6] M. Laurent, M.M. Deza, and V.P. Grishukhin. *The symmetries of the cut polytope and of some relatives*, pages 205–220. Number 4 in DIMACS series in discrete mathematics and theoretical computer science. American Mathematical Society, United States, 1991. Pagination: 16.

[7] Alexander Schrijver. *Theory of Linear and Integer Programming.* John Wiley & Sons, Inc., USA, 1986.

[8] Michel Deza and Monique Laurent. Facets for the cut cone i. *Mathematical Programming*, 56:121–160, 8 1992.

[9] T. Christof. SMAPO. Available at http://comopt.ifi.uni-heidelberg.de/software/SMAPO/cut/cut.html.

[10] Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43(3):441–466, 1991.

[11] Samuel Fiorini, Volker Kaibel, Kanstantsin Pashkovich, and Dirk Oliver Theis. Combinatorial bounds on nonnegative rank and extended formulations. *Discrete Mathematics*, 313(1):67–83, 2013.

[12] Changlin Wan, Wennan Chang, Tong Zhao, Mengya Li, Sha Cao, and Chi Zhang. Fast and efficient boolean matrix factorization by geometric segmentation, 2020.

[13] Thomas Rothvoss. The matching polytope has exponential extension complexity, 2017.

[14] P. Miettinen, T. Mielikäinen, A. Gionis, G. Das, and H. Mannila. The discrete basis problem. *IEEE Transactions on Knowledge and Data Engineering*, 20(10):1348–1362, 2008.

[15] V.P. Grishukhin. All facets of the cut cone cn for n = 7 are known. *European Journal of Combinatorics*, 11(2):115–117, 1990.

[16] David Avis. *A Revised Implementation of the Reverse Search Vertex Enumeration Algorithm*, pages 177–198. Birkhäuser Basel, Basel, 2000.

[17] Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, ii, 2013.