DISSERTATION

Cosystoles and Cheeger Constants of the Simplex

Autor:
Kai Michael RENKEN

Betreuer:
Prof. Dr. Dmitry
FEICHTNER-KOZLOV

Dissertation zur Erlangung des Doktorgrades Dr. rer. nat.

in der Forschungsgruppe

CALTOP
ALTA - Institute for Algebra, Geometry, Topology and their Applications

6. Mai 2019

Zusammenfassung

Das zentrale Thema dieser Arbeit ist, Methoden zu entwickeln, um die kosystolische Norm und die Korandausdehnung einer Kokette bestimmen oder zumindest besser abschätzen zu können. Diese Werte sind die Grundlage für die Bestimmung der Cheeger-Konstanten eines simplizialen Komplexes. Wir entwickeln eine allgemeine Theorie zur kosystolischen Norm von Koketten, in der wir unter anderem eine interessante Verbindung zwischen jener Norm und der Piercing-Zahl eines bestimmten Mengensystems untersuchen (siehe Kapitel 2). In Kapitel 3 beschränken wir unsere Untersuchungen auf eindimensionale Kosystolen eines Simplexes, welche ein wenig einfacher zu verstehen sind, sodass wir in diesem Fall oft bessere Resultate liefern können, unter anderem die explizite Bestimmung der größten eindimensionalen Kosystolen eines Simplexes und eine ungefähre Übersicht, wie alle 1-Kosystolen eines Simplexes im sogenannten kosystolischen Komplex angeordnet sind. In Kapitel 4 lösen wir ein schönes kombinatorisches Anordnungsproblem, welches nicht in direktem Zusammenhang mit dem Rest dieser Arbeit steht, aber während grundlegener Überlegungen darüber aufkam und interessant genug sein sollte, dem Leser ebenso zur Verfügung gestellt zu werden. In Anhang A stellen wir dem Leser den Quelltext zu einer Softwarebibliothek zur Verfügung, die einige der Probleme bezüglich kosystolischer Normen und Cheeger-Konstanten algorithmisch umsetzt und in Anhang B geben wir einen Algorithmus zur konkreten Berechnung der Lösungen des Anordnungsproblems aus Kapitel 4 an.

Abstract

The central interest of this thesis is to develop tools to get hands on the cosystolic norm and the coboundary expansion of a cochain, values which are important to determine the Cheeger constants of a simplicial complex. We develop a general theory about the cosystolic norm of a cochain, in which we, among other small results, study an interesting connection between that norm and the piercing number of a certain set system (see Chapter 2). In Chapter 3 we restrict our research to 1-dimensional cosystoles of a simplex which are slightly easier to understand, so we can provide more explicit results for that case, including the explicit determination of the largest 1-dimensional cosystoles of a simplex and a rough insight, how all 1-cosystoles of a simplex in a certain dimension can be arranged in the so-called cosystolic complex. In Chapter 4 we solve a beautiful combinatorial ordering problem, which is not directly related to the main subject of this thesis but arose during considerations about that and should be worth to be provided to the reader as well. In Appendix A we provide the source code of a software library which handles some of the problems concerning cosystolic norms and Cheeger constants algorithmically and in Appendix B we give an algorithm for the exact calculation of the solutions of the ordering problem from Chapter 4.

Acknowledgements

I want to thank my family, who always encouraged me on my academic way and especially my mum Angela Schneider who always found the right cheering as well as hard words, that helped me through all the hard times that arose on this way. I also want to thank my project advisor Prof. Dr. Dmitry Feichtner-Kozlov, who introduced me to the beautiful subject of Cheeger constants as well as all my colleages at the University of Bremen who brightened every day at the office with enlightening discussions and a warm, welcoming athmosphere. Espacially, I thank Fabian Redecker, who has been a reliable companion through my whole academic and private life. And at the end, very special thanks to Hannah Söger for her love and support at all times.

Contents

Preliminaries

Let us shortly recall some basic algebraic and combinatorial concepts, which we will use within this thesis.

Simplicial Complexes

Let S be some set (whose elements are called **vertices**) and $X \subseteq 2^S$ a family of subsets of S (we will use the notation 2^S for the power set of S within the whole thesis), such that for all $\sigma \in X$ and all $\sigma' \subseteq \sigma$ we have $\sigma' \in X$. Then we call X an **(abstract) simplicial complex**. We just use the notation "simplicial complex" in this thesis, because we will only consider abstract simplicial complexes and are not interested in their geometric realization.

An element of cardinality k of a simplicial complex X is called k-**simplex** and a **face** of a k-simplex $\sigma \in X$ is a $(k-1)$-simplex $\sigma' \in X$, such that $\sigma' \subset \sigma$. Furthermore, we denote the k-**skeleton** of a simplicial complex X by

$$X(k) := \{\sigma \in X : |\sigma| \leq k+1\},$$

and the **uniform** k-**skeleton** of X by

$$X^{(k)} := \{\sigma \in X : |\sigma| = k+1\}$$

Let $\sigma \subset S$ be a simplex and $s \in S \setminus \sigma$, then the simplex constructed by "adding" s to σ is denoted by $(\sigma, s) := \sigma \cup \{s\}$.
For a simplicial complex X and a simplex $\sigma \in X$ we call $\dim(\sigma) := |\sigma| - 1$ the **dimension** of σ and $\dim(X) := \max\{\dim(\sigma) : \sigma \in X\}$ the **dimension** of X (if it exists). Furthermore, for simplices σ and σ', such that $\sigma' \subseteq \sigma$ the **codimension** of σ' in σ is defined as $\dim(\sigma) - \dim(\sigma')$.
A simplicial complex is called **finite** if its vertex set is finite and **finite dimensional** if its dimension is finite.
The most frequently considered simplicial complex in this thesis will be the complex induced by the standard simplex on n vertices. It can be considered as the complete power set of $[n] := |\{i \in \mathbb{N} : 1 \leq i \leq n\}|$ and we will denote it by $\Delta^{[n]} := 2^{[n]}$.

Co- / Chain Complexes & Co- / Homology

Let X be a simplicial complex, then

$$C_k(X, \mathbb{Z}_2) := \left\{ \sum_{i \in I} c_i \sigma_i \, : \, \sigma_i \in X^{(k)}, c_i \in \mathbb{Z}_2 \right\}$$

is called the k-th **chain group** of X, where I is some index set. (The elements of $C_k(X, \mathbb{Z}_2)$ are called k-**chains**)
Note, that in general we have more possible coefficient systems than $\mathbb{Z}_2$ and X can be any topological space, but we will restrict ourselves to simplicial complexes in this thesis. Furthermore, since we only consider chain groups with $\mathbb{Z}_2$-coefficients in this thesis, we will use the notation $C_k(X) := C_k(X, \mathbb{Z}_2)$.
The linear map $\partial_k : C_{k+1}(X) \to C_k(X)$ defined on a simplex $\sigma = (v_0, \ldots, v_{k+1}) \in X$ as

$$\partial_k(\sigma) := \sum_{i=0}^{k+1} (-1)^i (v_0, \ldots, v_{i-1}, v_{i+1}, \ldots, v_{k+1})$$

is called the k-th **boundary map**. Note, that the boundary maps have the property $\partial_{k-1} \circ \partial_k = 0$, so we always have $\mathrm{Im}(\partial_k) \subseteq \ker(\partial_{k-1})$.

The k-th **homology group** of X is defined as

$$H_k(X) := \frac{\ker(\partial_{k-1})}{\mathrm{Im}(\partial_k)},$$

where the elements in $\ker(\partial_{k-1})$ are called k-**cycles** and the elements in $\mathrm{Im}(\partial_k)$ are called k-**boundaries**.
Dualizing this concept, we get the k-th **cochain group** of X by

$$C^k(X) := C^k(X, \mathbb{Z}_2) := \{\varphi : C_k(X) \to \mathbb{Z}_2 : \varphi \text{ is a linear map}\},$$

whose elements are called k-**cochains**, the k-th **coboundary map**
$\delta^k : C^k(X) \to C^{k+1}(X)$ by $\delta^k(\varphi) := \varphi \circ \partial_k$, and the k-th **cohomology group** of X by

$$H^k(X) := \frac{\ker(\delta^k)}{\mathrm{Im}(\delta^{k-1})}$$

Note, that $\delta^k \circ \delta^{k-1} = 0$ holds again which implies that we have $\mathrm{Im}(\delta^{k-1}) \subseteq \ker(\delta^k)$.
Furthermore, the sequence

$$\cdots \xrightarrow{\partial_{k+1}} C_{k+1}(X) \xrightarrow{\partial_k} C_k(X) \xrightarrow{\partial_{k-1}} C_{k-1}(X) \xrightarrow{\partial_{k-2}} \cdots \xrightarrow{\partial_0} C_0(X) \xrightarrow{\partial_{-1}} \{0\}$$

is called a **chain complex** and

$$\{0\} \xrightarrow{\delta^{-1}} C^0(X) \xrightarrow{\delta^0} \cdots \xrightarrow{\delta^{k-2}} C^{k-1}(X) \xrightarrow{\delta^{k-1}} C^k(X) \xrightarrow{\delta^k} C^{k+1}(X) \xrightarrow{\delta^{k+1}} \cdots$$

is called a **cochain complex**.

Let us still introduce the concept of reduced homology / cohomology. If we consider the augmented chain complex

$$\cdots \xrightarrow{\partial_{k+1}} C_{k+1}(X) \xrightarrow{\partial_k} C_k(X) \xrightarrow{\partial_{k-1}} C_{k-1}(X) \xrightarrow{\partial_{k-2}} \cdots \xrightarrow{\partial_0} C_0(X) \xrightarrow{\varepsilon} \mathbb{Z}_2 \longrightarrow \{0\},$$

with $\varepsilon\left(\sum_{i\in I} c_i\sigma_i\right) := \sum_{i\in I} c_i$ instead of the ordinary chain complex, then we call the corresponding homology groups the **reduced homology groups** of X and denote them by $\tilde{H}_k(X)$. Note, that only the 0-th reduced homology group differs from the ordinary 0-th homology group, such that we have $H_0(X) \cong \tilde{H}_0(X) \oplus \mathbb{Z}_2$ and $H_k(X) \cong \tilde{H}_k(X)$ for all $k \geq 1$. Dualizing this concept one can define reduced cohomology analogously.

Since, we are working with $\mathbb{Z}_2$-coeficients only, there is a very intuitive way to talk about chains (cochains, respectively). A k-chain is a linear combination of k-simplices with coefficients in $\mathbb{Z}_2$, so it can just be considered as a subset of the uniform k-skeleton of the underlying simplicial complex X. Even boundaries and coboundaries can be imagined very intuitively as the support of a boundary of a chain exactly consists of those simplices which are contained in an odd number of simplices from the support of the chain and the support of a coboundary exactly exists of those simplices which contain an odd number of simplices from the support of the cochain. Furthermore, there is a one-to-one correspondence between chains and cochains, so to every chain $c \in C_k(X)$ we can associate its characteristic cochain which we denote by $c^* \in C^k(X)$ and for every cochain $\varphi \in C^k(X)$ there exists a unique chain $c \in C_k(X)$, such that we have $c^* = \varphi$.
Let $c \in C_k(X)$ be some chain and $\varphi \in C^k(X)$ some cochain, then we denote the **evaluation** of φ on c as

$$\langle \varphi, c \rangle := \varphi(c) \in \mathbb{Z}_2,$$

and the **support** of φ as

$$\operatorname{supp}(\varphi) := \left\{ \sigma \in X^{(k)} : \langle \varphi, \sigma \rangle = 1 \right\}$$

Furthermore, we define the support of a chain $c \in C_k(X)$ as $\operatorname{supp}(c) := \operatorname{supp}(c^*)$.
Note, that for simplicity we will omit natural inclusion maps of the type
$i : \Delta^{[n]} \longrightarrow \Delta^{[n+d]}$ for some $n, d \in \mathbb{N}$ when adding chains / cochains, so that for $\varphi \in C^k(\Delta^{[n]})$, $\psi \in C^k(\Delta^{[n+d]})$ we will write $\varphi + \psi \in C^k(\Delta^{[n+d]})$ instead of $i(\varphi) + \psi \in C^k(\Delta^{[n+d]})$. From the context it should always be clear what we mean.

Graphs & Hypergraphs

Let V be some set and $E \subseteq \binom{V}{2}$ (we will always use the notation $\binom{V}{k} := \{S \in 2^V : |S| = k\}$ to denote the set of all subsets of cardinality k of a set V). Then the pair $G = (V, E)$ is called a **(simple) graph**, where the elements of V are called **vertices** and the elements of E are called **edges**. Since we only consider simple graphs (undirected graphs with no loops or double edges) in this thesis, we will just call them graphs. Even though we only consider undirected graphs, we want to stick to the common notation and denote an edge by $e = (v, w)$ instead of using set brackets $e = \{v, w\}$. In a graph $G = (V, E)$ two vertices $v_1, v_2 \in V$ are called **adjacent** if we have $(v_1, v_2) \in E$.
Note, that a graph can be considered as a 1-dimensional simplicial complex, where V is the 0-skeleton and E is the uniform 1-skeleton.
According to the terminology of simplicial complexes we call a graph **finite**, if the number of vertices is finite.
A graph $G = (V, E)$ is called **complete**, if $E = \binom{V}{2}$ and a graph $G' = (V', E')$ is called a **subgraph** of $G = (V, E)$, if $V' \subseteq V$ and $E' \subseteq E$. Furthermore, we call a subgraph $G' = (V', E')$ of $G = (V, E)$ a **spanning subgraph** if $E' \subseteq E$ and $V' = V$. For every vertex $v \in V$ of a graph $G = (V, E)$ we call $\deg_G(v) := |\{w \in V : (v, w) \in E\}|$ the **degree** of v and G is called t-**regular** if we have $\deg_G(v) = t$ for every $v \in V$. Let $S \subseteq V$ be a subset of vertices of a graph $G = (V, E)$ then S is a **connected component** of G if for each pair of vertices $v, v' \in V$ there exist vertices $v_1, \ldots, v_k \in S$ such that $(v, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k), (v_k, v') \in E$ and for each pair of vertices $v \in S$ and $w \in V \setminus S$ we have $(v, w) \notin E$. A graph is called **connected** if it has only one connected compotent.
A **hypergraph** is a pair $H = (V, E)$, where the edge set $E \subseteq 2^V$ can be any set of subsets of V. Note, that every simplicial complex is a hypergraph, but not every hypergraph is a simplicial complex, since subsets of an edge do not have to be an edge in a hypergraph. If all edges of a hypergraph have the same cardinality k, then we call it a k-**uniform hypergraph**. Analogously to the terminology of graphs, a hypergraph $H' = (V', E')$ is called a **subhypergraph** of the hypergraph $H = (V, E)$, if $V' \subseteq V$ and $E' \subseteq E$.

Chapter 1

Introduction

When considering a graph usually we can easily see if it is connected or not, but if it is connected what can we say about how stable or strong this connectedness is. As an example consider the following graph which might represent a computer network or something similar in applications:

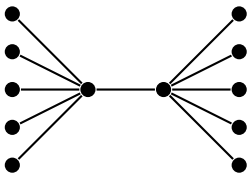

FIGURE 1.1: A weakly connected graph

We see that if we removed the middle edge the graph would split into two relatively large connected components, so intuitively we could say that its connectedness is not very strong. It would be helpful if we could measure the connectivity of a graph so that we can compare several graphs according to this value.
The classical Cheeger constant of a graph is a well-studied object and can be imagined as such a measure. Intuitively it is constructed as follows: From a connected graph we can delete edges to make it become disconnected and so there exists a smallest (in terms of numbers of vertices) connected component. Now, the Cheeger constant is the smallest quotient that can appear by deviding the number of removed edges by the size of the smallest of the resulting connected components. Formally the Cheeger constant of a graph $G = (V, E)$ is defined by:

$$h(G) := \min \left\{ \frac{|\delta(A)|}{|A|} A \subset V, 1 \leq |A| \leq \frac{|V|}{2} \right\},$$

with $\delta(A) := \{e = (v, w) \in Ev \in A, w \in V \setminus A\}$.

There is a lot of literature which studies this Cheeger constant for arbitrary graphs, whereas it is pretty easy to determine for the complete graph on n-vertices K_n, where we have:

$$h(K_n) = \left\lceil \frac{n}{2} \right\rceil$$

This can be easily verified as follows: For any subset $A \subset [n]$ we have

$$\frac{|\delta(A)|}{|A|} = \frac{|A|(n-|A|)}{|A|} = n - |A|,$$

and by the preceding definition we have $|A| \leq \frac{n}{2}$ so we immediately get:

$$h(K_n) = n - \left\lfloor \frac{n}{2} \right\rfloor = \left\lceil \frac{n}{2} \right\rceil$$

A disadvantage of this Cheeger constant is that it is only defined for graphs which can only represent relations among two vertices. To measure the connectivity of constructions which can represent relations among arbitrary numbers of vertices (namely simplicial complexes) we need a more general notion of the Cheeger constant which we will study in this thesis. It was first introduced by Lineal and Meshulam (see [2]) and later independently by Gromov (see [3]) and is defined as follows:

Let X be a simplicial complex. For a cochain $\varphi \in C^k(X)$ we define the **norm** of φ as $\|\varphi\| := |\mathrm{supp}(\varphi)|$. Let now $\varphi \in C^k(X)$, such that $\|\delta^{k-1}(\phi) + \varphi\| \geq \|\varphi\|$ holds for every $\phi \in C^{k-1}(X)$, then we call φ a k-**cosystole**. For general cochains $\varphi \in C^k(X)$ we define the **cosystolic norm** of φ by:

$$\|\varphi\|_{csy} := \min\left\{ \|\delta^{k-1}(\phi) + \varphi \| \phi \in C^{k-1}(X) \right\}$$

Furthermore, any $c \in \varphi + \mathrm{Im}(\delta^{k-1})$ satisfying $\|c\| = \|\varphi\|_{csy}$ is called a **cosystolic form** of φ.

The quotient

$$\|\varphi\|_{exp} := \frac{\|\delta^k(\varphi)\|}{\|\varphi\|_{csy}}$$

is called the **coboundary expansion** of φ and

$$h_k(X) := \min_{\substack{\varphi \in C^k(X) \\ \varphi \notin \mathrm{Im}(\delta^{k-1})}} \|\varphi\|_{exp}$$

is called the k-th **Cheeger constant** of X.

A cosystole $\varphi \in C^k(X)$ is called a k-**Cheeger cosystole** if $h_k(X) = \|\varphi\|_{exp}$.

Let us study a simple example which might help to understand the preceding definitions. Consider the complete 2-simplex $\Delta^{[3]}$ and let $\varphi := (\{1,2\})^*$ be the cochain whose support only consists of one 1-simplex. Obviously, φ is a cosystole since there exists no cochain $\phi \in C^0(\Delta^{[3]})$ such that $\delta^0(\phi) = \varphi$. Furthermore, we have

$\|\delta^1(\varphi)\| = 1$ so we immediately see that φ is a Cheeger cosystole since we get

$$\|\varphi\|_{exp} = \frac{\|\delta^1(\varphi)\|}{\|\varphi\|} = 1,$$

which coincides with the Cheeger constant in this case (see Equation 1.1).

We could even define the Cheeger constants more generally for polyhedral complexes (see [6]), but in this thesis we will only focus simplicial complexes.
Note, that the classical Cheeger constant of a graph coincides with the 0-th Cheeger constant $h_0(X)$ if we identify the cosystolic norm of a 0-cochain $\varphi \in C^0(X)$ with $\min\{|\text{supp}(\varphi), n - |\text{supp}(\varphi)|\}$, where n is the number of vertices in X. For larger k's the value of $h_k(X)$ is not even known for all standard simplices $X = \Delta^{[n]}$. By now we only have the estimate

$$\frac{n}{k+2} \leq h_k(\Delta^{[n]}) \leq \left\lceil \frac{n}{k+2} \right\rceil \tag{1.1}$$

which was proven by Wallach and Meshulam (see [4], Proposition 2.1), so we have the exact value $h_k(\Delta^{[n]}) = \frac{n}{k+2}$ when $k+2$ divides n. In [1] (Proposition 6.5) Kozlov showed that the upper bound is archieved when $k = n-3$, so we have $h_{n-3}(\Delta^{[n]}) = 2$ and furthermore he showed that $h_1(\Delta^{[n]}) = \frac{n}{3}$ even holds for every n which is not a power of 2.

The classical 0-th Cheeger constant of a graph is still pretty easy to understand intuitively, whereas the higher-dimensional generalizations raise the question what a measure of connectivity could mean in those cases. The following observation might help the reader to develop this intuition. For a graph $G = (V, E)$ the classical Cheeger constant $h(G)$ equals 0 if and only if G is disconnected, since for any non-empty proper subset of vertices $A \subset V$, the set $\delta(A)$ is empty, if and only if there is no edge between A and $V \setminus A$. More generally, for a simplicial complex X, the k-th Cheeger constant $h_k(X)$ equals zero, if and only if the k-th homology group $H_k(X)$ of X is not trivial, as follows:
For any cochain $\varphi \in C^k(X)$ we have $\|\varphi\|_{csy} > 0$ if and only if $\varphi \notin \text{Im}(\delta^{k-1})$ and $\delta^k(\varphi) = 0$ if and only if $\varphi \in \ker(\delta^k)$, but the existence of a cochain satisfying these two properties is equivalent to $\text{Im}(\delta^{k-1}) \subsetneq \ker(\delta^k)$, which just means that $H^k(X) \neq \{0\}$.
So, we have to study the Cheeger constants of those simplicial complexes whose homology vanishes and the most obvious example of those complexes is the standard simplex $\Delta^{[n]}$.

In the first part of this thesis we will develop some theory about the cosystolic norm of cochains, since a better understanding of cosystoles seems to be the key knowledge to determine the Cheeger constants. In the second part we will focus on the

special case of 1-cosystoles and the first Cheeger constant, where we have an interesting graph theoretical approach introduced by Kozlov in [1], that seems to be suited well to investigate those cosystoles in a purely combinatorial way. The last chapter adresses an interesting observation about partitioning consecutive numbers, which is not directly related to the topic of Cheeger constants, but arose during our research and might be helpful in other branches of combinatorics. At the end in the appendices we give practical algorithms including the calculation of cosystolic norms, Cheeger constants and partitionings according to the content of the last chapter.

Chapter 2

On general cosystoles and Cheeger constants

When we want to determine whether a cochain is a cosystole or not or to determine the cosystolic norm of a cochain by now we only have the original definition of cosystolicity, which does not seem to be very handy. In this chapter we want to develop tools to get hands on this problem, especially by estimating the cosystolic norm in various situations and investigating the structure, how cosystoles in certain simplicial complexes are arranged. Furthermore, at the end of this chaper we will develop some interesting statements about the combinatorial structure of Cheeger cosystoles.

2.1 Piercing numbers and the cycle detection theorem

The following definition is adopted from [6] (Definition 2.1).

Definition 2.1.1. *Let V be some set and $\mathcal{F} \subseteq 2^V$ a family of finite subsets of V. A subset $P \subseteq V$ is called a **piercing set** of $\mathcal{F}$ if we have $P \cap F \neq \emptyset$ for all $F \in \mathcal{F}$. The minimal cardinality which a piercing set of $\mathcal{F}$ can attain is called the **piercing number** of $\mathcal{F}$, denoted by $\tau(\mathcal{F})$. For $v \in V$ and $F \in \mathcal{F}$ we say that F **is pierced by** v, if $v \in F$.*

Example 2.1.1. *Let $V := \{1,2,3,4,5\}$ and $\mathcal{F} := \{\{1,2\}\{2,3,4\},\{1,5\},\{2,4,5\}\}$, then we have $\tau(\mathcal{F}) = 2$, since for example $P := \{2,5\}$ is a minimal piercing set of $\mathcal{F}$.*

Later we will talk about piercing numbers and piercing sets of families of k-chains, which we define as follows:

Definition 2.1.2. *Let X be a simplicial complex and $\mathcal{F} \subseteq C_k(X)$ a family of k-chains. The **piercing sets** and the **piercing number** of $\mathcal{F}$ are defined as the piercing sets and the piercing number of the family $\{\mathrm{supp}(F) : F \in \mathcal{F}\}$.*

In [6] (Theorem 2.2) Kozlov stated the following useful method to bound the cosystolic norm of a cochain.

Theorem 2.1.1 (The cycle detection theorem)**.** *Let X be a simplicial complex, $k \geq 1$, and $\varphi \in C^k(X)$. Let now $\mathcal{F} = \{\alpha_1, \ldots, \alpha_t\}$ be a family of k-cycles in $C_k(X)$, such that*

$\langle \varphi, \alpha_i \rangle = 1$ *for all* $1 \leq i \leq t$*, then we have:*

$$\|\varphi\|_{csy} \geq \tau(\mathcal{F})$$

Proof. Let $\psi \in C^{k-1}(X)$, then for any $1 \leq i \leq t$ we have:

$$\begin{aligned}\langle \varphi + \delta^{k-1}(\psi), \alpha_i \rangle &= \langle \varphi, \alpha_i \rangle + \langle \delta^{k-1}(\psi), \alpha_i \rangle \\ &= \langle \varphi, \alpha_i \rangle + \langle \psi, \partial_{k-1}(\alpha_i) \rangle \\ &= \langle \varphi, \alpha_i \rangle + \langle \psi, 0 \rangle \\ &= \langle \varphi, \alpha_i \rangle = 1\end{aligned}$$

This means that we have $\operatorname{supp}(\varphi + \delta^{k-1}(\psi)) \cap \operatorname{supp}(\alpha_i) \neq \emptyset$ for all $1 \leq i \leq t$, so $\operatorname{supp}(\varphi + \delta^{k-1}(\psi))$ is a piercing set of $\mathcal{F}$ and we get:

$$\|\varphi + \delta^{k-1}(\psi)\| = |\operatorname{supp}(\varphi + \delta^{k-1}(\psi))| \geq \tau(\mathcal{F})$$

Since ψ was chosen arbitrarily we are done. □

The following corollary (a special case of the preceding theorem) was also stated by Kozlov in [6] (Corollary 2.3).

Corollary 2.1.1. *Let X be a simplicial complex and $\varphi \in C^k(X)$.*
Let now $\mathcal{F} = \{\alpha_1, \ldots, \alpha_{\|\varphi\|}\}$ be a family of k-cycles in $C_k(X)$, such that $\langle \varphi, \alpha_i \rangle = 1$ for all $1 \leq i \leq \|\varphi\|$ and $\operatorname{supp}(\alpha_i) \cap \operatorname{supp}(\alpha_j) = \emptyset$ for all $i \neq j$, then φ is a cosystole.

Proof. Since the supports of the cycles $\alpha_1, \ldots, \alpha_{\|\varphi\|}$ are pairwise disjoint, we obviously have $\tau(\mathcal{F}) = \|\varphi\|$ and using the cycle detection theorem we are done. □

Example 2.1.2. *Consider the cochain*

$$\varphi = (\{1,2,4\} + \{2,4,6\} + \{2,5,6\} + \{3,4,6\})^* \in C^2(\Delta^{[6]})$$

(Figure 2.1 illustrates how the support of this cochain can be imagined) and the family of cycles

$$\begin{aligned}\mathcal{F} = \{&\{1,2,3\} + \{1,2,4\} + \{1,4,3\} + \{2,4,3\}, \\ &\{1,2,5\} + \{1,2,6\} + \{1,5,6\} + \{2,5,6\}, \\ &\{3,4,5\} + \{3,4,6\} + \{3,5,6\} + \{4,5,6\}, \\ &\{1,3,5\} + \{1,3,6\} + \{1,4,5\} + \{1,4,6\} + \\ &\{2,3,5\} + \{2,3,6\} + \{2,4,5\} + \{2,4,6\}\} \subset C_2(\Delta^{[6]})\end{aligned}$$

It is easy to check that we have $\langle \varphi, \alpha \rangle = 1$ for all $\alpha \in \mathcal{F}$ and $\tau(\mathcal{F}) = 4$, so we get $\|\varphi\|_{csy} \geq 4$ by the cycle detection theorem. We can immediately see, that φ is a cosystole, either by the fact, that we always have $\|\varphi\|_{csy} \leq \|\varphi\|$, so in our case we have $\|\varphi\| = 4 = \|\varphi\|_{csy}$, or by using the preceding corollary, since the supports of the cycles in $\mathcal{F}$ are pairwise disjoint.

Note, that φ is even a Cheeger cosystole, since we have $\|\varphi\|_{exp} = \frac{6}{4}$, so we know that for the case $n = 6$ and $k = 2$ the lower bound of the estimate 1.1 is archieved, even though $k+2$ does not divide n.

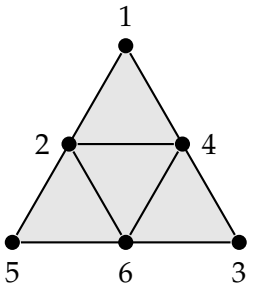

FIGURE 2.1: The support of a 2-Cheeger cosystole (The gray triangles represent the four 2-simplices)

To get the best possible results using the cycle detection theorem, the challenge is now for a certain cochain φ to find families of cycles, such that they have a large piercing number and φ evaluates to 1 on every cycle. The following construction seems to be suited well to get hands on this problem.
For a simplicial complex X and a cochain $\varphi \in C^k(X)$ we define the following set of cycles:

$$\mathcal{T}_\varphi := \left\{ \partial_k(\sigma) : \sigma \in \operatorname{supp}(\delta^k(\varphi)) \right\}$$

Proposition 2.1.1. *Let X be a simplicial complex and $\varphi \in C^k(X)$, then we have:*

$$\|\varphi\|_{csy} \geq \tau(\mathcal{T}_\varphi)$$

Proof. By the definition of the coboundary map, we obviously have
$\langle \varphi, \partial_k(\sigma) \rangle = 1$ for all $\partial_k(\sigma) \in \mathcal{T}_\varphi$ and so by the cycle detection theorem we are done. □

Note, that in many cases even the equality $\|\varphi\|_{csy} = \tau(\mathcal{T}_\varphi)$ can be reached, but unfortunately this is not always true as the following example shows.

Example 2.1.3. *Let $\varphi \in C^2(\Delta^{[6]})$ be the cosystole from Example 2.1.2. We have:*

$$\begin{aligned}\mathcal{T}_\varphi = \{&\{1,2,3\} + \{1,2,4\} + \{1,3,4\} + \{2,3,4\},\\ &\{1,2,4\} + \{1,2,5\} + \{1,4,5\} + \{2,4,5\},\\ &\{2,3,5\} + \{2,3,6\} + \{2,5,6\} + \{3,5,6\},\\ &\{1,2,5\} + \{1,2,6\} + \{1,5,6\} + \{2,5,6\},\\ &\{3,4,5\} + \{3,4,6\} + \{3,5,6\} + \{4,5,6\},\\ &\{1,3,4\} + \{1,3,6\} + \{1,4,6\} + \{3,4,6\}\}\end{aligned}$$

It is easy to check that we get $\tau(\mathcal{T}_\varphi) = 3$, but as shown in Example 2.1.1 we have $\|\varphi\|_{csy} = 4$.

In particular we see that we can just remove the simplex $\{2,4,6\}$ from the support of φ is the preceding example and the remaining simplices still form a piercing set of $\mathcal{T}_\varphi$.
Nevertheless, we conjecture that for the 1-dimensional case the equality holds, since removing a simplex from the support of a 1-cosystole φ can never result in a piercing set of $\mathcal{T}_\varphi$ as the following lemma shows.

Lemma 2.1.1. *Let $\varphi \in C^1(\Delta^{[n]})$ be a cosystole, $\sigma \in \operatorname{supp}(\varphi)$ and $P \coloneqq \operatorname{supp}(\varphi) \setminus \{\sigma\}$. Then P is not a piercing set of $\mathcal{T}_\varphi$.*

Proof. Assume P is a piercing set of $\mathcal{T}_\varphi$, and denote $\sigma = \{v_1, v_2\}$, then for all $v \in [n]$ there exists a $w \in \sigma$ such that $\{v, w\} \in \operatorname{supp}(\varphi)$, since otherwise we would have $\partial_1((\sigma, v)) \in \mathcal{T}_\varphi$, but $\operatorname{supp}(\partial_1((\sigma, v))) \cap P = \emptyset$, so P would not be a piercing set of $\mathcal{T}_\varphi$. It follows that there exists a $w \in \sigma$, such that

$$|\operatorname{supp}(\delta^0(w^*)) \cap \operatorname{supp}(\varphi)| \geq \left\lceil \frac{n-2}{2} \right\rceil + 1 = \left\lceil \frac{n-2}{2} + 1 \right\rceil = \left\lceil \frac{n}{2} \right\rceil > \left\lfloor \frac{n-1}{2} \right\rfloor,$$

so φ is not cosystolic, since for a 1-cosystole φ by definition we always have $|\operatorname{supp}(\delta^0(w^*)) \cap \operatorname{supp}(\varphi)| \leq \left\lfloor \frac{n-1}{2} \right\rfloor$ for every vertex $w \in [n]$ and we are done. □

It seems to be very difficult to determine the piercing number of $\mathcal{T}_\varphi$ explicitely, but the concept of piercing complexes might be useful on the way to solve this problem.

2.2 Piercing complexes

Let V be some set, $\mathcal{F} \subset 2^V$ a family of subsets and P a piercing set of $\mathcal{F}$. Then for any $v \in V$ obviously $P \cup \{v\}$ is also a piercing set of $\mathcal{F}$. We can use this fact to construct a simplicial complex, which contains all information about the piercing sets for a given family of sets as follows:

Definition 2.2.1. *Let V be a set and $\mathcal{F} \subset 2^V$ a family of subsets. Then the **piercing complex** of $\mathcal{F}$ is defined as:*

$$\Delta_{\mathcal{F}} \coloneqq \left\{ V' \subseteq V : (V \setminus V') \cap F \neq \emptyset, \text{ for all } F \in \mathcal{F} \right\}$$

So, $\Delta_{\mathcal{F}}$ consists of all subsets of V, such that their complements in V are piercing sets of $\mathcal{F}$ and indeed, $\Delta_{\mathcal{F}}$ defines a simplicial complex, since deleting an element from the complement of a piercing set is equivalent to adding an element to a piercing set, which preserves the condition of being a piercing set.
Note, that if $\mathcal{F}$ is a family of k-chains, the piercing complex $\Delta_{\mathcal{F}}$ is just defined as the piercing complex of the family $\{\operatorname{supp}(F) : F \in \mathcal{F}\}$.

Example 2.2.1. *Let V be an arbitrary set and $\mathcal{F} \coloneqq 2^V$ its power set. Then the piercing complex $\Delta_{\mathcal{F}}$ is empty, since even the complement of a single vertex $v \in V$ is not a piercing*

set of $\mathcal{F}$. More generally for $\mathcal{F} \subseteq 2^V$ we have that $\Delta_{\mathcal{F}}$ is empty if and only if $\{v\} \in \mathcal{F}$, for all $v \in V$.
On the other hand $\Delta_{\mathcal{F}}$ is a complete simplex on $|V|$ vertices if and only if $\mathcal{F}$ is empty, since only in this case even the empty set is a piercing set of $\mathcal{F}$.

We can now reformulate the question of determining the piercing number $\tau(\mathcal{F})$ by asking for the dimension of $\Delta_{\mathcal{F}}$, since we have the equality:

$$\tau(\mathcal{F}) = |V| - \dim(\Delta_{\mathcal{F}}) - 1$$

Since our main interest in this section will be to investigate the piercing complex of $\mathcal{T}_\varphi$ for a given cochain $\varphi \in C^k(X)$ (where X is some simplicial complex) we will use a shorter notation for this piercing complex and set $\Delta_\varphi := \Delta_{\mathcal{T}_\varphi}$. Then the preceding formula turns to:

$$\tau(\mathcal{T}_\varphi) = |X^{(k)}| - \dim(\Delta_\varphi) - 1$$

Theorem 2.2.1. *Let X be a simplicial complex and $\varphi \in C^k(X)$, then we have:*

$$\tilde{H}_i(\Delta_\varphi) \cong \{0\} \quad \textit{for all } i \leq k-1$$

Proof. Let $\varphi \in C^k(X)$ be chosen arbitrarily. Then for all $\sigma \in \operatorname{supp}(\delta^k(\varphi))$ we have $|\operatorname{supp}(\partial_k(\sigma))| = k+2$. Therefore, for all $S \subset X^{(k)}$ such that $|S| \leq k+1$ we have that $X^{(k)} \setminus S$ is a piercing set of $\mathcal{T}_\varphi$. This just means, that Δ_φ has a full k-skeleton, so we immediately get $\tilde{H}_i(\Delta_\varphi) \cong 0$ for all $i \leq k-1$. □

Definition 2.2.2. *Let X be a simplicial complex on the vertex set V. Then the simplicial complex*

$$X^\vee := \{\sigma \subseteq V : V \setminus \sigma \notin X\}$$

*is called the **Alexander dual** of X.*

The following theorem can be found in [8] (Theorem 1.1).

Theorem 2.2.2 (The Alexander duality theorem)**.** *Let X be a simplicial complex on n vertices and $X^\vee$ its Alexander dual. Then we have:*

$$\tilde{H}_i(X) \cong \tilde{H}^{n-i-3}(X)$$

Definition 2.2.3. *Let V be some set and $\mathcal{F} \subseteq 2^V$ a family of subsets of V. Then the simplicial complex*

$$\Delta[\mathcal{F}] := \{\sigma \subseteq V : \textit{there exists an } F \in \mathcal{F} \textit{ such that } \sigma \subseteq F\}$$

*is called the **induced complex** of $\mathcal{F}$.*

The following statement was developed in [9].

Proposition 2.2.1. *Let $\mathcal{F} \subseteq 2^V$ be a family of subsets of a set V and $\bar{\mathcal{F}} := \{V \setminus F : F \in \mathcal{F}\}$. Then we have:*

$$\Delta\left[\bar{\mathcal{F}}\right]^\vee = \Delta_{\mathcal{F}}$$

Proof. We have:

$$\begin{aligned}
& \sigma \in \Delta\left[\bar{\mathcal{F}}\right]^\vee \\
\iff\ & V \setminus \sigma \notin \Delta\left[\bar{\mathcal{F}}\right] \\
\iff\ & \text{There exists no } F \in \bar{\mathcal{F}} \text{ such that } V \setminus \sigma \subseteq F \\
\iff\ & \text{There exists no } F \in \bar{\mathcal{F}} \text{ such that } (V \setminus \sigma) \cap (V \setminus F) = \emptyset \\
\iff\ & \text{There exists no } F' \in \mathcal{F} \text{ such that } (V \setminus \sigma) \cap F' = \emptyset \\
\iff\ & V \setminus \sigma \text{ is a piercing set of } \mathcal{F} \\
\iff\ & \sigma \in \Delta_{\mathcal{F}}
\end{aligned}$$

□

Theorem 2.2.3. *Let X be a finite simplicial complex and $\varphi \in C^k(X)$, then we have:*

$$\tilde{H}_k(\Delta_\varphi) \cong \{0\}$$

Proof. For all $\partial_k(\sigma) \in \mathcal{T}_\varphi$ we have $|\mathrm{supp}(\partial_k(\sigma))| = k+2$, so we get:

$$\dim(\Delta[\bar{\mathcal{T}}_\varphi]) = |X^{(k)}| - (k+2) - 1 = |X^{(k)}| - k - 3$$

Now, there exist no two simplices of dimension $|X^{(k)}| - k - 3$ in $\Delta[\bar{\mathcal{T}}_\varphi]$, such that they have a face in common, since generally in a simplicial complex there can not exist two distinct simplices which differ by only one face. So we have:

$$\tilde{H}_{|X^{(k)}|-k-3}(\Delta[\bar{\mathcal{T}}_\varphi]) \cong \{0\}$$

By the Alexander duality theorem and Proposition 2.2.1 we get

$$\tilde{H}_k(\Delta_\varphi) \cong \tilde{H}^k(\Delta_\varphi) = \tilde{H}^{|X^{(k)}|-(|X^{(k)}|-k-3)-3}(\Delta_\varphi) \cong 0,$$

where the first isomorphy is true, because we consider homology / cohomology over a field and Δ_φ is finite. □

2.3 Large cosystoles of a simplex

In this section we focus our investigations onto the question, what is the largest norm a cosystole can attain when the underlying simplicial complex is a complete simplex.
Let us first introduce a simplicial complex which represents all the information about cosystoles we need.

Definition 2.3.1. *Let X be a simplicial complex and $1 \leq k \leq \dim(X)$. Then the simplicial complex*

$$\mathcal{C}_k(X) := \left\{ S \subseteq X^{(k)} : \left(\sum_{\sigma \in S} \sigma \right)^* \text{ is a cosystole} \right\}$$

*is called the **k-cosystolic complex** of X.*

Obviously, $\mathcal{C}_k(X)$ defines a simplicial complex since removing a simplex from the support of a cosystole and considering the corresponding cochain again preserves cosystolicity.
Now, for any simplicial complex X and any $1 \leq k \leq \dim(X)$, we define the following number (the largest norm, a cosystole can attain):

$$C_{max}(X,k) := \max \left\{ \|\varphi\|_{csy} : \varphi \in C^k(X) \right\}$$

Note, that we have the relation $\dim(\mathcal{C}_k(X)) = C_{max}(X,k) - 1$.

For the sake of completeness we still define the following number of the smallest norm a non-cosystolic cochain can attain:

$$C_{min}(X,k) := \min \left\{ \|\varphi\| : \varphi \in C^k(X), \|\varphi\| > \|\varphi\|_{csy} \right\}$$

Proposition 2.3.1. $C_{min}(\Delta^{[n]},k) \leq \left\lceil \frac{n-k+1}{2} \right\rceil$

Proof. Let $\sigma = [k]$ and

$$\varphi = \left(\sum_{i=k+1}^{\lceil \frac{n-k+1}{2} \rceil + k} (\sigma, i) \right)^* \in C^k(\Delta^{[n]}),$$

then we have $\|\varphi\| = \left\lceil \frac{n-k+1}{2} \right\rceil$, but $\|\varphi\|_{csy} \leq \left\lfloor \frac{n-k+1}{2} \right\rfloor - 1$, since
$\|\delta^{k-1}(\sigma^*) + \varphi\| = \left\lfloor \frac{n-k+1}{2} \right\rfloor - 1$, so we are done. □

Let us now return to the largest possible cosystolic norm and introduce the following family of cycles:

$$\mathcal{T}_n^k := \left\{ \partial_k(\sigma) : \sigma \in \binom{[n]}{k+2} \right\}$$

Remark 2.3.1. *If k is odd, then $\mathcal{T}_n^k$ coincides with $\mathcal{T}_\varphi$ for the cochain $\varphi = \left(\sum_{\sigma \in \binom{[n]}{k+1}} \sigma \right)^*$.*

We can use that family of cycles now to determine a lower bound for the number $C_{max}(\Delta^{[n]},k)$ if k is odd.

Proposition 2.3.2. *Let k be odd, then we have:*

$$C_{max}(\Delta^{[n]}, k) \geq \left\lceil \frac{\binom{n}{k+2}}{n-k-1} \right\rceil$$

Proof. Since any simplex $\sigma \in \binom{[n]}{k+1}$ intersects the support of exactly $n-k-1$ cycles from $\mathcal{T}_n^k$, any piercing set of $\mathcal{T}_n^k$ must contain at least $\left\lceil \frac{\binom{n}{k+2}}{n-k-1} \right\rceil$ elements and by Proposition 2.1.1 and Remark 2.3.1 we are done. □

We can also give a recursive formula providing a lower bound for $C_{max}(\Delta^{[n]}, k)$.

Proposition 2.3.3. $C_{max}(\Delta^{[n]}, k) \leq C_{max}(\Delta^{[n-1]}, k) + C_{max}(\Delta^{[n-1]}, k-1)$

Proof. Let $\varphi \in C^k(\Delta^{[n]})$ be an arbitrary cochain. Furthermore, let $\phi_1 \in C^k(\Delta^{[n-1]})$ and $\phi_2 \in C^k(\Delta^{[n]})$, such that $\operatorname{supp}(\phi_1) = \{\sigma \in \operatorname{supp}(\varphi) : n \notin \sigma\}$ and $\operatorname{supp}(\phi_2) = \{\sigma \in \operatorname{supp}(\varphi) : n \in \sigma\}$. Without loss of generality let ϕ_1 be a cosystole. Otherwise, just add a coboundary to φ, such that ϕ_1 becomes cosystolic. Now, we can identify ϕ_2 with a cochain $\phi_2' \in C^{k-1}(\Delta^{[n-1]})$ by removing the vertex n from every simplex in its support. Adding a coboundary to ϕ_2' is equivalent to adding the coboundary of a cochain $\psi \in C^{k-1}(\Delta^{[n]})$ to ϕ_2, where every simplex in the support of ψ contains the vertex n. Fortunately, the support of such a coboundary is always disjoint to the support of ϕ_1, so we have

$$\|\varphi\|_{csy} \leq \|\phi_1\|_{csy} + \|\phi_2'\|_{csy} \leq C_{max}(\Delta^{[n-1]}, k) + C_{max}(\Delta^{[n-1]}, k-1),$$

and we are done. □

For the 1-dimensional case we can find better estimates, since we are able to determine the piercing number of $\mathcal{T}_n^1$ exactly. A more elementary proof of the following estimate is given as Proposition 3.2.1 in the following chapter, where we will even see, that equality can be reached, shown in Theorem 3.2.1.

Proposition 2.3.4. $C_{max}(\Delta^{[n]}, 1) \geq \binom{n}{2} - \left\lfloor \frac{n^2}{4} \right\rfloor$

Proof. Asking for the smallest piercing set of $\mathcal{T}_n^1$ is equivalent to asking for the largest triangle-free graph (i.e. a graph on n vertices, containing as many edges as posssible, but no complete graph on 3 vertices as a subgraph) and considering the complement. Mantel's theorem (see [7]) says, that a triangle-free graph on n vertices has at most $\left\lfloor \frac{n^2}{4} \right\rfloor$ edges, so we immediately get

$$\tau(\mathcal{T}_n^1) = \binom{n}{2} - \left\lfloor \frac{n^2}{4} \right\rfloor$$

and by Proposition 2.1.1 and Remark 2.3.1 we are done. □

Unfortunately, determining the piercing number of $\mathcal{T}_n^k$ for $k \geq 2$, or equivalently, determining the largest $(k+1)$-uniform hypergraph on n-vertices, containing no complete $(k+1)$-uniform hypergraph on $k+2$ vertices as a subhypergraph, seems to be very difficult (see [7]), so we can not use the preceding procedure to say something about $C_{max}(\Delta^{[n]}, k)$ for larger k's in general.
Nevertheless, we can exactly determine this number for the ultimate and the penultimate proper dimension as follows.

Theorem 2.3.1. *$C_{max}(\Delta^{[n]}, n-2) = 1$, for all $n \geq 3$*

Proof. Let $\sigma \in \binom{[n]}{n-1}$ be chosen arbitrarily, $\varphi := \sigma^* \in C^{n-2}(\Delta^{[n]})$ and
$\mathcal{F} := \{\alpha\}$, where α is the boundary of the single $(n-1)$-dimensional simplex in $\Delta^{[n]}$.
Obviously, we have $\langle \varphi, \alpha \rangle = 1$ and $\tau(\mathcal{F}) = 1$, so by the cycle detection theorem we have $\|\varphi\|_{csy} \geq 1$.
Now, let $\sigma_1, \sigma_2 \in \binom{[n]}{n-1}$ be chosen arbitrarily again ($\sigma_1 \neq \sigma_2$) and $c := \sigma_1 \cap \sigma_2 \in \binom{[n]}{n-2}$.
Then we have $\delta^{n-3}(c^*) + \sigma_1^* + \sigma_2^* = 0$, so there exists no $(n-2)$-cosystole attaining norm 2. □

Lemma 2.3.1. *Let $n \geq 4$, then we have:*

$$\tau(\mathcal{T}_n^{n-3}) = \left\lceil \frac{n}{2} \right\rceil$$

Proof. For each $\sigma \in \binom{[n]}{n-2}$ there exist exactly two cycles $\alpha_1, \alpha_2 \in \mathcal{T}_n^{n-3}$
($\alpha_1 \neq \alpha_2$), such that $\sigma \in \operatorname{supp}(\alpha_1) \cap \operatorname{supp}(\alpha_2)$, so the largest possible number of cycles from $\mathcal{T}_n^{n-3}$ that can be pierced by one simplex is two. Furthermore, we have $\left|\mathcal{T}_n^{n-3}\right| = \binom{n}{n-1} = n$, so we get $\tau(\mathcal{T}_n^{n-3}) \geq \lceil \frac{n}{2} \rceil$.
On the other hand, for all $\alpha_1, \alpha_2 \in \mathcal{T}_n^{n-3}$ there exists a $\sigma \in \binom{[n]}{n-2}$, such that
$\sigma \in \operatorname{supp}(\alpha_1) \cap \operatorname{supp}(\alpha_2)$, so we get $\tau(\mathcal{T}_n^{n-3}) \leq \lceil \frac{n}{2} \rceil$. □

Lemma 2.3.2. *Let $S \subset \binom{[n]}{n-2}$, such that $|S| \geq \lfloor \frac{n}{2} \rfloor + 1$, then there exist $\sigma, \sigma' \in S$ ($\sigma \neq \sigma'$), such that $|\sigma \cap \sigma'| = n-3$.*

Proof. For $\sigma, \sigma' \in \binom{[n]}{n-2}$ the condition $|\sigma \cap \sigma'| < n-3$ is equivalent to the condition $([n] \setminus \sigma) \cap ([n] \setminus \sigma') = \emptyset$. Since we obviously have $|[n] \setminus \sigma| = 2$ for all $\sigma \in \binom{[n]}{n-2}$ we can find at most $\lfloor \frac{n}{2} \rfloor$ simplices $\sigma_1, \ldots, \sigma_{\lfloor \frac{n}{2} \rfloor} \in \binom{[n]}{n-2}$, such that the sets $[n] \setminus \sigma_1, \ldots, [n] \setminus \sigma_{\lfloor \frac{n}{2} \rfloor}$ are pairwise disjoint and we are done. □

Theorem 2.3.2. *$C_{max}(\Delta^{[n]}, n-3) = \lfloor \frac{n}{2} \rfloor$, for all $n \geq 4$*

Proof. Let $S \subset \binom{[n]}{n-2}$ be a minimal piercing set of $\mathcal{T}_n^{n-3}$ as constructed in the proof of Lemma 2.3.1 and $\varphi := S^* \in C^{n-3}(\Delta^{[n]})$.
If n is even, then $n-3$ is odd, so apply Proposition 2.1.1, Remark 2.3.1 and Lemma 2.3.1 and we get:

$$C_{max}(\Delta^{[n]}, n-3) \geq \tau(\mathcal{T}_n^{n-3}) = \frac{n}{2}$$

If n is odd then by the construction of φ there exists exactly one $\alpha' \in \mathcal{T}_n^{n-3}$, such that $\langle \varphi, \alpha' \rangle = 0$, since $|\mathrm{supp}(\alpha') \cap \mathrm{supp}(\varphi)| = 2$. Set $\mathcal{F} := \mathcal{T}_n^{n-3} \setminus \{\alpha'\}$, then we have $\langle \varphi, \alpha \rangle = 1$ for all $\alpha \in \mathcal{F}$ and $\tau(\mathcal{F}) \geq \tau(\mathcal{T}_n^{n-3}) - 1 = \lceil \frac{n}{2} \rceil - 1 = \lfloor \frac{n}{2} \rfloor$ by Lemma 2.3.1, so by the cycle detection theorem we get $C_{max}(\Delta^{[n]}, n-3) \geq \lfloor \frac{n}{2} \rfloor$. One the other hand let $\varphi \in C^{n-3}(\Delta^{[n]})$, such that $\|\varphi\| = \lfloor \frac{n}{2} \rfloor + 1$, then by Lemma 2.3.2 there exist $\sigma_1, \sigma_2 \in \mathrm{supp}(\varphi)$, such that $|\sigma_1 \cap \sigma_2| = n-3$. Now set $\psi := (\sigma_1 \cap \sigma_2)^* \in C^{n-4}(\Delta^{[n]})$, then we have $\|\delta^{n-4}(\psi)\| = 3$ and $\left|\mathrm{supp}(\delta^{n-4}(\psi)) \cap \mathrm{supp}(\varphi)\right| \geq 2$. Thus, we have $\|\delta^{n-4}(\psi) + \varphi\| \leq \|\varphi\| - 1$ and φ can not be a cosystole, so we get $C_{max}(\Delta^{[n]}, n-3) \leq \lfloor \frac{n}{2} \rfloor$ and we are done. □

2.4 Embeddings of cosystoles

When searching for cosystoles another approach is to consider cochains, which are already proven to be cosystolic and try to, roughly speaking, change or extend them to create new cosystoles. The most obvious way to do this is just to add a vertex to the underlying simplex by the embedding:

$$\begin{aligned} i_n^k : C^k(\Delta^{[n]}) &\longrightarrow C^k(\Delta^{[n+1]}) \\ \varphi &\longmapsto \left(\sum_{\sigma \in \mathrm{supp}(\varphi)} \sigma \right)^* \end{aligned}$$

The first observation is, that cosystolicity is preserved by this embedding.

Lemma 2.4.1. *Let $\varphi \in C^k(\Delta^{[n]})$ be a cosystole, then $i_n^k(\varphi) \in C^k(\Delta^{[n+1]})$ is a cosystole.*

Proof. Let $\phi \in C^{k-1}(\Delta^{[n+1]})$ and consider the partition $\mathrm{supp}(\phi) = C_1 \cup C_2$ with $C_1 := \{\sigma \in \mathrm{supp}(\phi) : n+1 \in \sigma\}$ and $C_2 = \{\sigma \in \mathrm{supp}(\phi) : n+1 \notin \sigma\}$. Now set $\phi_1 := C_1^*$ and $\phi_2 := C_2^*$, then we have

$$\delta^{k-1}(\phi) + i_n^k(\varphi) = \delta^{k-1}(\phi_1) + \delta^{k-1}(\phi_2) + i_n^k(\varphi),$$

but $\|\delta^{k-1}(\phi_2) + i_n^k(\varphi)\| \geq \|i_n^k(\varphi)\|$ holds by assumption and $\mathrm{supp}(\delta^{k-1}(\phi_1)) \cap \mathrm{supp}(i_n^k(\varphi)) = \emptyset$ holds by the construction of ϕ_1, so we get

$$\|\delta^{k-1}(\phi) + i_n^k(\varphi)\| \geq \|i_n^k(\varphi)\|$$

and we are done. □

Furthermore, if φ is a cosystole, the coboundary expansions of φ and $i_n^k(\varphi)$ are related as follows.

Proposition 2.4.1. *Let $\varphi \in C^k(\Delta^{[n]})$ be a cosystole, then we have:*

$$\|i_n^k(\varphi)\|_{exp} = \|\varphi\|_{exp} + 1$$

Proof. By Lemma 2.4.1 we know that $i_n^k(\varphi)$ is a cosystole and we have $\|\delta^k(i_n^k(\varphi))\| = \|\delta^k(\varphi)\| + \|\varphi\|$, since for every simplex σ in the support of φ, we get the additional simplex $(\sigma, n+1)$ in the support of $\delta^k(i_n^k(\varphi))$. Hence, we have:

$$\|i_n^k(\varphi)\|_{exp} = \frac{\|\delta^k(\varphi)\| + \|\varphi\|}{\|\varphi\|} = \|\varphi\|_{exp} + 1$$

□

2.5 On Cheeger cosystoles when k+2 does not divide n

In [4] and [6] Meshulam and Kozlov gave examples of Cheeger cosystoles in $C^k(\Delta^{[n]})$ when $k+2$ divides n. Recall, that in this case the Cheeger constant is explicitly determined by $h_k(\Delta^{[n]}) = \frac{n}{k+2}$. For the case when $k+2$ does not divide n, we can still explore some interesting properties about Cheeger cosystoles in terms of their norm.

Proposition 2.5.1. *Let $\varphi \in C^k(\Delta^{[n]})$ be a Cheeger cosystole, such that $k+2$ is a prime number and $\|\varphi\|_{exp} = \frac{n}{k+2}$, then $k+2$ divides n or $k+2$ divides $\|\varphi\|$.*

Proof. We have $\|\delta^k(\varphi)\| = \frac{n\|\varphi\|}{k+2}$, since

$$\frac{\|\delta^k(\varphi)\|}{\|\varphi\|} = \|\varphi\|_{exp} = \frac{n}{k+2} = \frac{n\|\varphi\|}{(k+2)\|\varphi\|} = \frac{\frac{n\|\varphi\|}{k+2}}{\|\varphi\|},$$

but $\|\delta^k(\varphi)\|$ is a natural number, so since $k+2$ is prime $k+2$ must divide n or $\|\varphi\|$. □

An interesting consequence of the preceding statement is, that for the case when we do not know the Cheeger constant which is exactly when $k+2$ does not devide n, we would just have to show that the norm of a Cheeger cosystole can not be a multiple of $k+2$ to prove that the Cheeger constant must be strictly larger than $\frac{n}{k+2}$. This might help especially for the 1-dimensional case studied in the following chapter, since in that case $k+2 = 3$ is prime.

Chapter 3

Cut-minimal graphs and Cheeger graphs of a simplex

In this chapter we start from the work of Kozlov (see [1]) in which a graph theoretical approach to the first Cheeger constant of a simplex was developed. In the course of this approach the so called cut-minimal graphs appeared, which exactly describe 1-cosystoles in a very intuitive way. As a consequence of the first main result of this chapter (Theorem 3.2.1) we will determinine the dimension and partly the homology of the simplicial complex $\mathcal{C}_1(\Delta^{[n]})$. In the second part of this chapter, we face the research on the first Cheeger constant of a simplex by investigating combinatorial properties of the Cheeger graphs which are just the graphic version of the 1-Cheeger cosystoles.

3.1 Basic definitions and properties of cut-minimal graphs

The following definition and some words about motivation and intuition for it can be found in [1] (Definition 2.3).

Definition 3.1.1. *Consider a graph $G = ([n], E)$. For any subsets $A, B \subset [n]$ define*

$$E_G(A, B) \coloneqq \{(v, w) \in E \,:\, v \in A, w \in B\}$$

and

$$NE_G(A, B) \coloneqq \{(v, w) \notin E \,:\, v \in A, w \in B\}$$

*A graph $G = ([n], E)$ is called **cut-minimal**, if for every $S \subset [n]$ we have*

$$|E_G(S, [n] \setminus S)| \leq |NE_G(S, [n] \setminus S)|,$$

which is equivalent to

$$|E_G(S, [n] \setminus S)| \leq \frac{|S|(n - |S|)}{2}$$

Note, that there is a one-to-one correspondence between the graphs on n vertices and the 1-chains as follows:
For a graph $G = ([n], E)$ set $c_G \coloneqq \sum_{e \in E} e \in C_1(\Delta^{[n]})$ and for a chain $c \in C_1(\Delta^{[n]})$ set

$G_c := ([n], E)$, with $E := \text{supp}(c)$. Considering characteristic cochains we also get a one-to-one corresponsing between graphs on n vertices and 1-cochains and it is easy to see that a graph $G = ([n], E)$ is cut-minimal if and only if the corresponding cochain c^*_G is a cosystole.

Remark 3.1.1. *In fact for a graph G to be cut-minimal we only need to require the preceding condition holding for all $S \subset [n]$, such that $1 \leq |S| \leq \frac{n}{2}$, since for all $S \subset [n]$ we have $E_G(S, [n] \setminus S) = E_G([n] \setminus S, S)$ and $NE_G(S, [n] \setminus S) = NE_G([n] \setminus S, S)$.*

Example 3.1.1. *A simple cycle respresented by the graph $G = ([n], E)$ with $E := \{(i, i+1) : 1 \leq i \leq n-1\} \cup \{(n, 1)\}$ is cut-minimal for all $n \geq 7$ as follows. One can easily see that for all $S \subset [n]$, such that $|S| \leq \frac{n}{2}$, we have $|E_G(S, [n] \setminus S)| \leq 2|S|$ and the inequality $2|S| \leq \frac{|S|(n-|S|)}{2}$ holds for all $n \geq |S| + 4$, so by $|S| \leq \frac{n}{2}$ the statement is true for all $n \geq 7$.*

Definition 3.1.2. *For any $n \geq 1$ we define the set of all cut-minimal graphs on n vertices:*

$$CMG(n) := \{G = ([n], E) : G \text{ is cut-minimal}\}$$

3.2 Maximal cut-minimal graphs

Let us study the the topological and simplicial structure of the simplicial complex $\mathcal{C}_1(\Delta^{[n]})$ which was, more generally, already introduced in the preceding chapter. Understanding this complex will help us to explore combinatorial properties of the cut-minimal graphs on a certain number of vertices.
The first thing we will do is to determine the maximum number of edges, a cut-minimal graph on a certain number of vertices can have, which immediately gives us the dimension of $\mathcal{C}_1(\Delta^{[n]})$.

Proposition 3.2.1. $C_{max}(\Delta^{[n]}, 1) \geq \binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2}$

Proof. Let us construct a graph G on n vertices as follows. Choose a set $V' \subset [n]$, such that $|V'| = \lceil \frac{n}{2} \rceil$ and connect each pair of vertices from V' by an edge. Then connect each pair of the remaining $\lfloor \frac{n}{2} \rfloor$ vertices by an edge. In other words our graph consists of two complete graphs. If n is even, they are identical, otherwise they differ by one vertex. In total we get $\binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2}$ edges. We will show that this graph is cut-minimal.
Let $S \subset [n]$ and define $A := S \cap V'$ and $B := S \setminus A$. If n is even, we have:

$$\begin{aligned} |E_G(S, [n] \setminus S)| &= |A|(\frac{n}{2} - |A|) + |B|(\frac{n}{2} - |B|) \\ &= \frac{n|A|}{2} - |A|^2 + \frac{n|B|}{2} - |B|^2 \\ &= \frac{n|S|}{2} - (|A|^2 + |B|^2) \end{aligned}$$

$$\begin{aligned} &\leq \frac{n|S|}{2} - \frac{|S|^2}{2} \\ &= \frac{|S|(n-|S|)}{2}, \end{aligned}$$

where the inequality comes from:

$$\begin{aligned} & (|A|-|B|)^2 \geq 0 \\ \iff \quad & |A|^2 - 2|A||B| + |B|^2 \geq 0 \\ \iff \quad & \frac{|A|^2}{2} - |A||B| + \frac{|B|^2}{2} \geq 0 \\ \iff \quad & |A|^2 + |B|^2 \geq \frac{|A|^2}{2} + |A||B| + \frac{|B|^2}{2} \\ \iff \quad & |A|^2 + |B|^2 \geq \frac{|S|^2}{2}. \end{aligned}$$

If n is odd, we have:

$$\begin{aligned} |E_G(S, [n] \setminus S)| &= |A|(\frac{n+1}{2} - |A|) + |B|(\frac{n-1}{2} - |B|) \\ &= \frac{n|A|}{2} + \frac{|A|}{2} - |A|^2 + \frac{n|B|}{2} - \frac{|B|}{2} - |B|^2 \\ &= \frac{n|S|}{2} - (|A|^2 + |B|^2 - \frac{|A|-|B|}{2}) \\ &\leq \frac{n|S|}{2} - \frac{|S|^2}{2} \\ &= \frac{|S|(n-|S|)}{2}, \end{aligned}$$

where the inequality comes from:

$$\begin{aligned} & (|A|-|B|)^2 - (|A|-|B|) \geq 0 \\ \iff \quad & (|A|-|B|)^2 - |A| + |B| \geq 0 \\ \iff \quad & |A|^2 + |B|^2 - |A| + |B| \geq 2|A||B| \\ \iff \quad & \frac{|A|^2}{2} + \frac{|B|^2}{2} - \frac{|A|}{2} + \frac{|B|}{2} \geq |A||B| \\ \iff \quad & |A|^2 + |B|^2 - \frac{|A|}{2} + \frac{|B|}{2} \geq \frac{|A|^2}{2} + \frac{2|A||B|}{2} + \frac{|B|^2}{2} \\ \iff \quad & |A|^2 + |B|^2 - \frac{|A|-|B|}{2} \geq \frac{(|A|+|B|)^2}{2} \\ \iff \quad & |A|^2 + |B|^2 - \frac{|A|-|B|}{2} \geq \frac{|S|^2}{2}. \end{aligned}$$

Hence, the constructed graph is cut-minimal. □

Note, that we already proved the equivalent statement as Proposition 2.3.4, since a simple calculation immediately shows that $\binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2} = \binom{n}{2} - \left\lfloor \frac{n^2}{4} \right\rfloor$. Even-though, it seemed to be worth to state this alternative proof, as it gets along with

pure combinatorics, without using any algebraic properties. Furthermore, it gives an exact description of the "shape" of such a large cut-minimal graph.

Remark 3.2.1. *For further proofs and calculations it might be helpful to keep mind that we have:*

$$\binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2} = \frac{n^2 - 2n + 1}{4}, \qquad \text{for } n \text{ odd, and}$$
$$\binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2} = \frac{n^2 - 2n}{4}, \qquad \text{for } n \text{ even.}$$

Now, we are able to state the main theorem of this section, saying that the preceding estimate is even an upper bound.

Theorem 3.2.1. $C_{max}(\Delta^{[n]}, 1) = \binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2}$

Proof. In a cut-minimal graph the maximum degree of each vertex is $\lfloor \frac{n-1}{2} \rfloor$, so if n is even, by Remark 3.2.1 we have:

$$C_{max}(\Delta^{[n]}, 1) \leq \frac{n \lfloor \frac{n-1}{2} \rfloor}{2} = \frac{n^2 - 2n}{4} = \binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2}$$

If n is odd, the situation becomes more complicated. We only know that

$$C_{max}(\Delta^{[n]}, 1) \leq \frac{n \lfloor \frac{n-1}{2} \rfloor}{2} = \frac{n^2 - n}{4},$$

but in this case unfortunately the right hand side is bigger than $\binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2}$, so we have to find a smaller upper bound for $C_{max}(\Delta^{[n]}, 1)$. The following investigation shows, that a graph with $\frac{n^2-n}{4}$ edges can not be cut-minimal anymore, which will lead to the requested bound.
Consider a cut-minimal graph $G = ([n], E)$, and choose a vertex $v \in [n]$, such that $\deg_G(v) = \frac{n-1}{2}$. If such a vertex does not exist, using Remark 3.2.1 we have

$$|E| \leq \frac{n(\frac{n-1}{2} - 1)}{2} = \frac{n^2 - 3n}{4} < \frac{n^2 - 2n + 1}{4} = \binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2}$$

and we are done. Now there exist exactly $\frac{n-1}{2}$ vertices $v_1, \ldots, v_{\frac{n-1}{2}} \in [n]$, such that $(v, v_i) \notin E$ for all $i = 1, \ldots, \frac{n-1}{2}$. If we had $\deg_G(v_i) = \frac{n-1}{2}$ for one of these vertices, we would get

$$|E_G(\{v, v_i\}, [n] \setminus \{v, v_i\})| = 2\frac{n-1}{2} = n - 1 > n - 2 = \frac{2(n-2)}{2},$$

so G would not be cut-minimal anymore. It follows that the degree of these $\frac{n-1}{2}$ vertices has to be at least one lower than assumed, so the number of edges has to be

at least $\frac{n-1}{4}$ lower than assumed, which provides the new inequality:

$$\begin{aligned} C_{max}(\Delta^{[n]}, 1) &\leq \frac{n^2 - n}{4} - \frac{n-1}{4} \\ &= \frac{n^2 - 2n + 1}{4} \\ &= \binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2} \end{aligned}$$

Hence, we have $C_{max}(\Delta^{[n]}, 1) \leq \binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2}$ in general and by Proposition 3.2.1 we are done. □

A formula for the dimension of $\mathcal{C}_1(\Delta^{[n]})$ follows immediately from the preceding theorem.

Corollary 3.2.1. $\dim(\mathcal{C}_1(\Delta^{[n]}) = \binom{\lceil \frac{n}{2} \rceil}{2} + \binom{\lfloor \frac{n}{2} \rfloor}{2} - 1$

We have seen that the proof of Proposition 3.2.1 even contains a description of the shape of a cut-minimal graph with the maximum number of edges, which furthermore provides information about how a top dimensional simplex is embedded in $\mathcal{C}_1(\Delta^{[n]})$. Now, the following theorem which was developed in [10] does not only represent a first statement about the homology of $\mathcal{C}_1(\Delta^{[n]})$ but even shows that the construction in the proof of Proposition 3.2.1 is the only possible shape of such graphs (top dimensional simplices respectively).

Theorem 3.2.2. $H_{\dim(\mathcal{C}_1(\Delta^{[n]}))}(\mathcal{C}_1(\Delta^{[n]})) \cong 0$

Proof. We will first show, that a top dimensional simplex in $\mathcal{C}_1(\Delta^{[n]})$ can only be represented as a graph of the type we constructed in the proof of Proposition 3.2.1.
Let n be even. If we set $n = 2t + 2$, then by the number $C_{max}(\Delta^{[n]}, 1)$ and cut-minimality, the graph G corresponding to a top dimensional simplex in $\mathcal{C}_1(\Delta^{[n]})$ must be t-regular. Furthermore, for any three pairwise distinct vertices $v, w, u \in [n]$ by cut-minimality we have

$$E_G(\{v, w, u\}, [n] \setminus \{v, w, u\}) \leq \frac{3(2t-1)}{2} = 3t - \frac{3}{2},$$

so by t-regularity among any three vertices at least two of them must be adjacent. Now choose a vertex $v \in [n]$ and set A to be the set consisting of all vertices, which are not adjacent to v. Then we have $|A| = t + 1$ by t-regularity and by the preceding result any two vertices in A have to be adjacent. Thus, A provides a complete graph on $t + 1$ vertices, so by t-regularity $[n] \setminus A$ must also provide a complete graph on $t + 1$ vertices. Hence, every top dimensional simplex in $\mathcal{C}_1(\Delta^{[n]})$ (for n even) corresponds to a graph of that shape.
Let now n be odd. If we set $n = 2t + 3$, then by the number $C_{max}(\Delta^{[n]}, 1)$ and cut-minimality there exist at least $t + 2$ vertices having degree $t + 1$. Let A denote the set

of these vertices. For any $v, w \in A$, such that $v \neq w$ we have:

$$E_G(\{v,w\}, [n] \setminus \{v,w\}) \leq \frac{2(2t+3-2)}{2} = 2t+1 < 2t+2 = \deg_G(v) + \deg_G(w),$$

so all vertices in A are adjacent. By cut-minimality again we have $|A| = t+2$, so A provides a complete graph on $t+2$ vertices. The number of remaining edges $\binom{t+1}{2}$ shows, that the remaining $t+1$ vertices must also provide a complete graph, which is disjoint from the first one, because any other constellation would destroy cut-minimality. Thus, every top dimensional simplex in $\mathcal{C}_1(\Delta^{[n]})$ corresponds to a graph of the requested type again.
Now we see that deleting an edge from such a graph (which is the same as deleting a vertex from a top dimensional simplex) produces a graph corresponding to a simplex which appears as a face of one top dimensional simplex, but can not be a face of another top dimensional simplex, since we can not construct a graph of the requested type by adding an edge at any other place than the place where we just deleted it. Thus, all top dimensional simplices are at most connected via simplices of codimension 2, which implies that $\mathcal{C}_1(\Delta^{[n]})$ can be continiously retracted to a complex of codimension 1 and so homology in top dimension vanishes. □

Definition 3.2.1. *A cut-minimal graph $G = ([n], E)$ is called **maximal**, if for all $e \in NE_G([n],[n])$ the graph $G^e := ([n], E \cup \{e\})$ is not cut-minimal. We denote the set of all maximal cut-minimal graphs on n vertices by $MAX(n)$.*

Since deleting in edge in a cut-minimal graph produces a cut-minimal graph again, it turnes out that we only have to determine all maximal cut-minimal graphs to find all cut-minimal graphs.

Definition 3.2.2. *Two graphs $G = ([n], E)$ and $G' = ([n], E')$ are called **isomorphic**, if there exists a map $f : [n] \to [n]$, such that $(i,j) \in E$ if and only if $(f(i), f(j)) \in E'$.*
*For a graph $G = ([n], E)$ the **isomorphism class** of G is the set*

$$\{G' = ([n], E') : G' \text{ and } G \text{ are isomorphic}\}$$

Remark 3.2.2. *Note, that an isomorphism of graphs preserves most properties studied in this chapter, especially cut-minimality and the constant $h(G)$, studied in the next section.*

Figure 3.1 illustrates, how the isomorphism classes of cut-minimal graphs on 6 vertices are arranged, where the connecting lines represent the relations between the classes refering to the deletion or addition of edges. Note, that the choosen representatives in the figure do not always satisfy the property of being a deletion or an addition of the shown representative of a class below or above, but in this case there is always another representative in the class which does. We see that beside the class of maximal cut-minimal graphs constructed in the proof of Proposition 3.2.1, we have three more classes of maximal cut-minimal graphs here.

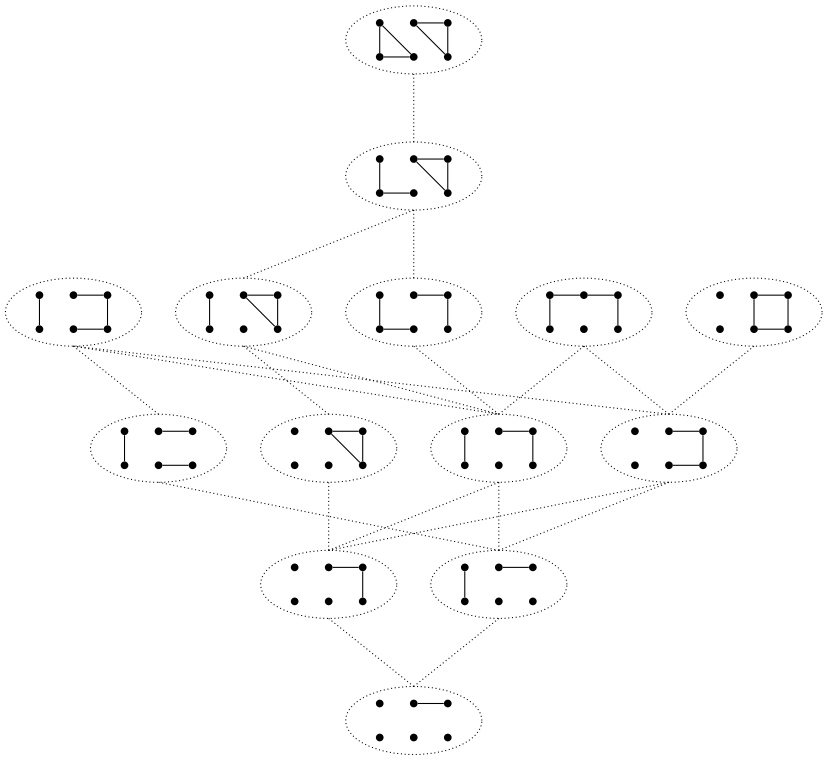

FIGURE 3.1: All cut-minimal graphs on 6 vertices (up to isomorphism)

Except for the maximal cut-minimal graphs with maximum number of edges we do not know anything about the remaining classes of cut-minimal graphs until now. The following statements now approach this challenge by providing new classes of cut-minimal graphs in general which do not appear as deletions of those largest maximal cut-minimal graphs.

Lemma 3.2.1. *Let $u_1, \dots, u_t, s_1, \dots, s_t \in \mathbb{N} \cup \{0\}$, such that for all $i = 1, \dots, t$ we have*

$$u_i + s_i \leq \sum_{\substack{j=1 \\ j \neq i}}^{t} u_j + s_j,$$

then we have:

$$\sum_{i=1}^{t} s_i (u_i - \sum_{\substack{j=1 \\ j \neq i}}^{t} u_j) \leq 0$$

Proof. If we have

$$u_i \leq \sum_{\substack{j=1 \\ j \neq i}}^{t} u_j$$

for all $i = 1, \dots, t$, then we are done, since all u_i's and s_i's are positive. So, let there exist a $k \in [t]$, such that

$$u_k > \sum_{\substack{j=1 \\ j \neq k}}^{t} u_j$$

Obviously, there is at most one unique u_k satisfying this property. Now for all $i \neq k$ we have

$$u_i - \sum_{\substack{j=1 \\ j\neq i}}^{t} u_j \leq -(u_k - \sum_{\substack{j=1 \\ j\neq k}}^{t} u_j),$$

and furthermore we have

$$s_k < \sum_{\substack{j=1 \\ j\neq k}}^{t} s_j$$

by assumption and so we get:

$$\sum_{\substack{i=1 \\ i\neq k}}^{t} s_i(u_i - \sum_{\substack{j=1 \\ j\neq i}}^{t} u_j) \leq -s_k(u_k - \sum_{\substack{j=1 \\ j\neq k}}^{t} u_j)$$

Hence, we have:

$$\sum_{i=1}^{t} s_i(u_i - \sum_{\substack{j=1 \\ j\neq i}}^{t} u_j) \leq 0$$

□

Proposition 3.2.2. *Let the largest connected component of a graph $G = ([n], E)$ not contain more than $\frac{n}{2}$ vertices, then G is cut-minimal.*

Proof. Let $C_1, \ldots, C_t \subset [n]$ be the connected components of G and let $S \subset [n]$, $S_i := S \cap C_i$ and $U_i := C_i \setminus S_i$ for all $i = 1, \ldots, t$. Then we have

$$|E_G(S, [n] \setminus S)| \leq \sum_{i=1}^{t} |S_i||U_i|$$

and

$$|NE_G(S, [n] \setminus S)| \geq \sum_{i=1}^{t} (|S_i| \sum_{\substack{j=1 \\ j\neq i}}^{t} |U_j|)$$

So, we have to show that

$$\sum_{i=1}^{t} |S_i||U_i| \leq \sum_{i=1}^{t} (|S_i| \sum_{\substack{j=1 \\ j\neq i}}^{t} |U_j|),$$

which is equivalent to

$$\sum_{i=1}^{t} |S_i|(|U_i| - \sum_{\substack{j=1 \\ j\neq i}}^{t} |U_j|) \leq 0,$$

but this follows directly from Lemma 3.2.1, since by the assumption $|C_i| \leq \frac{n}{2}$ for all i, we have:

$$|S_i| + |U_i| = |C_i| \leq \sum_{\substack{j=1 \\ j \neq i}}^{t} |C_j| = \sum_{\substack{j=1 \\ j \neq i}}^{t} |S_j| + |U_j|$$

□

Let us now determine the counterpart of the number $C_{max}(\Delta^{[n]}, 1)$, namely the minimal number of edges a non-cut-minimal graph can have, as it was already (more generally) defined in the preceding chapter.

Theorem 3.2.3. $C_{min}(\Delta^{[n]}, 1) = \lceil \frac{n}{2} \rceil$

Proof. By Proposition 2.3.1 we get $C_{min}(\Delta^{[n]}, 1) \leq \lceil \frac{n}{2} \rceil$. On the other hand if we have a graph $G = ([n], E)$ with $|E| = \lceil \frac{n}{2} \rceil - 1$, then it must be cut-minimal, since for any set of vertices $S \subset [n]$, such that $|S| < \frac{n}{2}$ we have $\frac{|S|(n-|S|)}{2} \leq \frac{(|S|+1)(n-(|S|+1))}{2}$ and we have $|E| = \lceil \frac{n}{2} \rceil - 1 = \lfloor \frac{n-1}{2} \rfloor$, so inductively $|E(S, [n] \setminus S)| \leq \frac{|S|(n-|S|)}{2}$ holds for every $S \subset [n]$ such that $|S| \leq \frac{n}{2}$. Hence, we get $C_{min}(\Delta^{[n]}, 1) \geq \lceil \frac{n}{2} \rceil$ and we are done. □

Obviously, $\mathcal{C}_1(\Delta^{[n]})$ contains all simplices of dimension lower than $C_{min}(\Delta^{[n]}, 1)$, which leads to the following observation.

Corollary 3.2.2. $H_k(\mathcal{C}_1(\Delta^{[n]})) \cong 0$ *for all* $1 \leq k \leq \lceil \frac{n}{2} \rceil - 3$

Proof. By Theorem 3.2.3 $\mathcal{C}_1(\Delta^{[n]})$ has a full k-skeleton for all $k \leq \lceil \frac{n}{2} \rceil - 2$ and we are done. □

Since adding a vertex to a cut-minimal graph will always preserve its property to be cut-minimal (see Lemma 2.4.1), we can define the following natural embedding:

$$\begin{aligned} i_n : CMG(n) &\longrightarrow CMG(n+1) \\ ([n], E) &\longmapsto ([n+1], E) \end{aligned}$$

Note, that the embedding i_n is just the graphic version of the more general embedding i_n^k for cochains from the preceding chapter, restricted to cut-minimal graphs. Now, the first thing we see is that maximality of cut-minimal graphs always becomes destroyed by embedding them.

Proposition 3.2.3. *Let* $G \in MAX(n)$*, then we have* $i_n(G) \notin MAX(n+1)$.

Proof. Let $G = ([n], E) \in MAX(n)$. If n is odd, Theorem 3.2.1 gives

$$|E| \leq \frac{n^2 - 2n + 1}{4} < \frac{n^2 - n}{4} = \frac{n\frac{n-1}{2}}{2},$$

so there exists a $v \in [n]$, such that $\deg_G(v) < \frac{n-1}{2}$. Now define
$G' := ([n+1], E \cup \{(v, n+1)\})$ and let $S \subset [n+1]$, such that $1 \leq |S| \leq \frac{n+1}{2}$, then we have

$$\begin{aligned} |E_{G'}(S, [n+1] \setminus S)| &\leq |E_G(S \setminus \{n+1\}, [n] \setminus S)| + 1 \\ &\leq \frac{|S|(n-|S|)}{2} + 1 \\ &= \frac{|S|(n + \frac{2}{|S|} - |S|)}{2} \\ &\leq \frac{|S|(n+1-|S|)}{2}, \end{aligned}$$

for all $S \subset [n+1]$, such that $|S| \geq 2$. For $|S| = 1$, the upper condition is also satisfied, since we have $\deg_G(v) < \frac{n-1}{2}$.
Hence, G' is cut-minimal and so we have $i_n(G) = ([n+1], E) \notin MAX(n+1)$.
If n is even, define $G' := ([n+1], E \cup (v, n+1))$ for an arbitrary $v \in [n]$. Then by the same calculations as in the first part we have

$$|E_{G'}(S, [n+1] \setminus S)| \leq \frac{|S|(n+1-|S|)}{2},$$

for all $S \subset [n+1]$ such that $|S| \geq 2$, and for $|S| = 1$ we have:

$$\begin{aligned} |E_{G'}(S, [n+1] \setminus S)| &\leq \max\{\deg_{G'}(w) : w \in [n+1]\} \\ &\leq \left\lfloor \frac{n-1}{2} \right\rfloor + 1 \\ &= \frac{n-2}{2} + 1 = \frac{n}{2} = \frac{(n+1)-1}{2} \end{aligned}$$

Hence, G' is cut-minimal and $i_n(G) \notin MAX(n+1)$. □

3.3 Basic definitions and properties of Cheeger graphs

The following definition is completely adopted from [1] (Definition 2.6).

Definition 3.3.1. *Consider a graph $G = ([n], E)$, then we set:*

$$T(G) := \{(v, e) \,:\, v \in [n], e = (w, u) \in E, v \notin e, |\{(v, w), (v, u), (w, u)\} \cap E\}| \text{ is odd}\}$$

We have

$$|T(G)| = \sum_{e \in E} t(e),$$

where for an edge $e = (v, w) \in E$, we set

$$t(e) := \sum_{\substack{u \in [n] \\ u \neq v, w}} \tau_e(u),$$

with

$$\tau_e(u) := \begin{cases} 1, & \text{if } (v,u),(w,u) \notin E \\ \frac{1}{3}, & \text{if } (v,u),(w,u) \in E \\ 0, & \text{otherwise} \end{cases}$$

Furthermore, we define

$$h(G) := \frac{|T(G)|}{|E|},$$

*and call a cut-minimal graph $G = ([n], E)$ a **Cheeger graph**, if*

$$h(G) = \min_{G' \in CMG(n)} h(G')$$

*The **first Cheeger constant of a simplex** is defined by*

$$h_1(\Delta^{[n]}) := h(G),$$

where G is some Cheeger graph on n vertices.

We already know by [1] (Theorem 4.1 and Corollary 4.3) that we have $\frac{n}{3} \leq h_1(\Delta[n]) \leq \lceil \frac{n}{3} \rceil$ and the lower bound is archieved if n is not a power of 2. If two graphs G and G' belong to the same isomorphism class, we obviously have $|T(G)| = |T(G')|$ and $h(G) = h(G')$, so taking up the example from the preceding section, Figure 3.2 shows the numbers $h(G)$ for all cut-minimal graphs on 6 vertices with the same partially ordering as in Figure 3.1 and we can see that there is one isomorphy class of Cheeger graphs attaining the Cheeger constant $\frac{8}{4}$.

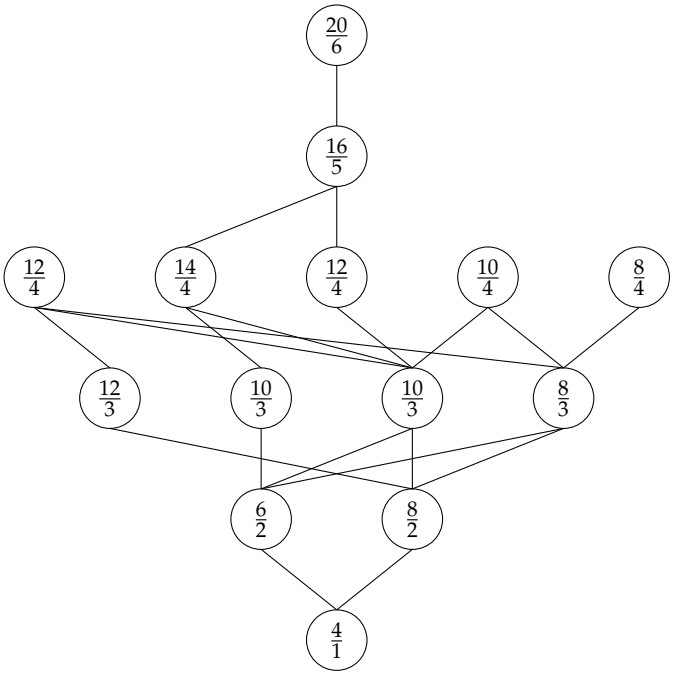

FIGURE 3.2: The numbers $h(G)$ for all cut-minimal graphs on 6 vertices

3.4 A restriction on the size of connected components in Cheeger graphs

The following statement restricts the shape of Cheeger graphs in terms of the sizes of their connected components.

Proposition 3.4.1. *Let $G = ([n], E)$ be a Cheeger graph, then G must have a connected component of size at least $n - \lceil \frac{n}{3} \rceil$.*

Proof. Let $C \subseteq [n]$ be the largest connected component of G, then we obviously have

$$\left\lceil \frac{n}{3} \right\rceil \geq h(G) \geq \frac{|E|(n - |C|)}{|E|} = n - |C|,$$

which is equivalent to

$$|C| \geq n - \left\lceil \frac{n}{3} \right\rceil$$

□

A direct consequence of the preceding statement is, that for $n \geq 6$ a Cheeger graph can never appear as a subgraph of a largest cut-minimal graph such as constructed in the proof of Proposition 3.2.1.

3.5 Embeddings of Cheeger graphs

Based on the fact from the preceding chapter (see Lemma 2.4.1) that adding a vertex to a cut-minimal graph will always result in a cut-minimal graph again, we will now study the consequences of adding a vertex to a Cheeger graph. It will turn out that the resulting graph will not be a Cheeger graph anymore. Using this fact we will be able to find a lower bound on the number of edges in Cheeger graphs. The following statement is just the graphic version of Proposition 2.4.1.

Proposition 3.5.1. *Let $G = ([n], E)$ be a cut-minimal graph, then we have:*

$$h(i_n(G)) = h(G) + 1$$

Proposition 3.5.2. *Let $G \in CMG(n)$, then $i_n(G) \in CMG(n+1)$ is not a Cheeger graph.*

Proof. Without loss of generality let G be a Cheeger graph. Otherwise, there exists a graph $G' \in CMG(n)$, such that $h(G') < h(G)$, so by Proposition 3.5.1 we get:

$$h(i_n(G')) = h(G') + 1 < h(G) + 1 = h(i_n(G))$$

Let now $n+1$ not be a power of 2. Since we have $h(G) \geq \frac{n}{3}$, by Proposition 3.5.1 we get

$$h(i_n(G)) \geq \frac{n}{3} + 1 = \frac{n+3}{3} > \frac{n+1}{3},$$

so $i_n(G)$ can not be a Cheeger graph. On the other hand if $n+1$ is a power of 2, then n can not be a power of 2 and we have:

$$h(i_n(G)) = h(G) + 1 = \frac{n}{3} + 1 \geq \left\lceil \frac{n+1}{3} \right\rceil$$

But by [1] (Theorem 4.6) we have that $h_1(\Delta^{[n+1]}) < \lceil \frac{n+1}{3} \rceil$ holds if $n+1$ is a power of 2, so $i_n(G)$ can not be a Cheeger graph. □

We can now calculate a lower bound for the number of edges in Cheeger graphs.

Proposition 3.5.3. *Let $G = ([n], E)$ be a Cheeger graph, then we have $|E| \geq \lceil \frac{n-1}{2} \rceil$.*

Proof. Assume we have $|E| < \lceil \frac{n-1}{2} \rceil$. Then by Theorem 3.2.3 and since G must contain an isolated vertex, there exists a graph $G' \in CMG(n-1)$ such that $i_{n-1}(G') = G$, so by Proposition 3.5.2 G can not be a Cheeger graph and we are done. □

3.6 The case when n is a power of 2

From [1] (Corollary 4.3) we know that $h(n) > \frac{n}{3}$ can only be valid, if n is a power of 2. An interesting question is, if this inequality is always strict for such n. For a graph $G = ([n], E)$ and a vertex $v \in [n]$ let us introduce the notation
$A_v := \{w \in [n] : (v, w) \in E\}$, so we have $|A_v| = \deg_G(v)$. The following useful fact was given by Kozlov (see [1], Section 6.3).

Lemma 3.6.1. *Let $G = ([n], E)$ be a cut-minimal graph, then we have $h(n) = \frac{n}{3}$ if and only if for each vertex $v \in [n]$ we have $|E(A_v, [n] \setminus A_v)| = |NE(A_v, [n] \setminus A_v)|$.*

We can now make a statement about the possible degrees a vertex in a Cheeger graph on $n = 2^t$ vertices can attain, assuming the first Cheeger constant exactly equals $\frac{n}{3}$.

Proposition 3.6.1. *Let $n = 2^t$ and $G = ([n], E)$ be a cut-minimal graph, such that $h(G) = \frac{n}{3}$. Then for each vertex $v \in [n]$ we have $|A_v| \in \{0 \leq i \leq \frac{n-4}{2} : i \text{ is even}\} \setminus \{2\}$.*

Proof. The proof of the fact, that $|A_v|$ is even can be found in [1] (Section 6.3). Since G is cut-minimal and $n = 2^t$, we have:

$$|A_v| = |E(\{v\}, [n] \setminus \{v\})| \leq \left\lfloor \frac{n-1}{2} \right\rfloor = 2^{t-1} - 1$$

But $2^{t-1} - 1$ is always odd, so we get:

$$|A_v| \leq 2^{t-1} - 2 = \frac{n-4}{2} \tag{3.1}$$

Now, assume there exists a $v \in [n]$, such that $|A_v| = 2$, then by Lemma 3.6.1 we have

$$|E(A_v, [n] \setminus A_v)| = |NE(A_v, [n] \setminus A_v)| = \frac{2(n-2)}{2} = n - 2,$$

so there must exist a $w \in A_v$, such that $|A_w| \geq \frac{n-2}{2} = \frac{2^t-2}{2} = 2^{t-1} - 1$, but this is a contradiction to 3.1. □

Example 3.6.1. *In [1] (Section 6.3) Kozlov already used the fact, that a vertex in a graph satisfying the conditions of the preceding proposition can not attain an odd degree to prove the strict bound $h(8) > \frac{8}{3}$. Applying the preceding slightly stronger proposition we get this result immediately, since then every vertex in a Cheeger graph $G = ([8], E)$, satisfying $h(G) = \frac{8}{3}$ must attain degree 0, which obviously leads to a contradiction.*

We can even say more about how the degrees of adjacent vertices interact in some cases as follows:

Lemma 3.6.2. *Let $n = 2^t$ and $G = ([n], E)$ be a cut-minimal graph, such that $h(G) = \frac{n}{3}$. Then for each vertex $v \in [n]$, such that $|A_v| = 4$ we have $|A_w| = \frac{n-4}{2}$ for all $w \in A_v$.*

Proof. Let $v \in [n]$ such that $|A_v| = 4$, then by Lemma 3.6.1 we have

$$|E(A_v, [n] \setminus A_v)| = \frac{|A_v|(n - |A_v|)}{2} = 2^{t+1} - 8$$

so there must exist a $w \in A_v$ such that $|A_w| \geq \frac{2^{t+1}-8}{4} = 2^{t-1} - 2$. Now, applying Proposition 3.6.1 we get that $|A_w| = 2^{t-1} - 2 = \frac{n-4}{2}$ holds for all $w \in A_v$.

□

Chapter 4

A theorem about partitioning consecutive numbers

4.1 Preliminaries

As an introducing example to the main theorem we want to prove later, consider the following "staircase shaped" Young tableau:

1				
2	2			
3	3	3		
4	4	4	4	
5	5	5	5	5

FIGURE 4.1: "Staircase shaped" Young tableau consisting of 15 boxes

We have one box in the first line and one more in every following line. Now, we can "rebuild" this tableau by reordering the lines, such that we have not necessarily one box in the first line anymore, but always have still one more box in every following line and such that we do not have to split the single lines, as the numbers show:

4	4	4	4	2	2	1	
5	5	5	5	5	3	3	3

FIGURE 4.2: Rebuilt Young tableau

The interesting question which will be answered by our main theorem is, if there is always a possibility to reorder the lines of a Young tableau of the first type without having to split them to build a Young tableau of the second type, provided that the numbers of boxes are the same. The question, how many different Young tableaux of this type exist, meaning Young tableaux consisting of a fixed number of boxes such that based on any line the following line has one box more, has already been answered in [5]. Namely, for every odd divisor of the number of boxes, there exists exactly one Young tableau of this type.
Note, that in this chapter we set $\mathbb{N}$ to be the set of natural numbers without 0 and a

sequence of consecutive numbers always denotes a sequence $n_1, \ldots, n_t \in \mathbb{N}$ (or $\mathbb{Z}$), such that $n_{i+1} = n_i + 1$.

Lemma 4.1.1. *Let $n, a, b \in \mathbb{N}$, such that $n < a \leq b$ and $\sum_{i=1}^{n} i = \sum_{i=a}^{b} i$. Then we have $n \geq 2(b - a + 1)$.*

Proof. Note, that $b - a + 1$ is the number of summands in $\sum_{i=a}^{b} i$. Then we have:

$$\begin{aligned} & \sum_{i=1}^{n} i = \sum_{i=a}^{b} i \\ \iff \quad & \frac{n(n+1)}{2} = (a+b)\frac{b-a+1}{2} \\ \iff \quad & \frac{n}{2}(n+1) = (b-a+1)\frac{a+b}{2} \end{aligned}$$

Now, by assumption we have $\frac{a+b}{2} \geq n + 1$, so we get $\frac{n}{2} \geq b - a + 1$, which is equivalent to $n \geq 2(b - a + 1)$. □

Lemma 4.1.2. *For every $m \in \mathbb{N}$ and every $l \in \mathbb{Z}$ there exist pairs of numbers $(x_1, x_1'), \ldots, (x_m, x_m') \in \mathbb{N}^2$, such that $x_1, x_1', \ldots, x_m, x_m'$ are all distinct, $x_i' - x_i = i$ for every $i = 1 \ldots, m$, $l = \min\{x_1, x_1', \ldots, x_m, x_m'\}$ and $\max\{x_1, x_1', \ldots, x_m, x_m'\} \leq 2m + l$.*

Proof. Consider the following pairs (y_i, y_i') for i odd:

$$\left(y_{2\left\lceil \frac{m}{2} \right\rceil - 1}, y'_{2\left\lceil \frac{m}{2} \right\rceil - 1}\right) = \left(1, 2\left\lceil \frac{m}{2} \right\rceil\right), \ldots, (y_1, y_1') = \left(\left\lceil \frac{m}{2} \right\rceil, \left\lceil \frac{m}{2} \right\rceil + 1\right)$$

and the following for i even:

$$\left(y_{2\left\lfloor \frac{m}{2} \right\rfloor}, y'_{2\left\lfloor \frac{m}{2} \right\rfloor}\right) = \left(2\left\lceil \frac{m}{2} \right\rceil + 1, 2m + 1\right), \ldots, (y_2, y_2') = \left(\left\lceil \frac{m}{2} \right\rceil + m, 2m + 2 - \left\lfloor \frac{m}{2} \right\rfloor\right),$$

where $(y_{i+2}, y_{i+2}') = (y_i - 1, y_i' + 1)$ for all $i = 1, \ldots, m - 2$. Obviously, these pairs satisfy the assumption $y_i' - y_i = i$ and we have $\min\{y_1, y_1', \ldots, y_m, y_m'\} = 1$ and $\max\{y_1, y_1', \ldots, y_m, y_m'\} = 2m + 1$ for every $m \in \mathbb{N}$. (In fact, we even have $\max\{y_1, y_1'\} = 2m < 2m + 1$ for the case $m = 1$). Now, set $x_i := y_i + l - 1$ and $x_i' = y_i' + l - 1$ for all $i = 1, \ldots, m$ and we are done. □

4.2 The main theorem

Theorem 4.2.1. *Let $n, a, b \in \mathbb{N}$ ($b \geq a$), such that $\sum_{i=1}^{n} i = \sum_{i=a}^{b} i$, then for every $a \leq t \leq b$ there exists a subset $U_t \subseteq [n]$, such that $U_i \cap U_j = \emptyset$ for all $i \neq j$ we have $[n] = \bigcup_{a \leq t \leq b} U_t$ and $\sum_{i \in U_t} i = t$.*

Proof. Without loss of generality let $a > n$. Otherwise, we have $U_t = \{t\}$ for all $a \leq t \leq n$ and $\sum_{i=1}^{a-1} i = \sum_{i=n+1}^{b} i$, so the remaining problem is reduced to the requested case, since we have $n+1 > a-1$.
Now, denote the number of summands in the sum $\sum_{i=a}^{b} i$ by $s := b-a+1$ and set:

$$\begin{aligned} P = \{p_1, \ldots, p_s\} &:= \{n-2s+1, n-2s+2, \ldots, n-s-1, n-s\}, \\ Q = \{q_1, \ldots, q_s\} &:= \{n-s+1, n-s+2, \ldots, n-1, n\}, \\ R = \{r_1, \ldots, r_s\} &:= \{a, a+1, \ldots, b-1, b\} \end{aligned}$$

We have $P, Q, R \subset \mathbb{N}$, since $n-2s+1 > 0$ holds by Lemma 4.1.1. Furthermore, we see that we have $p_i + q_j = 2n-2s+1$ for all i, j, such that $i+j = s+1$. Now, set $c := 2n-2s+1$ and for every t such that $r_t - c < 0$ consider pairs $\{p_{i_t}, q_{j_t}\}$ and $\left\{p_{i'_t}, q_{j'_t}\right\}$, such that $i_t + j_t = i'_t + j'_t = s+1$, $q_{j'_t} - q_{j_t} = c - r_t$ and all appearing numbers in all pairs are distinct (we actually only have to require that all $q_{j_t}, q_{j'_t}$ are distinct, then all the rest is distinct by the condition $i_t + j_t = i'_t + j'_t = s+1$). We can find those kinds of tuples of pairs for every t satisfying $r_t - c < 0$ by Lemma 4.1.2 as follows. For every t, such that $r_t - c < 0$ there exists a t', such that $r_{t'} - c = -(r_t - c)$, since otherwise we had $\sum_{i=1}^{s} p_i + q_i > \sum_{i=1}^{s} r_i$, because the numbers $r_1 - c, \ldots, r_s - c$ are obviously consecutive, but this is a contradiction to the assumption $\sum_{i=1}^{n} i = \sum_{i=a}^{b} i$. By this observation, there must still exist a t, such that $r_t - c = 0$, since $r_1, \ldots, r_s$ are consecutive numbers, so letting m denote the maximum number, such that there exists a t satisfying $r_t - c = -m$, we have at least $2m+1$ consecutive numbers in Q. Now, choose such a set of $2m+1$ consecutive numbers from Q and let l denote the minimum number in this set, then applying Lemma 4.1.2 we get the requested tuples of pairs satisfying $q_{j'_t} - q_{j_t} = c - r_t$. Finally, set $U_{r_t} := \left\{p_{i'_t}, q_{j_t}\right\}$ and $U_{r_t + 2(c-r_t)} := \left\{p_{i_t}, q_{j'_t}\right\}$ for every t satisfying $r_t - c < 0$ and we have $\sum_{i \in U_{r_t}} i = r_t$ and $\sum_{i \in U_{r_t+2(c-r_t)}} i = r_t + 2(c-r_t)$. For the remaining t's satisfying $r_t - c \geq 0$ we set $U_{r_t} := \{p_{i_t}, q_{j_t}\}$, where $\{p_{i_t}, q_{j_t}\}$ can be any of the remaining pairs, satisfying $p_{i_t} + q_{j_t} = c$. For those t's we have $\sum_{i \in U_{r_t}} i = c$. So, the problem is reduced to a smaller one of the type $\sum_{i=1}^{n-2s} i = \sum_{i=r_k-c}^{r_s-c} i$, where k is the minimal number such that $r_k - c > 0$. By induction we are done as follows. The sets U_{r_t} and $U_{r_t+2(c-r_t)}$, where $r_t - c < 0$ stay the same till the end, whereas the other sets U_{r_t}, where we still have $\sum_{i \in U_{r_t}} i = c < r_t$ will be "filled up" during the next steps of induction. □

The reader may develop a better intuition for the preceding proof by considering the following example.

Example 4.2.1. *Consider $n = 14$, $a = 15$ and $b = 20$, then we obviously have $\sum_{i=1}^{n} i = \sum_{i=a}^{b} i$.*

Now, according to the proof of Theorem 4.2.1 we have $s = 6$ *and we consider the following pairs:*

$$\{p_1, q_6\} = \{3, 14\}, \quad \{p_2, q_5\} = \{4, 13\}, \quad \{p_3, q_4\} = \{5, 12\},$$
$$\{p_4, q_3\} = \{6, 11\}, \quad \{p_5, q_2\} = \{7, 10\}, \quad \{p_6, q_1\} = \{8, 9\}$$

We see that we have $c = 17$*, so by subtracting* c *from the numbers* $15, \ldots, 20$ *only the last 3 numbers stay strictly positive. Now we "swap" the second components of two pairs just as we did in the preceding proof and construct the following sets:*

$$U_{15} = \{3, 12\}, \quad U_{16} = \{6, 10\}, \quad U_{17} = \{8, 9\},$$
$$U_{18} = \{7, 11\}, \quad U_{19} = \{5, 14\}, \quad U_{20} = \{4, 13\}$$

We can see that we already have $\sum_{i \in U_t} i = t$ *for all* $t = 15, \ldots, 19$ *and* $\sum_{i \in U_{20}} i = c = 17 < 20$*, so the problem is reduced to* $\sum_{i=1}^{2} i = 3$ *and inductively we will be done by the next step. Formally this means, that* $U_{15}, \ldots, U_{19}$ *stay the same and we "fill up" the remaining set* U_{20} *with the remaining numbers* $1, 2$*, such that we get* $U_{20} = \{13, 4, 1, 2\}$.

There are obviously many more ways of partitioning numbers in our sense than the one way the "algorithm" in the proof of Theorem 4.2.1 gives us, but until now this is the only working way we know in general and we even still do not know, how many partitions leading to the requested result exist.

Chapter 5

Perspectives

In this last chapter we want to discuss some open problems and and talk about possible approaches for further reseach which have not yet been covered within this thesis and might motivate the interested reader for further research.

5.1 Applying the cycle detection theorem

The cycle detection theorem (Theorem 2.1.1) by Kozlov seems to be a key tool to find new cosystoles, but as discussed in Chapter 2 it is difficult on the one hand to find a large number of cycles on which a given cochain evaluates to 1 and on the other hand, once we found such a family of cycles, to determine its piercing number.
The first approach might be to develop a theory about piercing numbers for families of cycles, which most likely becomes very challenging, since even for the most obvious family of cycles, namely the boundaries of simplices, a proper generalization of Mantel's theorem (see [7]) has not been discovered yet.
An important observation from Chapter 2 also shows, that it is not sufficient only to study those obvious families of cycles. The family $\mathcal{T}_\varphi$ introduced in Chapter 2 is obviously the largest possible family of boundaries of simplices doing the job, since by definition it consists of all boundaries of simplices, on which φ evaluates to 1. But considering Example 2.1.3 we see that this family is not always sufficient to determine to cosystolic norm of φ by its piercing number.
Nevertheless, we conjecture that using general families of cycles might be sufficient to determine all cosystoles.
The second approach is to search for families of cycles, such that the supports of the cycles in a family are paiswise disjoint, so that we do not have to care about piercing numbers anymore and can just apply Corollary 2.1.1. The disadvantige of this option is, that not all the cosystoles can be determined this way, as an example easily shows. Consider the cochain

$$\varphi = (\{1,2\} + \{3,4\})^* \in C^1(\Delta^{[4]})$$

(Figure 5.1 shows how the support of this cochain can be imagined), then we can only find a single cycle in $C_1(\Delta^{[4]})$ satisfying the necessary assumpions, since the support of every second cycle would intersect the support of the first one, but it is

easy to see, that we have $\|\varphi\|_{csy} = 2$. Note, that the preceding cosystole is even a Cheeger cosystole.

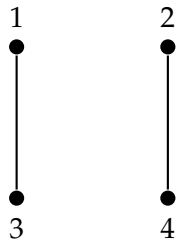

FIGURE 5.1: The support of a 1-cosystole, which can not be determined using disjoint cycles

Even though we can not determine all cosystoles by this "disjoint cycles" method, there is a conjecture, that we might be able to determine all Cheeger cosystoles this way, except for the cases when $k = n - 3$, since all the Cheeger cosystoles we know until now could be determined like that (see [1] and [6]).
Finding large families of cycles with pairwise disjoint supports is a problem, that has been for example approached by Peter Keevash (see [11]), who determined the number of vertices n, for which it is possible to partition the uniform k-skeleton of $\Delta^{[n]}$, such that every part is the boundary of a $(k+1)$-simplex.
For the case of cut-minimal graphs it might be easier to find large families of edge-disjoint cycles, but since we are not only interested in finding cut-minimal graphs but Cheeger graphs, we still have to handle the challenge of constructing graphs G by choosing edges in those cycles, such that the number $T(G)$ becomes as small as possible.

5.2 Finding maximal cosystoles of a simplex

In Theorem 3.2.1 we determined the maximum number of edges a cut-minimal graph can have and even discovered the unique (up to isomorphy) shape of such a graph. It always consists of two disjoint as equally large as possible complete graphs. For higher-dimensional cosystoles by now we have no idea about the largest possible norm and how those cosystoles look like. Perhaps, it might be possible to find a lower bound for the maximal norm of a cosystole in general by choosing an adequate family of cycles and using the cycle detection theorem.

5.3 The shape of Cheeger graphs if n is a power of 2

In [1] (Theorem 4.6) Kozlov gave an upper bound for the Cheeger constant $h_1(\Delta^{[n]})$, if n is a power of 2, using the so called staircase graphs (a certain class of bipartite graphs G for which the number $h(G)$ is easy to determine). It is based on the idea to construct Cheeger graphs inductively by adding edges to a graph $G = ([n], \emptyset)$,

such that G stays cut-minimal and the number $T(G)$ stays as small as possible. Unfortunately, it is not clear, if we will always meet a Cheeger graph on this way or if in some cases only an indirect route (in terms of minimizing $T(G)$) leads to a Cheeger graph. Nevertheless, for all the cases when n is not a power of 2 Kozlov showed that it is possible and that this way always leads to a Cheeger graph, which is a staircase graph. So we conjecture that all Cheeger graphs might be staircase graphs, but even the weaker conjectures, that all Cheeger graphs are bipartite or just triangle-free could not be proven yet. Furthermore, if n is a power of 2, we only know the exact value of the first Cheeger constant for $n = 4$ and $n = 8$ (see [1], Section 4.2), where it is strictly bigger than $\frac{n}{3}$. It would be interesting to find it for higher n at least to know, if it might never exactly attain $\frac{n}{3}$. To approach this challenge, the results from Section 2.5 and Chapter 3 might be helpful as they restrict the shape of Cheeger graphs. Maybe one could make it to produce a contradition to one of those or similar statements if one assumes that $h(G) = \frac{n}{3}$ holds if n is a power of 2.

5.4 The shape of Cheeger cosystoles in general

For general Cheeger cosystoles we have a first statement about their shape as Proposition 2.5.1 restricting their norm. Maybe one could find more statements of this kind as we already did for Cheeger graphs. Note, that all Cheeger graphs which appeared explicitely within this thesis and in [1] and [6] were never subgraphs of the largest cut-minimal graphs we discovered in Chapter 3. One might conjucture, that this property holds in general. Furthermore, one could generalize the concept of staircase graphs by Kozlov (see [1], Section 3) to general cochains. However, the disadvantage of this approach will be that might become technically difficult and if there are cases where we have $h_k(\Delta^{[n]}) > \frac{n}{k+2}$, we will not be able to determine the exact Cheeger constant using this technique by the same reasons as we had for the Cheeger graphs.

5.5 Cheeger constants for other complexes

Another approach for further research which we totally omitted in this thesis is to study the Cheeger constant for other classes of simplicial or polyhedral complexes. For example in [6] (Theorem 4.3) Kozlov showed that for the d-dimensional hypercube Q_d we always have $h_k(Q_d) = 1$. For the more general case of the complex constructed by taking the product of a cell complex and a simplex we only have a rough estimate for the Cheeger constant.

5.6 Probabilistic approaches

Another approach which we totally omitted in this thesis is to use results from probability theory to archieve exact answers to our questions. For example the question,

if the cosystolic norm of every cochain can be determined by the cycle detection theorem might be solved in the following way. If we determined the probability of a randomly chosen cochain $\varphi \in C^k(\Delta^{[n]})$ to be cosystolic and the probability of a ramdomly chosen cochain $\varphi \in C^k(\Delta^{[n]})$ to provide the smallest piercing set of $\mathcal{T}_\varphi$ as its support, we would get the number of cochains which satisfy each of these properties as expectancy values. If they equaled, we would have proven the conjecture. In fact, we would not even need the exact values of those probabilities. Estimations in the right direction would suffice since we know that any cochain $\varphi \in C^k(\Delta^{[n]})$ which is not a cosystole can not provide the smallest piercing set for $\mathcal{T}_\varphi$ as its support.

5.7 Weighted Cheeger constants

Appendix A

Source code for the Cheeger Tool Box (CTB)

The *Cheeger Tool Box (CTB)* is a software library written in C++11, containing methods to handle some important tasks concerning the topic of Cheeger constants of a simplex, such as calculating the cosystolic norm and the coboundary expansion of a given cochain or calculating the k-th Cheeger constant of a simplex on n vertices. A cochain $\varphi \in C^k(\Delta^{[n]})$ is always represented as a column vector with $\binom{n}{k+1}$ (i.e. a matrix of one column and $\binom{n}{k+1}$ lines, such that every entry i is either 1 or 0, depending on if the i-th simplex in the uniform k-skeleton of $\Delta^{[n]}$ is contained in the support of φ. Here, the simplices of the uniform k-skeleton of $\Delta^{[n]}$ are indexed lexicographically based on the contained vertices, which are indexed from 1 to n.
To use the library, the reader should just save the single files in an ordering as shown in the labellings underneath each code block and include the header files into his own code. Note, that by now the methods only use the basic definitions of cosystolicity and Cheeger constants for the calculation. The interested reader is invited to use more results from theoretical research to improve its performance.

```
1  #ifndef MATRIX_H
2  #define MATRIX_H
3
4  typedef std::vector<int> line;
5
6  class Matrix
7  {
8      public:
9
10         Matrix(int pLines, int pColumns);
11
12         // Returns the number of lines of the matrix
13         int getLines();
14
15         // Returns the number of columns of the matrix
16         int getColumns();
17
18         // Returns a certain entry of the matrix
```

```
19          int getEntry(int pLine, int pColumn);
20
21          // Sets a certain entry of the matrix
22          void setEntry(int pLine, int pColumn, int pEntry);
23
24          // Returns a certain column of the matrix as a column matrix
25          Matrix* getColumn(int j);
26
27          // Checks if a given column matrix is contained in the column
        matrix as a subset
28          bool containsColumn(Matrix* column);
29
30          // Returns the norm of the column matrix
31          int getNorm();
32
33          // Returns the cosystolic norm of the column matrix representing a
         k-dimensional cochain of the n-simplex (requires the coboundaryMatrix(
        n,k-1) as coboundaryMat)
34          int getCosystolicNorm(int n, int k, Matrix* coboundaryMat);
35
36          // Returns the coboundary expansion of the column matrix
        representing a k-dimensional cochain of the n-simplex (requires the
        coboundaryMatrix(n,k) as coboundaryMat and the coboundaryMatrix(n,k-1)
        as coboundaryMat1)
37          float getCoboundaryExpansion(int n, int k, Matrix* coboundaryMat,
        Matrix* coboundaryMat1);
38
39          // Checks if the column matrix is a k-dimensional cosystole of the
         n-simplex (requires the coboundaryMatrix(n,k-1) as coboundaryMat)
40          bool isCosystole(int n, int k, Matrix* coboundaryMat);
41
42      private:
43          int lines;
44          int columns;
45          std::vector<line> matrix;
46  };
47
48  #endif
```

CTB/headers/Matrix.h

```
1   #include <vector>
2   #include "../headers/Matrix.h"
3   #include "../headers/basics.h"
4
5   Matrix::Matrix(int pLines, int pColumns)
6   {
7       lines = pLines;
8       columns = pColumns;
9
10      matrix.resize(lines);
11
```

```
12      for (int i = 0; i < lines; i++)
13      {
14          matrix[i].resize(columns);
15      }
16  }
17
18  int Matrix::getLines()
19  {
20      return lines;
21  }
22
23  int Matrix::getColumns()
24  {
25      return columns;
26  }
27
28  int Matrix::getEntry(int pLine, int pColumn)
29  {
30      return matrix[pLine][pColumn];
31  }
32
33  void Matrix::setEntry(int pLine, int pColumn, int pEntry)
34  {
35      matrix[pLine][pColumn] = pEntry;
36  }
37
38  Matrix* Matrix::getColumn(int j)
39  {
40      Matrix* result = new Matrix(lines, 1);
41
42      for (int i = 0; i < lines; i++)
43      {
44          result->setEntry(i, 0, matrix[i][j]);
45      }
46
47      return result;
48  }
49
50  bool Matrix::containsColumn(Matrix* column)
51  {
52      int checker;
53
54      for (int i = 0; i < column->getLines(); i++)
55      {
56          checker = 0;
57
58          for (int l = 0; l < lines; l++)
59          {
60              if (matrix[l][0] == column->getEntry(i,0))
61              {
62                  checker = 1;
63              }
```

```
64          }
65
66          if (checker == 0)
67          {
68              return false;
69          }
70      }
71
72      return true;
73  }
74
75  int Matrix::getNorm()
76  {
77      int result = 0;
78
79      for (int i = 0; i < lines; i++)
80      {
81          result = result + matrix[i][0];
82      }
83
84      return result;
85  }
86
87  int Matrix::getCosystolicNorm(int n, int k, Matrix* coboundaryMat)
88  {
89      int length = nChooseK(n, k);
90      int result = this->getNorm();
91      int tempNorm;
92      Matrix* simplices;
93      Matrix* column;
94      Matrix* cobound;
95      Matrix* tempVec;
96
97      for (int i = 1; i <= length; i++)
98      {
99          simplices = binaryCombinations(length, i);
100
101         for (int j = 0; j < simplices->getColumns(); j++)
102         {
103             column = simplices->getColumn(j);
104             cobound = multMats(coboundaryMat, column);
105             tempVec = addMats(cobound, this);
106
107             tempNorm = tempVec->getNorm();
108
109             if (tempNorm < result)
110             {
111                 result = tempNorm;
112             }
113
114             delete column;
115             delete cobound;
```

```
116                 delete tempVec;
117             }
118
119             delete simplices;
120         }
121
122         return result;
123     }
124
125     float Matrix::getCoboundaryExpansion(int n, int k, Matrix* coboundaryMat,
        Matrix* coboundaryMat1)
126     {
127
128         Matrix* coboundary = multMats(coboundaryMat, this);
129
130         int cosNorm = this->getCosystolicNorm(n, k, coboundaryMat1);
131
132         if (cosNorm == 0)
133         {
134             return 0;
135         }
136
137         float result = (float)coboundary->getNorm() / (float)cosNorm;
138
139         return result;
140     }
141
142     bool Matrix::isCosystole(int n, int k, Matrix* coboundaryMat)
143     {
144         int length = nChooseK(n, k);
145         int norm = this->getNorm();
146         int tempNorm;
147         Matrix* simplices;
148         Matrix* column;
149         Matrix* cobound;
150         Matrix* tempVec;
151
152         for (int i = 1; i <= length; i++)
153         {
154             simplices = binaryCombinations(length, i);
155
156             for (int j = 0; j < simplices->getColumns(); j++)
157             {
158                 column = simplices->getColumn(j);
159                 cobound = multMats(coboundaryMat, column);
160                 tempVec = addMats(cobound, this);
161
162                 tempNorm = tempVec->getNorm();
163
164                 if (tempNorm < norm)
165                 {
166                     delete column;
```

```
167                 delete cobound;
168                 delete tempVec;
169                 delete simplices;
170                 return false;
171             }
172
173             delete column;
174             delete cobound;
175             delete tempVec;
176         }
177
178         delete simplices;
179     }
180
181     return true;
182 }
```

CTB/sources/Matrix.cpp

```
1  #ifndef BASICS_H
2  #define BASICS_H
3
4  // Returns an integer array whose entries are the digits of the binary
       representation of a given integer
5  int* intToBinary(int n);
6
7  // Returns the binomial coefficient n choose k
8  int nChooseK(int n, int k);
9
10 // Returns a matrix with n lines and nChooseK(n,k) columns whose columns
       represent all possibilities how one can arrange k ones on n places (the
        other entries are zeros)
11 Matrix* binaryCombinations(int n, int k);
12
13 // Returns a matrix with k lines and nChooseK(v->getLines(), k) columns
       whose columns represent all possible combinations of k entries from v,
       where v is a column matrix
14 Matrix* vChooseK(Matrix* v, int k);
15
16 // Returns the product of matrix mat1 with matrix mat2
17 Matrix* multMats(Matrix* mat1, Matrix* mat2);
18
19 // Returns the matrix representing the boundary operator from the (k+1)-th
        chain group to the k-th chain group of the n-simplex.
20 Matrix* boundaryMatrix(int n, int k);
21
22 // Returns the matrix representing the coboundary operator from the k-th
       cochain group to the (k+1)-th cochain group of the n-simplex.
23 Matrix* coboundaryMatrix(int n, int k);
24
25 // Returns the transposed of the given matrix M
26 Matrix* transposeMatrix(Matrix* M);
```

```
27
28 // Returns the sum of matrix mat1 with matrix mat2
29 Matrix* addMats(Matrix* mat1, Matrix* mat2);
30
31 // Returns the k-th Cheeger constant of the n-simplex.
32 float cheegerConstant(int n, int k);
33
34 #endif
```

CTB/headers/basics.h

```
1  #include <stdlib.h>
2  #include <math.h>
3  #include <stdio.h>
4  #include <vector>
5  #include "../headers/Matrix.h"
6  #include "../headers/basics.h"
7
8  int* intToBinary(int n)
9  {
10     int digits = floor(log2(n)) + 1;
11     int* result = new int[digits];
12     int temp = n;
13
14     for (int i = digits - 1; i >= 0; i--)
15     {
16         result[i] = temp % 2;
17         temp = floor(temp / 2);
18     }
19
20     return result;
21 }
22
23 int nChooseK(int n, int k)
24 {
25
26     if (k == 0)
27     {
28         return 1;
29     }
30
31     if (2*k > n)
32     {
33         k = n-k;
34     }
35
36     int result = 1;
37
38     for (int i = 1; i <= k; i++)
39     {
40         result = result * (n - k + i) / i;
41     }
```

```
42
43      return result;
44  }
45
46  Matrix* binaryCombinations(int n, int k)
47  {
48      if (n == k)
49      {
50          Matrix* result = new Matrix(n,1);
51
52          for (int i = 0; i < n; i++)
53          {
54              result->setEntry(i, 0, 1);
55          }
56          return result;
57      }
58
59      if (k == 1)
60      {
61          Matrix* result = new Matrix(n,n);
62
63          for (int i = 0; i < n; i++)
64          {
65              for (int j = 0; j < n; j++)
66              {
67                  if (i == j)
68                  {
69                      result->setEntry(i, j, 1);
70                  } else {
71                      result->setEntry(i, j, 0);
72                  }
73              }
74          }
75          return result;
76      }
77
78      Matrix* result = new Matrix(n, nChooseK(n, k));
79      Matrix* temp1 = binaryCombinations(n - 1, k - 1);
80      Matrix* temp2 = binaryCombinations(n - 1, k);
81
82      for (int c = 0; c < nChooseK(n - 1, k - 1); c++)
83      {
84          result->setEntry(0, c, 1);
85
86          for(int i = 1; i <= n-1; i++)
87          {
88              for (int j = 0; j < nChooseK(n - 1, k - 1); j++)
89              {
90                  result->setEntry(i, j, temp1->getEntry(i-1, j));
91              }
92          }
93      }
```

```
94
95      for (int c = nChooseK(n - 1, k - 1); c < nChooseK(n, k); c++)
96      {
97          result->setEntry(0, c, 0);
98
99          for(int i = 1; i <= n-1; i++)
100         {
101             for (int j = nChooseK(n - 1, k - 1); j < nChooseK(n, k); j++)
102             {
103                 result->setEntry(i, j, temp2->getEntry(i-1, j - nChooseK(n
        - 1, k - 1)));
104             }
105         }
106     }
107
108     delete temp1;
109     delete temp2;
110
111     return result;
112 }
113
114 Matrix* vChooseK(Matrix* v, int k)
115 {
116     Matrix* result = new Matrix(k, nChooseK(v->getLines(), k));
117     Matrix* binCombs = binaryCombinations(v->getLines(), k);
118     int counter;
119
120     for (int i = 0; i < binCombs->getColumns(); i++)
121     {
122         counter = 0;
123
124         for (int j = 0; j < binCombs->getLines(); j++)
125         {
126             if (binCombs->getEntry(j, i) == 1)
127             {
128                 result->setEntry(counter, i, v->getEntry(j,0));
129                 counter++;
130             }
131         }
132     }
133
134     delete binCombs;
135     return result;
136 }
137
138 Matrix* multMats(Matrix* mat1, Matrix* mat2)
139 {
140     int temp;
141     int prod;
142
143     Matrix* result = new Matrix(mat1->getLines(), mat2->getColumns());
144
```

```
145     for (int i = 0; i < mat1->getLines(); i++)
146     {
147         for (int j = 0; j < mat2->getColumns(); j++)
148         {
149             temp = 0;
150 
151             for (int n = 0; n < mat1->getColumns(); n++)
152             {
153                 if (mat1->getEntry(i,n) == 1 && mat2->getEntry(n,j) == 1)
154                 {
155                     if (temp == 1)
156                     {
157                         temp = 0;
158                     } else {
159                         temp = 1;
160                     }
161                 }
162             }
163             result->setEntry(i,j,temp);
164         }
165     }
166 
167     return result;
168 }
169 
170 Matrix* transposeMatrix(Matrix* M)
171 {
172     Matrix* result = new Matrix(M->getColumns(), M->getLines());
173 
174     for (int i = 0; i < M->getLines(); i++)
175     {
176         for (int j = 0; j < M->getColumns(); j++)
177         {
178             result->setEntry(j, i, M->getEntry(i,j));
179         }
180     }
181 
182     return result;
183 }
184 
185 Matrix* boundaryMatrix(int n, int k)
186 {
187     Matrix* result = new Matrix(nChooseK(n, k+1), nChooseK(n, k+2));
188     Matrix* vertices = new Matrix(n, 1);
189     Matrix* columnJ;
190     Matrix* columnI;
191 
192     for (int i = 0; i < n; i++)
193     {
194         vertices->setEntry(i, 0, i+1);
195     }
196 
```

```
197     Matrix* s1 = vChooseK(vertices, k+2);
198     Matrix* s0 = vChooseK(vertices, k+1);
199 
200     for (int j = 0; j < result->getColumns(); j++)
201     {
202         columnJ = s1->getColumn(j);
203 
204         for (int i = 0; i < result->getLines(); i++)
205         {
206             columnI = s0->getColumn(i);
207 
208             if (columnJ->containsColumn(columnI) == true)
209             {
210                 result->setEntry(i, j, 1);
211             } else {
212                 result->setEntry(i, j, 0);
213             }
214 
215             delete columnI;
216         }
217 
218         delete columnJ;
219     }
220 
221     delete vertices;
222     delete s1;
223     delete s0;
224 
225     return result;
226 }
227 
228 Matrix* coboundaryMatrix(int n, int k)
229 {
230     Matrix* result = transposeMatrix(boundaryMatrix(n, k));
231 
232     return result;
233 }
234 
235 Matrix* addMats(Matrix* mat1, Matrix* mat2)
236 {
237     Matrix* result = new Matrix(mat1->getLines(), mat1->getColumns());
238 
239     for (int i = 0; i < result->getLines(); i++)
240     {
241         for (int j = 0; j < result->getColumns(); j++)
242         {
243             if (mat1->getEntry(i, j) == mat2->getEntry(i, j))
244             {
245                 result->setEntry(i, j, 0);
246             } else {
247                 result->setEntry(i, j, 1);
248             }
```

```
249         }
250     }
251
252     return result;
253 }
254
255 float cheegerConstant(int n, int k)
256 {
257     float result = ( n / (k + 2) ) + 1;
258     float tempRes;
259     Matrix* column;
260     Matrix* cobound;
261     Matrix* simplices;
262     Matrix* coboundaryMat = coboundaryMatrix(n, k);
263     Matrix* coboundaryMat1 = coboundaryMatrix(n, k-1);
264
265     int length = nChooseK(n, k+1);
266
267     for (int i = 1; i <= length; i++)
268     {
269         simplices = binaryCombinations(length, i);
270
271         for (int j = 0; j < simplices->getColumns(); j++)
272         {
273             column = simplices->getColumn(j);
274
275             if (column->isCosystole(n, k, coboundaryMat1) == true)
276             {
277                 cobound = multMats(coboundaryMat, column);
278                 tempRes = (float)cobound->getNorm() / (float)column->
    getNorm();
279
280                 if (tempRes < result)
281                 {
282                     result = tempRes;
283                 }
284                 delete cobound;
285             }
286
287             delete column;
288         }
289
290         delete simplices;
291     }
292
293     delete coboundaryMat;
294     delete coboundaryMat1;
295
296     return result;
297 }
```

CTB/sources/basics.cpp

Appendix B

Source code for Partitioning Consecutive Numbers (PCN)

The program *Partitioning Consecutive Numbers (PCN)* realizes the algorithm presented in Chapter 4 as a console application written in C++11. To use it, the reader should just save the single files in an ordering as shown in the labellings underneath each code block and build all files. Then just type *PCN* followed by three arguments seperated by a space character into the console. The first argument is the number n representing the sum from 1 to n, the second and the third argument are the numbers a and b representing the sum from a to b. Make sure, that the conditions of Theorem 4.2.1 are satisfied, meaning $a \leq b$ and the sum from 1 to n should equal the sum from a to b. Then the program produces a matrix whose lines respresent the parts of the partition of $[n]$, such that each line sums up to a number between a and b. The entries of the matrix which are not used become filled with zeros.

```
1  #include "../headers/Matrix.h"
2  #include "../headers/findPartitions.h"
3  #include "../headers/basics.h"
4  #include <iostream>
5  #include <math.h>
6  #include <cstdlib>
7
8  int main(int argn, char *argv[])
9  {
10     // Check number of input arguments
11     if (argn != 4)
12     {
13         displayHowTo();
14         return 0;
15     }
16
17     // Convert input arguments to integers
18     int pN = strtol(argv[1], NULL, 0);
19     int pA = strtol(argv[2], NULL, 0);
20     int pB = strtol(argv[3], NULL, 0);
21
```

```
22      // Check if conversion succeeded
23      if (pN == 0 || pA == 0 || pB == 0)
24      {
25          displayHowTo();
26          return 0;
27      }
28
29      // Check if input arguments satisfy the assumptions of the statement
30      double n = (double) pN;
31      double a = (double) pA;
32      double b = (double) pB;
33
34      if ((n * (n + 1)) / 2 != ((a + b) / 2) * (b - a + 1))
35      {
36          displayHowTo();
37          return 0;
38      }
39
40      // Create the output matrix
41      Matrix* mat = new Matrix(b - a + 1, n);
42
43      // Compute the partitioning
44      findPartitions(n, a, b, mat, 0, 0);
45
46      // Display the output
47      for (int i = 0; i < b - a + 1; i++)
48      {
49          for (int j = 0; j < n; j++)
50          {
51              std::cout << mat->getEntry(i,j) << " ";
52          }
53          std::cout << std::endl;
54      }
55
56      delete mat;
57
58      return 0;
59  }
```

PCN/sources/PCN.cpp

```
1   #pragma once
2   #include <vector>
3
4   typedef std::vector<int> line;
5
6   class Matrix
7   {
8
9       public:
10          Matrix(int pLines, int pColumns);
11
```

```
12          // Get number of lines
13          int getLines();
14
15          // Get size of a line
16          int getColumns();
17
18          // Get a certain entry of the matrix
19          int getEntry(int pLine, int pColumn);
20
21          // Set a certain entry of the matrix
22          void setEntry(int pLine, int pColumn, int pEntry);
23
24      private:
25          int lines;
26          int columns;
27          std::vector<line> matrix;
28  };
```

PCN/headers/Matrix.h

```
1   #include "../headers/Matrix.h"
2   #include "../headers/basics.h"
3
4   Matrix::Matrix(int pLines, int pColumns)
5   {
6       lines = pLines;
7       columns = pColumns;
8
9       matrix.resize(lines);
10
11      for (int i = 0; i < lines; i++)
12      {
13          matrix[i].resize(columns);
14      }
15  }
16
17  int Matrix::getLines()
18  {
19      return lines;
20  }
21
22  int Matrix::getColumns()
23  {
24      return columns;
25  }
26
27  int Matrix::getEntry(int pLine, int pColumn)
28  {
29      return matrix[pLine][pColumn];
30  }
31
32  void Matrix::setEntry(int pLine, int pColumn, int pEntry)
```

```
33 {
34     matrix[pLine][pColumn] = pEntry;
35 }
```

PCN/sources/Matrix.cpp

```
1  #pragma once
2
3  // Find pairs of natural numbers with distance 'pStep' within the most
       narrow range
4  int findPairsByStep(int pStep, int pSteps, int pShift, bool pComp);
5
6  // Find first non-used number within the range of findPairsByStep
7  int findFirstFreeSpace(int pSteps, int pShift);
8
9  // Find second non-used number within the range of findPairsByStep
10 int findSecondFreeSpace(int pSteps, int pShift);
11
12 // Display how to use PCN
13 void displayHowTo();
```

PCN/headers/basics.h

```
1  #include "../headers/basics.h"
2  #include <iostream>
3  #include <math.h>
4  #include <vector>
5
6  int findPairsByStep(int pStep, int pSteps, int pShift, bool pComp)
7  {
8      if (pStep > pSteps)
9      {
10         return 0;
11     }
12
13     double step = (double) pStep;
14     double steps = (double) pSteps;
15     double result;
16
17     if (pStep % 2 == 1)
18     {
19         result = ceil(steps / 2) - ceil(step / 2);
20     } else {
21         result = (ceil(steps / 2) * 2) + floor(steps / 2) - (step / 2);
22     }
23
24     if (pComp == 0)
25     {
26         return (int) result + pShift;
27     } else {
28         return (int) result + pStep + pShift;
29     }
```

```
30 }
31
32 int findFirstFreeSpace(int pSteps, int pShift)
33 {
34     double steps = (double) pSteps;
35     double result;
36     result = (2 * ceil(steps / 2)) + floor(steps / 2);
37     return (int) result + pShift;
38 }
39
40 int findSecondFreeSpace(int pSteps, int pShift)
41 {
42     if (pSteps == 1)
43     {
44         return 2 + pShift;
45     } else if (pSteps == 0) {
46         return pShift;
47     } else {
48         double steps = (double) pSteps;
49         double result;
50         result = (2 * ceil(steps / 2)) + (2 * floor(steps / 2)) + 1;
51         return (int) result + pShift;
52     }
53 }
54
55 void displayHowTo()
56 {
57     std::cout << "PCN - Partitioning Consecutive Numbers (written by Kai
        Renken, 2018)" << std::endl;
58     std::cout << std::endl;
59     std::cout << "Type 'PCN [Parameter n] [Parameter a] [Parameter b]'," 
        << std::endl;
60     std::cout << "such that the sum of the numbers 1 to n equals the sum
        of the numbers a to b" << std::endl;
61 }
```

PCN/sources/basics.cpp

```
1 #pragma once
2
3 void findPartitions(int pN, int pA, int pB, Matrix* pMat, int pIter, int
      pFirstLine);
```

PCN/headers/findPartitions.h

```
1 #include "../headers/basics.h"
2 #include "../headers/Matrix.h"
3 #include "../headers/findPartitions.h"
4 #include <map>
5
6 void findPartitions(int pN, int pA, int pB, Matrix* pMat, int pIter, int
      pFirstLine)
```

```
7  {
8      // Set entries for all numbers from 1 to pN, which are between pA and
       pB and call next iteration step
9      if (pA <= pN)
10     {
11         for (int i = pA; i <= pN; i++)
12         {
13             pMat->setEntry(i - pA + pFirstLine, pIter, i);
14
15             for (int j = pIter + 1; j < pMat->getColumns(); j++)
16             {
17                 pMat->setEntry(i - pA + pFirstLine, j, 0);
18             }
19         }
20
21         for (int i = pN + 1; i <= pB; i++)
22         {
23             pMat->setEntry(i - pA + pFirstLine, pIter, 0);
24         }
25
26         if (pA != 1)
27         {
28             int nextLine = pN - pA + 1 + pFirstLine;
29             findPartitions(pA - 1, pN + 1, pB, pMat, pIter + 1, nextLine);
30         }
31
32         return;
33     }
34
35     // Define some variables needed for the computation
36     int s = pB - pA + 1;
37     int c = (2 * pN) - (2 * s) + 1;
38     int m;
39     int steps = c - pA;
40     int posX;
41     int posY;
42     int temp;
43     int nextLine;
44
45     if (steps < 0)
46     {
47         steps = 0;
48     }
49
50     std::map<int,int> result;
51
52     for (int i = pN; i > pN - s; i--)
53     {
54         result[i] = c - i;
55     }
56
57     int i = pA;
```

```
58      m = c - i;
59
60      // Find pairs for negative summing and swap second components
61      while (m > 0)
62      {
63          posX = findPairsByStep(m, steps, pN - s + 1, 0);
64          posY = findPairsByStep(m, steps, pN - s + 1, 1);
65          temp = result[posX];
66          result[posX] = result[posY];
67          result[posY] = temp;
68          pMat->setEntry(i - pA + pFirstLine, pIter, posX);
69          pMat->setEntry(i - pA + pFirstLine, pIter + 1, result[posX]);
70          pMat->setEntry(i - pA + (2 * m) + pFirstLine, pIter, posY);
71          pMat->setEntry(i - pA + (2 * m) + pFirstLine, pIter + 1, result[
        posY]);
72          i++;
73          m = c - i;
74      }
75
76      // Find pair for zero summing
77      if (steps > 0)
78      {
79          posX = findFirstFreeSpace(steps, pN - s + 1);
80          pMat->setEntry(i - pA + pFirstLine, pIter, posX);
81          pMat->setEntry(i - pA + pFirstLine, pIter + 1, result[posX]);
82      }
83
84      posX = findSecondFreeSpace(steps, pN - s + 1);
85
86      // Find pairs for non-zero summing
87      if (steps == 0)
88      {
89          for (i = 0; i < s; i++)
90          {
91              pMat->setEntry(i + pFirstLine, pIter, posX);
92              pMat->setEntry(i + pFirstLine, pIter + 1, result[posX]);
93              posX++;
94          }
95
96      } else {
97          for (i = (2 * steps) + 1; i < s; i++)
98          {
99              pMat->setEntry(i + pFirstLine, pIter, posX);
100             pMat->setEntry(i + pFirstLine, pIter + 1, result[posX]);
101             posX++;
102         }
103     }
104
105     // Call next iteration step
106     if (pN - (2 * s) > 0)
107     {
108         for (int i = pFirstLine; i < s + pFirstLine; i++)
```

```
109         {
110             posX = pMat->getEntry(i, pIter);
111             posY = pMat->getEntry(i, pIter + 1);
112 
113             if (posX + posY != pA + i - pFirstLine)
114             {
115                 nextLine = i;
116                 break;
117             }
118         }
119 
120         for (int i = pFirstLine; i < nextLine; i++)
121         {
122             for (int j = pIter + 2; j < pMat->getColumns(); j++)
123             {
124                 pMat->setEntry(i, j, 0);
125             }
126         }
127 
128         if (c - pA >= 0)
129         {
130             findPartitions(pN - (2 * s), pA + (2 * steps) + 1 - c, pB - c,
     pMat, pIter + 2, nextLine);
131         } else {
132             findPartitions(pN - (2 * s), pA - c, pB - c, pMat, pIter + 2,
    nextLine);
133         }
134 
135     // Set remaining entries to zero
136     } else {
137         for (int i = pIter + 2; i < pMat->getColumns(); i++)
138         {
139             pMat->setEntry(pMat->getLines() - 1, i, 0);
140         }
141     }
142 }
```

PCN/sources/findPartitions.cpp

Bibliography

[1] Dmitry N. Kozlov, *The first Cheeger constant of a simplex*, Graphs and Combinatorics (2017) 33: 1543. https://doi.org/10.1007/s00373-017-1853-9

[2] N. Linial, R. Meshulam, *Homological connectivity of random 2-complexes*, Combinatorica 26, 2006, no. 4, 475-487

[3] M. Gromov, *Singularities, expanders and topology of maps. Part 2. From combinatorics to topology via algebraic isoperimetry*, Geom. Funct. Anal. 20, (2010), no. 2, 416-526.

[4] M. Wallach and R. Meshulam, *Homological connectivity of random k-dimensional complexes*, Random Structures Algorithms 34, 2009, no. 3, 408–417

[5] J. J. Sylvester and F. Franklin, *A Constructive Theory of Partitions, Arranged in Three Acts, an Interact and an Exodion*, American Journal of Mathematics Vol. 5, No. 1 (1882), pp. 251-330

[6] Dmitry N. Kozlov and Roy Meshulam, *Quantitative aspects of acyclicity*, arXiv:1802.03210 [math.CO], 2018

[7] Peter Keevash, *Hypergraph Turán problems*, Surveys in Combinatorics, Cambridge University Press, 2011

[8] Anders Björner and Martin Tancer, *Combinatorial Alexander Duality - A Short and Elementary Proof*, Discrete Comput Geom (2009) 42: 586–593 DOI 10.1007/s00454-008-9102-x, Springer Science+Business Media, LLC 2008

[9] Tim Lindemann, *Personal communication*, University of Bremen, 2018

[10] Dmitry N. Kozlov, *Personal communication*, University of Bremen, 2017

[11] Peter Keevash, *The existence of designs*, http://people.maths.ox.ac.uk/keevash/papers/designsI.pdf, 2018