Karlsruhe Institute of Technology

Higher-Dimensional Triangulations, k-Trees, Schnyder Realizers, and their Contact Representations

Masterthesis of

Tobias Knorr

at the Department of Informatics
Institute of Theoretical Informatics (ITI)

Reviewer:	Dr. T. Ueckert
Second reviewer:	Prof. Dr. T. Bläsius
Advisor:	Dr. T. Ueckert

November 26th 2022 – Mai 26th 2023

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.
Karlsruhe, den 5. Juli 2020

(Tobias Knorr)

Abstract

Schnyder realizers are a well-known concept for planar triangulations. We define a more generalized version of Schnyder realizers with more than three colors for triangulations in higher dimensions. For these triangulations, we notice that there exists a discrepancy between the minimal and maximal possible number of edges. We discover that we can embed every K_n in $\mathbb{R}^3$ as a tetrahedization. For all minimal triangulations of higher dimension, we show their relation to k-trees. We show that minimal triangulations and simple k-trees are the same graph class. Along the way we use a generalized version of canonical orderings for higher-dimensional triangulations. In addition, we show that minimal triangulations induce a Schnyder realizer with k colors. Based on these results and an additional alternative representation of simple k-trees, we find in our research a tetrahedron contact representation and an octahedron side-contact representation for simple 4-trees in $\mathbb{R}^3$.

Kurzfassung

Schnyder-Realisierer sind ein bekanntes Konzept für planare Triangulationen. Wir definieren eine verallgemeinerte Version von Schnyder-Realisierern mit mehr als drei Farben für Triangulationen in höheren Dimensionen. Bei diesen Triangulationen stellen wir fest, dass es eine Diskrepanz zwischen der minimalen und maximalen Anzahl von Kanten gibt. Wir entdecken, dass wir jedes K_n in $\mathbb{R}^3$ als Tetraedisierung einbetten können. Für alle minimalen Triangulationen höherer Dimension zeigen wir deren Beziehung zu k-trees auf. Wir zeigen, dass minimale Triangulationen und simple k-trees die gleiche Graphenklasse sind. Zwischendurch verwenden wir eine verallgemeinerte Version von kanonischer Ordnungen für höherdimensionale Triangulationen. Darüber hinaus zeigen wir, dass minimale Triangulationen einen Schnyder-Realisierer mit k Farben beinhalten. Basierend auf diesen Ergebnissen und einer zusätzlichen alternativen Darstellung von simple k-trees finden wir in unserer Forschung eine Tetraeder-Kontakt-Repräsentation und eine Oktaeder-Seitenkontakt-Repräsentation für simple 4-trees in $\mathbb{R}^3$.

Contents

1. Introduction

A planar graph is a graph that can be drawn in the plain. This means for each planar graph exists a drawing where no edge crosses another except in the endpoints. We call a face of a planar graph, the area that is bounded by the surrounding edges. A maximal planar graph has the maximum number of edges. This means we cannot find two non-connected vertices that can be connected and the resulting graph is still planar. A direct result of the maximality of a planar graph is that each face is a triangle. We call the maximal planar graphs planar triangulations. In the following Figure 1.1 a planar triangulation is illustrated.

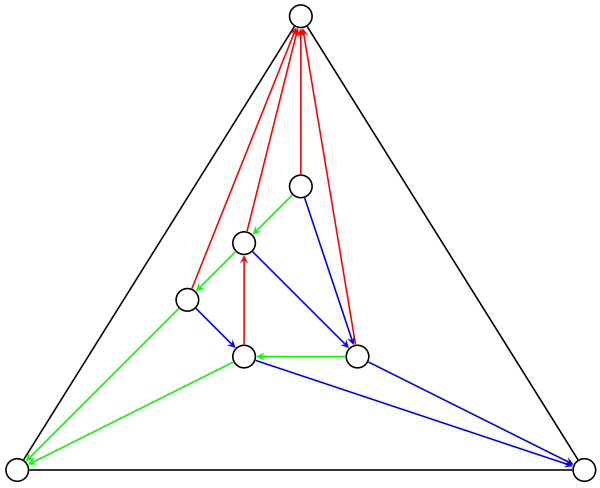


Figure 1.1.: A planar triangulation with a schnyder coloring.

Schnyder realizers are a fundamental concept for drawing planar graphs. In Figure 1.1 a Schnyder realizer is illustrated. The most known use case is the drawing of planar graphs as straight-line drawings.Schnyder realizers are also important for other planar graph representations. Apart from graph drawing, the concept is more widely used like encodings [2], routing algorithms [14] or even quantum mechanics [17].
A Schnyder realizer yields an edge-decomposition of a planar triangulation into three colored trees and an outer triangle. The definition of Schnyder realizers naturally induces the question: What does a graph that has a Schnyder realizer with more than three colors look like, and is there a relation to higher-dimensional triangulations.

Despite our knowledge that the number of edges for a planar triangulation is $3n - 6$ (where n is the number of vertices), we do not know the number of edges of a higher-dimensional triangulation. This number of edges depends on the internal structure of the specific higher-dimensional triangulation. We show that the tight lower bound for the number of edges is

$kn - \binom{k+1}{2}$ for d-dimensional triangulation. For 3-dimensional triangulations, we present a construction to embed K_n for all $n \in \mathbb{N}$ into a triangulation. Therefore, the upper bound for the 3rd dimension is $\binom{n}{2}$.

To show the lower bound of the number of edges for a higher-dimensional triangulation, we look into a generalized version of canonical orderings. For planar triangulations, the canonical ordering respects the structure of the triangulations. In the graph-decomposition of a canonical ordering process, only vertices that are not part of a chord can be removed. For higher-dimensional triangulations, we use a generalized version of a chord we call chord simplex. Based on chord simplices, we define a canonical ordering for triangulations of higher dimension.
If a triangulation is edge-minimal, we find a surface regular canonical ordering, which can be used to find a Schnyder realizer. In addition, it can be shown that these minimal triangulations are simple k-trees.

Based on the theoretical results of this thesis, we present a construction for contact representations for minimal triangulations. We use the fact, that minimal triangulations of higher dimension are simple k-trees. We show in this thesis that minimal triangulations and simple k-trees can be represented as k-color trees and vice versa.
For 4-color trees, we show two different constructions for contact representations. The first construction represents a 4-color tree as a tetrahedron contact representation. The second construction represents the 4-color tree in a hole-free octahedron side-contact representation. The constructions for the shown representations use a common technique based on the structure of the k-color tree. This technique can be used for all k-color trees and for all geometrics objects if a proper invariant for these shapes can be found.

2. Related Work

In this thesis, we look into higher-dimensional triangulations. In the plane we know the number of edges for a triangulation with n vertices is $3n - 6$. For higher-dimensional triangulations, the number of vertices does not determine the number of edges. The book `Triangulations: structures for algorithms and applications` [7, p. 85] describes a way to build higher-dimensional triangulations based on gluing together simplices. The shown construction also produces a triangulation with the minimal number of edges. But since we want to find corresponding graphs to triangulations, we demand, that the convex hull of the triangulation is a simplex. This property distinguishes the triangulations described in the book [7] from the triangulations we observe in this thesis. The triangulations described in the book [7] and other triangulations with a non-simplicial convex hull later will be referred as inner triangulations. We show that the number of edges has a tight lower bound for a d-dimensional triangulation ant that the the bound is $kn - \binom{k+1}{2}$ for $k = d + 1$. In the same book [7] the authors describe the concept of a cyclic polytope $\mathbf{C}(n, d)$. A cyclic polytope is an inner triangulation, which represents a K_n for all $\mathbb{R}^d$. We will use this to find for our definition of triangulations that we can embed for all $n \in \mathbb{N}$ a K_n in the d-dimensional space.

Walter Schnyder firstly introduced Schnyder realizers, also known as Schnyder Woods, in 1989 in his paper `Planar graphs and poset dimension` [19]. Since then, they not only got used in graph drawing [20], but also in other fields like routing algorithms [14]. Schnyder originally defined the Schnyder Woods on planar triangulations, since then, others have tried to find similar constructs for other graphs.
We call a graph connected, if for each vertex exists a path to each of the other vertices. We call a graph $G = (V, E)$ k-connected, if for each subset $\tilde{V} \subset V$ of the vertices of size $k-1$ the induced subgraph $G[V \setminus \tilde{V}]$ is connected.
For 3-connected planar graphs, there is a similar definition of Schnyder realisers [3]. To accommodate the missing edges, some edges are bidirected and bicolored. Others have found a version for graphs that are embeddable on surfaces of higher genus [4].
In this thesis, we look at Schnyder realizers with more than three colors. The concept of d-realizers as described in `Graphs admitting d-realizers: spanning-tree-decompositions and box-representations` [8] are a pure combinatorial generalized version of Schnyder realizers from 3 to d colors. Evans et al. show that any homothetic k-simplex contact representation of a graph has a k-realizer. In addition, they stated that simple k-trees have a k-realizer. Even though, we show based on Schnyder realizers with 4 colors that a simple 4-tree have a tetrahedron contact representation, we cannot answer the question from the paper [8] if a simple 4-tree has a homothetic contact representation. It is unclear what the relation between our generalized definition of Schnyder realizers and the the k-realizers of the paper [8] is. We will discuss this further in the conclusion of this thesis.

Apart from concepts closely related to Schnyder realizers, there are other concepts, which yield other decompositions of graphs. Schnyder subdivides the internal edges into three spanning trees. There are a few edge-decompositions, which subdivide graphs into two spanning subgraphs, sometimes even trees. Regular edge labelings [15] described by G. Kant and X. He are an edge-decomposition of 4-connected planar graphs. Éric Fusy rediscovered the concept and called it traversal structures [11]. For Traversal structures, we can find straight-line drawings [15] and visibility representations [11].
Quadrilanguations are planar graphs and all faces are surrounded by four edges. For quadrilanguations, we can find a separating decompositions [9]. A separating decomposition (Q, Y) is a coloring and orientation Y of the edges from the quadrilateral Q. The vertices of Q are colored black or white and the edges of Q are colored red and blue. The vertices s, t are the sinks of Q and therefore have only red or blue incoming edges, respectively. Depending on the color (black or white) of the other vertices they, are split into two intervals of red and blue edges. The arrangement inside the intervals is an incoming edge and then outgoing edges for a vertex of white color in clockwise order, or vise versa for a black vertex. Seperating decompositions have a relation to Baxter permutations, twin pairs of binary trees, and other related objects [9].

In the second part of this thesis, we show representations for simple 4-trees. For planar triangulations, we can find a lot of different representations and a lot of them are based on Schnyder realizers. The original paper of Walter Schnyder [20] shows an embedding of every planar triangulation onto a grid of size $(n-2) \times (n-2)$. Based on the earlier result of Schnyder, a lot of other representations for planar triangulations have been found. We will now look at contact graphs for planar triangulations.
In the paper `Equilateral L-contact graphs` [5] axis-aligned L-shapes are used to represent a planar triangulation in a contact representation. An L-shape are two orthogonal axis-aligned line segments that are connected in an endpoint. To represent the graph only one of the four rotations is used and the ends of the L-shape therefore end on another orthogonal line segment of an other L-shape corresponding, to an outgoing neighbor in the Schnyder realizer. De Fraysseix et al. describes contact representation with T-shapes [6]. T-shapes axis-aligned T-shapes are two orthogonal axis-aligned line segments that form a T. In this representation oriented axis-aligned T-shapes are used to represent the graph. In the same paper the autors show a Y-shape contact representation. Both representations originate from a triangle contact representation described in the paper. The triangle contact representation induces an edge, if a triangle touches the edge of an other triangle by its edge. This representation is similar to the tetrahedron representation of this thesis. Other tetrahedron contact representations are known for tripartite graphs, for all K_n ($n \in [10]$) and all graphs with less than seven vertices [1].

The octahedron side-contact representation we provide in this thesis is similar to the corresponding hexagon side-contact representation for planar graphs described in `Optimal polygonal representation of planar graphs` [12]. Instead of a touching corner we require for a side-contact representation, that two geometric objects share a line segment in the plane or a polygon for 3-dimensional representations. If we look at the octahedron side-contact representation of a specific simple 4-trees the resulting octahedron side-contact representation is the Sierpinski-tetrahedron [22], but all holes are filled with octahedrons. For better understanding you may look at Figure 5.15.

3. Theoretical Background

This chapter offers an introduction to the fundamental knowledge needed to understand this thesis. Part of the introduction is a clarification of definitions and notations used in this thesis. Firstly, we introduce Schnyder realizers for planar graphs. Based on the definition of the original properties of Schnyder realizers for three colors, we define a more general version for more than three colors. After clarifying what Schnyder realizers are, we focus on a graph class known as k-trees. By looking at simple k-trees we will give an intuition why they naturally induce a Schnyder realizer with k colors. In the next chapter we will show this property of simple k-trees. The Schnyder coloring will be useful in the third and fourth chapter for our discussion about calculating contact representations for these kind of graphs. Prior to looking at the calculations of the representations, we will introduce a compact encoding of simple k-trees.

3.1. Preliminaries

In this section, we introduce the most common notations used throughout this thesis. We call $G = (V, E)$ a graph, V or $V(G)$ are called the vertices of G and $E = E(G) \subseteq \binom{V}{2}$ the edges of G. For this section v is a vertex, V, W are vertex sets, E, F are edge sets and $G = (V, E), H = (W, F)$ are graphs :

notation	definition	meaning
$[n]$	$\{1, \dots, n\}$	the set of the first n integers
E_n	$([n], \emptyset)$	the empty graph with n vertices
K_n	$([n], \binom{[n]}{2})$	the ‘complete graph with n vertices
P_n	$([n], \{\{i, i+1\} \mid i \in [n-1]\})$	the path with n vertices
$N_G(v)$	$\{u \mid \{u, v\} \in E(G)\}$	the neighborhood of v in G
$N^\star_G(v)$	$N_G(v) \dot\cup \{v\}$	the neighborhood of v and v in G
$\deg_G(v)$	$\lvert N_G(v)\rvert$	the degree of vertex v
$H \subseteq G$	$H = (W, F), W \subseteq V, F \subseteq E$	H is a subgraph of G
$G[W]$	$(W, E(G) \cap \binom{W}{2})$	$G[W]$ is the induced subgraph of G by the vertex set $W \subseteq V$
$L(G)$	$(E, \{\{e_1, e_2\} \lvert e_1, e_2 \in E, v \in e_1, v \in e_2\})$	$L(G)$ is the line graph of G

3.2. Schnyder Realizers

In 1990 Walter Schnyder published `Embedding planar graphs on the grid` [20]. In this publication he described the properties of a Schnyder realizer. For any planar triangulated graph $G = (V, E)$ ($n := |V|, m := |E|$) and a chosen outer triangle $\triangle$, we call $E_\triangle = E(\triangle)$ the outer edges of G and $E_\curlywedge = E \backslash E_\triangle$ the inner edges of G. Analogously we define the outer vertices $V_\triangle = V(\triangle) = \{v_r, v_g, v_b\}$ and the inner vertices $V_\curlywedge = V \backslash V_\triangle$. Schnyder proved that we can divide $E_\curlywedge$ into three directed trees T_1, T_2, T_3 and the corresponding edge sets E_1, E_2, E_3. For a more intuitive explanation, we assign a color to each of these trees, denoting T_1 as the red tree ($T_r = (V_\curlywedge \cup \{v_r\}, E_r = E_1)$), T_2 as the green tree ($T_g = (V_\curlywedge \cup \{v_g\}, E_g = E_2)$) and T_3 as the blue tree ($T_b = (V_\curlywedge \cup \{v_b\}, E_b = E_3)$). Because the trees split up $E_\curlywedge$ in disjoint sets, $E = E_\triangle \dot\cup E_r \dot\cup E_g \dot\cup E_b$ holds. When we shift our focus on the number of edges, we know a planar triangulation has $3n - 6$ edges. Further we know each tree with n vertices has $n - 1$ edges. Every of our colored trees uses all inner vertices and one of the outer vertices. As a result $|E| = |E_\triangle| + 3 \cdot |V_\curlywedge| = 3 + 3 \cdot (n - 3) = 3n - 6$. This is no proof that a described division of the edges always exists, but the proof can be found in the original paper from Schnyder. We restate a Schnyder Realizer for the outer triangle $\triangle$ from Schnyders paper:

Definition 3.1 (Schnyder realizer)**.** *For a planar triangulation G, an outer triangle $\triangle = \{v_r, v_g, v_b\}$, an edge coloring which devides the inner edges into E_r, E_g, E_b, and an orientation of the inner edges, we define a Schnyder realizer $R_\triangle = (T_r, T_g, T_b)$. Where $T_x = (V_\curlywedge \cup \{v_x\}, E_x)$ is a tree and all edges of T_x are colored with the color x. The Schnyder realizer holds the following properties:*

1. *For all $v \in V_\curlywedge$, v has exactly one outgoing edge in T_r, T_g and T_b.*
2. *For all $v \in V_\curlywedge$ the edges around v are ordered starting from the outgoing edge in T_r. Going clockwise, v has incoming edges from T_b, the outgoing edge in T_g, incoming edges from T_r, the outgoing edge in T_b, incoming edges from T_g and again the outgoing edge in T_r.*
3. *For all $v_x \in V_\triangle$, v_x has only incoming edges from inside the outer triangle of color x.*

Going forward, we disregard which triangle is the outer triangle, therefore we will denote $R_\triangle$ as R.

Illustration of Schnyder realizer properties

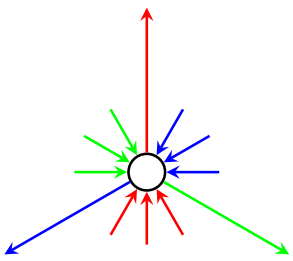


Figure 3.1.: property 1 and 2

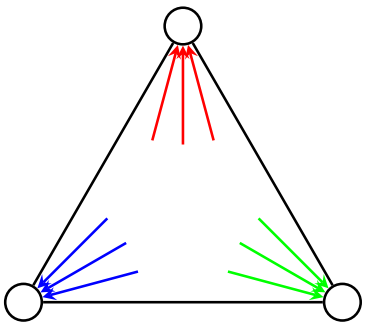


Figure 3.2.: property 3

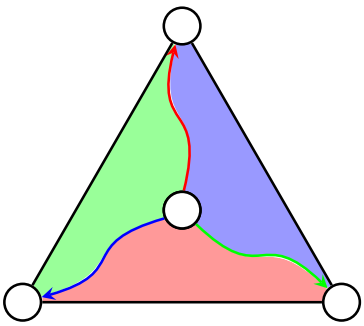


Figure 3.3.: property 4

In Figure 3.1 property 1 and 2 are illustrated. Bear in mind that the inner vertex v may have incoming edges from T_r, T_g and T_b. In other words, it is also possible that two outgoing edges are next to each other and there is no incoming edge between them. For example if v has a red and green outgoing edge next to each other, v is a leaf in the blue tree. In Figure 3.2 property 3 is illustrated. For any outer vertex, the two edges of the outer triangle separate the plane into two regions. The outer region which contains no edges and the inner region that contains all incoming edges of one tree. Since an outer vertex has no outgoing edges, it is the root-vertex of one of the three trees.
Based on the three properties, it is easy to show a fourth property illustrated in Figure 3.3. Given a vertex v, there is a unique path to each root of our colored trees. The paths separate the inside of the outer triangle into three areas. This property is for example used when drawing G in a straight-line drawing.

3.3. Simplices ($\mathcal{S}$)

Definition 3.2. *(k-simplex) For $k \in \mathbb{N}_0$ and a set of k vertices $V = \{v_1, \ldots, v_k\}$ we define a simplex $\mathcal{S}^k = \{s_1, \ldots, s_k\}$.*

For $k \in \mathbb{N}$ a simplex or k-simplex is the simplest polytope in the $(k-1)$-th dimension. If the vertices of a k-simplex are a subset of $\mathbb{R}^{k-1}$, we require that the k points are in general position. k points are in general position, if the coordinates of the points are linearly independent. A point or vertex is a 1-simplex, a line segment is a 2-simplex, a 3-simplex is known as a triangle and a simplex with four points is a tetrahedron. These four simplices can be represented in $\mathbb{R}^3$, for all $k > 4$, all k-simplices cannot be represented.

A k-simplex $\mathcal{S}^k$ has for all $l \in [k-1]$ substructures of size l. The substructures of size l are l-simplices and are subsets of the vertices of $\mathcal{S}$. We call these substructures the l-faces of $\mathcal{S}^k$. the subsets of size l. A simplex $\mathcal{S}$ always contains simplices of a smaller number than k as a substructure. For example the tetrahedron $T = \{v_1, v_2, v_3, v_4\}$ has a 3-face $\{v_1, v_2, v_3\}$ as a substructure, or in other words a triangle.

The vertices and 2-faces of a k-simplex define the graph of a k-simplex, which is always isomorph to a complete graph with k vertices or short K_k.

3.3.1. Barycentric Coordinates

Barycentric coordinates are used to interpolate between points in space. In computer graphics, three points in $\mathbb{R}^2$ form a triangle. To calculate the color of a pixel we need to interpolate between the three colors of the triangle's corners. Let us denote p_1, p_2, p_3 the three corners and c_1, c_2, c_3 their colors, respectively. The pixel p has a position inside the triangle, therefore we can write $p = \lambda_1 \cdot p_1 + \lambda_2 \cdot p_2 + \lambda_3 \cdot p_3$. Since we interpolate between points, the sum $\lambda_1 + \lambda_2 + \lambda_3 = 1$ and $\lambda_1, \lambda_2, \lambda_3 \in [0, 1]$. The color, can be calculated as $c = \lambda_1 \cdot c_1 + \lambda_2 \cdot c_2 + \lambda_3 \cdot c_3$. We now generalize this approach to k points in $\mathbb{R}^{k-1}$.

Definition 3.3 (Convex Combination). *For a given $k \in \mathbb{N}$, k points $P = \{p_1, \ldots, p_k\}$ in $\mathbb{R}^k$, we define a convex combination $\lambda = (\lambda_1, \ldots, \lambda_k)^\intercal \in \mathbb{R}^k$ with the properties: $\sum_{i=1}^{k} \lambda_i = 1$ and $\lambda_1, \ldots, \lambda_k \in [0, 1]$.*

We call $p = bar_P(\lambda) = \sum_{i=1}^{k} = \lambda_i \cdot p_i$ the λ generated barycentric point.

Definition 3.4 (Convex Hull). *For a set of points $P = \{p_1, \ldots, p_k\} \in \mathbb{R}^{k-1}$ we define the convex hull $\mathrm{conv}(P) = \{\mathrm{bar}_P(\lambda) \mid \lambda \in \{x \mid x \in [0,1]^k \wedge \sum_{i=1}^{k} x_i = 1\}\}$. We define the inner convex hull $\mathrm{conv}^\star(P) = \mathrm{conv}(P) \setminus (\bigcup_{x \in \binom{P}{k-1}} \mathrm{conv}(x))$*

3.3.2. Simplicial Complex ($\mathcal{K}$)

For $n, m \in \mathbb{N}$ we can glue together simplices S_1^n, S_2^m by finding a vertex-correspondence for $l \in [\min(n, m)]$ between an l-face in S_1 with an l-face in S_2. After gluing together the two simplices, they both contain the same vertices as an l-face. We call a structure of multiple glued together simplices a simplicial complex $\mathcal{K}$.

Definition 3.5 (k-Simplicial Complex). *For $k \in \mathbb{N}, i \in [k]$ and a vertex set V, we define a k-simplicial complex $\mathcal{K}^k$ as a k-tuple $(T^1, \ldots, T^k)$. The set $T^i \subset \binom{V}{i}$ holds the simplices of size i of $\mathcal{K}^k$. The k-simplicial complex holds the property:*

For all $i \in [k]$ all i-simplices $s \in T^i$: $\emptyset \neq t \subset s \Rightarrow t \in T^{|t|}$.

We call $E = T^2$, T^3 and T^4 the edges, the triangles, and the tetrahedrons of the simplicial complex respectively. Since we have defined vertices and edges, we call $G = (V, E)$ the graph of the simplicial complex.

In addition, we define a few functions on $\mathcal{K}$. We define the dimension of a simplicial complex, as the maximal number of vertices in a simplex in $\mathcal{K}$ ($dim(\mathcal{K}) = \max\{k \mid T^k \neq \emptyset\}$). To get all l-faces of a simplicial complex of dimension k, we define f_l.

$$f_l(\mathcal{K}) = \begin{cases} \emptyset, & dim(\mathcal{K}) < l \\ T^l, & otherwise \end{cases}$$

Definition 3.6 (l-Dual Graph for a k-Simplicial Complex). *For a k-simplicial complex $\mathcal{K}^k$ we define the l-dual graph $d_l^k(\mathcal{K}^k) = (f_k(\mathcal{K}^k), \{(s,t) \mid s, t \in f_k(\mathcal{K}^k) : s \cap t \in f_l(\mathcal{K}^k)\})$*

We call a simplicial complex pure, if each l-simplex(with $l \in [0, k-1]$) in $\mathcal{K}$ is a face of at least one k-simplex in $\mathcal{K}$.

Definition 3.7. *Space-partitioning Drawing For a pure k-simplicial complex $\mathcal{K}^k$, we define a space-partitioning drawing as a straight-line drawing of the underlying graph in $\mathbb{R}^{k-1}$. We choose an outer simplex and call the other simplices, the inner simplices of the simplicial complex. In the drawing only the vertices of the outer simplex of $\mathcal{K}^k$ form the convex hull and the inner vertices lay inside the convex hull. In addition, the inner k-simplices subdivide the convex hull of the drawing into non-intersecting simplices. We call two simplices S_1, S_2 intersecting if a vertex $v \in S_1$ exists such that $v \in \mathrm{conv}^\star(S_2)$.*

Definition 3.8 (Geometric Simplicial Complex)**.** *We call a Simplicial Complex $\mathcal{K}$ a Geometric Simplicial Complex, if $\mathcal{K}$ has a geometric realization in $\mathbb{R}^{dim(\mathcal{K})-1}$ in form of a space partitioning drawing.*

Definition 3.9 (k-Triangulation ($\mathcal{T}^k$))**.** *For $k \geq 2$ a k-triangulation $\mathcal{T}^k$ is a pure geometric k-simplicial complex and every $k-1$-face is part of exactly two k-simplices.*

Remark 3.10. *Because a k-triangulation is a pure geometric k-simplicial complex for every $k-1$-face holds: the face is part of two k-simplices.*

For this thesis we will only look at connected k-triangulations. Therefore k-triangulations have only one connected component in the underlying graph. For $k = 2$ a 2-triangulation is a cycle, a more detailed explanation can be found in 4.1. We call a k-triangulation a triangulation for $k = 3$ and a tetrahedization for $k = 4$.

We will later look at k-triangulations that can be drawn in $\mathbb{R}^{k-1}$. To draw the k-triangulation, we have to choose an outer simplex. All other simplices and vertices of the triangulation are drawn inside this outer simplex. That is why we call the other vertices, edges and simplices that are not part of the outer simplex, inner vertices, inner edges and inner simplices, respectively. When we talk about the convex hull of a k-triangulation, we mean the convex hull of the outer simplex.

Remark 3.11. *A k-triangulation has at least $k + 1$ vertices. If we try to build a k-triangulation with only k vertices, we are only able to build one k-simplex with the k vertices. But for every $k-1$-face of the k-triangulation, the face is part of two k-simplices. As a result no k-triangulation with k vertices is possible. For $k + 1$ vertices, $G = K_{k+1}$ yields a k-triangulation, if we use $\binom{V(G)}{k}$ as the set of k-simplices. The k-simplices $S_1, S_2 \in \binom{V(G)}{k}, S_1 \neq S_2$ share $k-1$ vertices and therefore a $k-1$-face.*

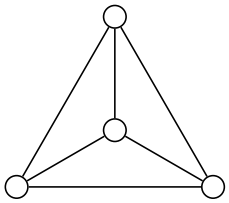


Figure 3.4.: K_4 as a 3-triangulation

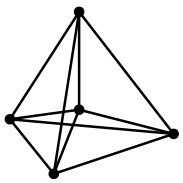


Figure 3.5.: K_5 as a 4-triangulation

Lemma 3.3.1. *For the corresponding graph $G_{\mathcal{T}^k}$ of a k-triangulation $\mathcal{T}^k$ a vertex $v \in V(G_{T^k})$ has degree of at least k.*

Proof. Every vertex v is at least part of two k-simplices, since a k-triangulation is a pure simplicial complex. As a result v is part of at least one $k-1$-face and two different k-simplices. The vertex v is with $k-2$ other vertices part of the $k-1$-face. In addition, each of the k-simplices contains a unique vertex, which the other simplex does not contain. For the corresponding graph G, this gives us at least k neighbors, since v forms a 2-face with each of these vertices. □

We call a k-triangulation $\mathcal{T}^k$ a minimal k-triangulation, if we cannot find another$k$-triangulation with the same number of vertices and fewer edges. Later we will show that the minimal number of edges is $ST^k(|V(\mathcal{T}^k)|) = k \cdot n - \binom{k+1}{2}$ and this lower bound is tight. For now, we only prove that we cannot have a certain type of edges in a minimal k-triangulation.

Definition 3.12 (Removable Edge)**.** *We call an edge $\{v, w\}$ in a k-triangulation $\mathcal{T}^k$ removable, if the common neighbors of v and w induce a $k{-}1$-simplex in $\mathcal{T}^k$, but the simplex is not in $f_{k-1}(\mathcal{T}^k)$.*

Lemma 3.3.2 (Minimal-Removable-Lemma)**.** *A minimal k-triangulation contains no removable edge.*

Proof. We assume we have a minimal k-triangulation $\mathcal{T}^k$ and a removable edge $e = \{v, w\}$. The set $R = \{v, w\} \cup N_{\mathcal{T}^k}(v) \cap N_{\mathcal{T}^k}(w)$ induces the k-simplices around e. By removing e and the adjacent k-simplices from the triangulation, we leave a hole inside the triangulation. This results in $k{-}1$-faces that are no longer part of two k-simplices. We accommodate this, by adding two new k-simplices $R \setminus \{v\}$ and $R \setminus \{w\}$ to the complex. They fill the hole to satisfy the properties of a k-triangulation. In detail, the $k{-}1$-faces $(\binom{R\setminus\{v\}}{k-1} \cup \binom{R\setminus\{w\}}{k-1}) \setminus N_{\mathcal{T}^k}(v)$ are the $k{-}1$-faces which are only part of one simplex after the removal of e. These $k{-}1$-faces are for the glued together[1] simplices $R \setminus \{v\}$ and $R \setminus \{w\}$, the only $k{-}1$-faces which are only part of one simplex. This replacement of simplices inside the triangulation is a contradiction to the minimality. □

Further down the line, we will need to find a decomposition of a k-triangulation. While we decompose the triangulation, we cannot keep our outer simplex, but we can keep our inner structure of the triangulation.

Definition 3.13 (Inner k-Triangulation)**.** *For a $k \geq 3$ and two disjoint vertex sets V_S and V_I we define an inner k-triangulation as a pure geometric k-simplicial complex $\mathcal{T}^k$ on the vertices V_S and V_O. The inner k-triangulation has the property: every $k{-}1$-face that is a subset of V_S is part of exactly one k-simplex all other $k{-}1$-faces are part of two k-simplices.*
We call V_I the inner vertices and V_S the outer vertices of the inner k-triangulation. All $k{-}1$-faces which are a subset of V_S are part of a $k{-}1$-triangulation on the vertices V_S. We call this $k{-}1$-triangulation the surface of the inner k-triangulation and it triangulates the surface of an $k{-}1$-sphere.

Remark 3.14. *The $k{-}1$ dual graph $d^k_{k-1}(\mathcal{T}^k)$ of an inner k-triangulation $\mathcal{T}^k$ is connected. This is a necessary condition for an inner k-triangulation because the surface is a $k{-}1$-triangulation.*

For an inner k-triangulation $\mathcal{T}^k$ and a vertex v of the triangulation we define the functions $N^O_{\mathcal{T}^k}(v) = N_{\mathcal{T}^k}(v) \cap V_S$ as the neighborhood on the surface, and $N^I_{\mathcal{T}^k}(v) = N_{\mathcal{T}_k}(v) \cap V_I$ the neighborhood inside the triangulation.
When we remove only the outer simplex[2] of a k-triangulation the left simplicial complex also holds the properties of an inner k-triangulation. This observation helps us in the next section, so we do not need to distinguish between a k-triangulation and an inner k-triangulation.

[1] $R \setminus \{v\}$ and $R \setminus \{w\}$ are glued together by the $k{-}1$-face $N_{\mathcal{T}^k}(v)$
[2] We only remove the outer simplex. In particular, we keep for all $l \in [k{-}1]$ the l-faces of the outer simplex

3.4. Schnyder Realizers with k Colors

Since we now have defined a Schnyder realizer, a simplex, and a k-triangulation, we can define what we call a Schnyder realizer on k colors or short a k-color realizer ($\mathcal{R}^k$). We define a k-color realizer for a straight-line drawing of a graph G. In particular, even though we will later only look at k-triangulations, we define k-color realizers for graphs with a straight-line drawing in $\mathbb{R}^{k-1}$.
When we refer to an embedding or an embedding in $\mathbb{R}^{k-1}$ of a graph or a k-triangulation in the rest of a thesis, we mean a straight-line drawing and therefore a mapping from the vertices to the points in $\mathbb{R}^{k-1}$. The drawing of the edges and other faces in case of a k-triangulation get induced by the line segment between the vertices or the convex hull of the face.
The convex hull of an embedding is a k-simplex. These k vertices, called the outer vertices $V_\triangle$, form a clique in G. All other vertices $V_\curlywedge$ are elements of $\mathrm{conv}^\star(V_\triangle)$. Analogously, we define the outer and inner edges $E_\triangle$ and $E_\curlywedge$.

In the definition of Schnyder realizers, we use that we can draw a 3-triangulation in the plane and therefore we can order the edges around a vertex clockwise. We look at the chirotope to order the neighboring edges around a vertex.

Definition 3.15 (Chirotope)**.** *For an ordered k-tuple of points $P = (p_1, \ldots, p_k), p_1, \ldots, p_k \in \mathbb{R}^{k-1}$ we define the chirotope $\chi(p_1, \ldots, p_k) : \mathbb{R}^{k-1 \times k} \to \{-1, 0, 1\}$:*

$$\chi(p_1, \ldots, p_k) = \mathrm{sgn}(\det \begin{pmatrix} | & | & \ldots & | \\ p_1 & p_2 & \ldots & p_k \\ | & | & \ldots & | \\ 1 & 1 & \ldots & 1 \end{pmatrix})$$

The chirotope $\chi(P)$ describes the combinatorial orientation of the points P. If the points $p_1, \ldots, p_k$ are not in general position, every ordered tuple T' on these points has the result $\chi(T') = 0$. If the points $p_1, \ldots, p_k$ are in general position based on the order of the points the sign of the results is positive or negative.

We now look at two properties of chirotopes:

Lemma 3.4.1. *For n points $p_1, \ldots, p_n$ in $\mathbb{R}^{n-1}$ and a translation $v \in \mathbb{R}^{n-1}$ the orientation of the chirotope does not change: $\chi(p_1 - v, \ldots, p_n - v) = \chi(p_1, \ldots, p_n)$.*

Proof.

$$\begin{aligned}
\chi(p_1 - v, \dots, p_n - v) &= \operatorname{sgn}(\det \begin{pmatrix} | & | & \dots & | \\ p_1 - v & p_2 - v & \dots & p_n - v \\ | & | & \dots & | \\ 1 & 1 & \dots & 1 \end{pmatrix}) \\
&= \operatorname{sgn}(\det(\begin{pmatrix} | & | & \dots & | \\ p_1 & p_2 & \dots & p_n \\ | & | & \dots & | \\ 1 & 1 & \dots & 1 \end{pmatrix} \cdot \begin{pmatrix} I_{n-1} & -v \\ 0 & 1 \end{pmatrix})) \\
&= \operatorname{sgn}(\det(\begin{pmatrix} | & | & \dots & | \\ p_1 & p_2 & \dots & p_n \\ | & | & \dots & | \\ 1 & 1 & \dots & 1 \end{pmatrix}) \cdot \underbrace{\det(\begin{pmatrix} I_{n-1} & -v \\ 0 & 1 \end{pmatrix})}_{=1}) = \chi(p_1, \dots, p_n)
\end{aligned}$$

□

Lemma 3.4.2. *For n points $p_1, \dots, p_n$ in $\mathbb{R}^{n-1}$ and a point v in $\mathbb{R}$ for n factors $\lambda_1, \dots, \lambda_n$ in $\mathbb{R}, \sum_{i\in[n]} \lambda_i = 1$, if we can find $\lambda_1, \dots, \lambda_n$ such that $v = \sum_{i\in[n]} \lambda_i p_i$ than the chirotope of $p_1, \dots, p_n$ has the following properties for two cases:*

$\forall l \in [k]\colon \lambda_l \in [0, 1]$
For all $i \in [n]\colon \chi(p_1, \dots, p_{i-1}, v, p_{i+1}, \dots, p_n) = \chi(p_1, \dots, p_n)$

$\exists i \in [k]\colon \lambda_i < 0$
$\chi(p_1, \dots p_{i-1}, v, p_{i+1}, \dots, p_n) = -\chi(p_1, \dots, p_n)$

Proof.

$$\begin{aligned}
\chi(p_1, \dots, p_{i-1}, v, p_{i+1}, \dots, p_n) &= \operatorname{sgn}(\det(\begin{pmatrix} | & \dots & | & | & | & \dots & | \\ p_1 & \dots & p_{i-1} & v & p_{i+1} & \dots & p_n \\ | & \dots & | & | & | & \dots & | \\ 1 & \dots & 1 & 1 & 1 & \dots & 1 \end{pmatrix})) \\
&= \operatorname{sgn}(\sum_{l\in[n]} \det(\begin{pmatrix} | & \dots & | & | & | & \dots & | \\ p_1 & \dots & p_{i-1} & \lambda_l p_l & p_{i+1} & \dots & p_n \\ | & \dots & | & | & | & \dots & | \\ 1 & \dots & 1 & \lambda_l & 1 & \dots & 1 \end{pmatrix})) \\
&= \operatorname{sgn}(\sum_{l\in[n]} \lambda_l \underbrace{\det(\begin{pmatrix} | & \dots & | & | & | & \dots & | \\ p_1 & \dots & p_{i-1} & p_l & p_{i+1} & \dots & p_n \\ | & \dots & | & | & | & \dots & | \\ 1 & \dots & 1 & 1 & 1 & \dots & 1 \end{pmatrix})}_{=0, \text{ for } l \neq i}) \\
&= \operatorname{sgn}(\lambda_i \det(\begin{pmatrix} | & \dots & | \\ p_1 & \dots & p_n \\ | & \dots & | \\ 1 & \dots & 1 \end{pmatrix})) = \operatorname{sgn}(\lambda_i) \cdot \chi(p_1, \dots, p_n)
\end{aligned}$$

Therefore, if $\lambda_i \in [0,1] : \chi(p_1,\dots,p_{i-1},v,p_{i+1},\dots,p_n) = \chi(p_1,\dots,p_n)$ and for $\lambda_i < 0 : \chi(p_1,\dots,p_{i-1},v,p_{i+1},\dots,p_n) = -\chi(p_1,\dots p_n)$. □

Since we have to find the orientation of the edges around an inner vertex v, we define the chirotope for an ordered edge k-tuple $(e_1,\dots,e_k), e_i = \{v,w_i\} \in E(G[N^\star(v)]), i \in [k]$ in relation to the vertex v:

$$\chi_v(e_1,\dots,e_k) = \chi(w_1 - v,\dots,w_k - v)$$

With this definition of the orientation of the edges, we can define a Schnyder realizer with k colors.

Definition 3.16 (k-Color Realizer). *For $k > 3$ a given graph with an embedding in $\mathbb{R}^{k-1}$, an outer k-simplex $\triangle = V_\triangle = \{v_1,\dots,v_k\}$, the colors $c_1,\dots,c_k$ and a coloring and orientation of the inner edges we define a k-color realizer $R^k_\triangle = (T_1,\dots,T_k)$. $T_1,\dots,T_k$ subdivide the inner edges of G into $E_1,\dots,E_k$ of color $c_1,\dots,c_k$ respectively. For $x \in [k]$ T_x describs the subgraph of color x : $T_x = (V_\curlywedge \cup \{v_x\}, E_x)$. The realizer $R^k_\triangle$ has the following properties:*

1. *For the outer vertices the orientation of the chirotope $\chi(v_1,\dots,v_k) = 1$ is positive.*
2. *All incoming edges of the outer vertices $v_i, i \in [k]$ are of color c_i.*
3. *For all $v \in V_\curlywedge$ has exactly one outgoing edge of each color $c_1,\dots,c_k$ $e_1,\dots,e_k$ in $T_1,\dots,T_k$ respectively and $\chi_v(e_1,\dots,e_k) = 1$. For the incoming edges of v hold:*

   For $l \in [k], \chi_v(e_1,\dots,e_{l-1},e,e_{l+1},\dots,e_k) = -1$ if e has color c_l.

To find and prove that for each minimal k-triangulation a k-color realizer exists, we now need to define another construct called a k-canonical ordering.

A canonical order or canonical representation, as described in `Small sets supporting Fary embeddings of planar graphs` [10], is an order on the vertices of a planar triangulation. Let $v_1,\dots,v_n$ be the canonical ordered vertices of a planar triangulation. We can reconstruct a triangulation and a proper Schnyder-coloring by the edge set of the triangulation.
Starting with G_3 as the triangle $\{v1,v2,v3\}$ which also forms the border. The border is a cycle $c^i = \{c^i_1, c^i_2,\dots,c^i_s\}$. The vertices $c^i_1 = v_1$ and $c^i_2 = v_2$, always stay on the cycle in the following construction, since they are part of the outer triangle. We construct the triangulation by adding the vertices $v_4,\dots,v_n$ in the provided order. In the i-th step v_i gets connected to $c^{i-1}_l,\dots,c^{i-1}_r, l < r$. As a result v_i gets inserted inbetween c^{i-1}_l and c^{i-1}_r, and the vertices between c^{i-1}_l and c^{i-1}_r get concealed and are no longer on the border of the new graph G_i. When inserting the vertex v_i, we color and orient the new edges. The directed edge $\{v^{i-1}_l, v_i\}$ gets colored green, the edge $\{v^{i-1}_r, v_i\}$ gets colored blue, and all other edges get colored red and are incoming edges of v_i. When inserting v_n we do not color the edges to c^{n-1}_l and c^{n-1}_r because these are two edges of the outer triangle and do not get colored as described in section 3.2. Note that at the start of the construction we color $\{v_3,v_1\}$ and $\{v_3,v_2\}$ blue and green respectively if $n \neq 3$.
In the construction above, we can observe, that in every step of the construction, G_i is an inner 3-triangulation. This observation will be important later.

The resulting coloring of the constructed triangulation is a Schnyder realizer as described in [20] and a canonical ordering always exists as proven in [10]. Since we now have at least a rough understanding about a canonical ordering, we shall define the more general version the k-canonical ordering.

Definition 3.17 (Chord Simplex)**.** *For an inner k-triangulation we call for $l \in [2, k_1]$ a chord l-simplex an induced simplex of l vertices on the surface of an inner k-triangulation, if and only if the induced l-simplex is not part of the surface of the inner k-triangulation.*

Note that a chord $k-1$-simplex is always a simplex in the inside of the inner k-triangulation. By the definition of an inner k-triangulation, every chord $k-1$-simplex is part of two k-simplices. As a result the $k-1$-dual graph of a k-triangulation is always connected if we do not remove a vertex of a chord $k-1$-simplex.

Definition 3.18 (Steady k-Triangulation)**.** *We call a k-triangulation $\mathcal{T}^k$ a steady k-triangulation if for each $\tilde{V} \subseteq V(\mathcal{T}^k)$ with $G(\mathcal{T}^k)[\tilde{V}]$ is an inner k-triangulation $\mathcal{T}^k_{\tilde{V}}$: $\mathcal{T}^k_{\tilde{V}}$ has no chord l-simplex with $l \in [2, k-2]$.*

Assumption 3.19. *For the rest of the thesis we assume all k-triangulations are steady k-triangulations.*

Lemma 3.4.3 (Peeling Lemma)**.** *For any inner k-triangulation of a steady k-triangulation with k or more vertices, there always exist a vertex v which is not part of a chord $k-1$-simplex. We call such a vertex a chord-free vertex. When removing v and all incident edges, the remaining graph is still an inner k-triangulation.*

Proof. We start by looking at an inner k-triangulation a choosen forbidden $k-1$-simplex s_f of the surface and a vertex $v \in V_S \setminus s_f$. If v is not part of a chord $k-1$-simplex, we are done. In the other case we find a chord $k-1$-simplex s. The chord simplex s separates the inner triangulation into to halves. In both halves, s is part of the surface. We choose the half which does not contain the forbidden s_f and mark s as our new s_f. We repeat this process until we find a vertex which is not part of a chord simplex or the half has only k vertices. In the case of k vertices we find a vertex w which was not part of the last chord simplex.

Since we choose every iteration, the half which did not contain a previous forbidden $k-1$-simplex, the vertex w is part of the surface of the original inner triangulation and is not part of a chord simplex. By removing this vertex we keep our inner triangulation. The removal of w may bring new vertices to the surface of the $k-1$-triangulation, but most importantly, we keep the $k-1$-dual graph connected, since we did not remove a chord k-simplex. □

Definition 3.20 (k-Canonical Ordering)**.** *We call an order on the vertices $v_1, \ldots, v_n$ of a k-triangulation $\mathcal{T}^k$ a k-canonical ordering if for every subset $\mathcal{V}_i = \{v_1, \ldots, v_i\}$, the induced subgraph $\mathcal{T}^k_i = G(\mathcal{T}^k)[\mathcal{V}_i]$ is an inner k-triangulation and v_i is a vertex inside V_S for $\mathcal{T}^k[V]$.*

We call a k-canonical ordering $\tilde{v}_1, \ldots, \tilde{v}_n$ l surface regular, if for every step $i \in [k, n]$ we remove a chord-free vertex $\tilde{v}_i$ which has l vertices on the surface of the inner k-triangulation $|N^O_{\mathcal{T}^k[\{\tilde{v}_1,\ldots,\tilde{v}_i\}]}(\tilde{v}_i)| = l$.

Theorem 3.21. *For every steady k-triangulation $\mathcal{T}^k$ we can find at least one k-canonical order of the vertices.*

Proof. We start with a k-triangulation and remove the outer simplex to get an inner k-triangulation $\mathcal{T}^k$. We enumerate the outer vertices $V_\triangle = \{v_1^\triangle, \ldots, v_k^\triangle\}$. For all steps of the decomposition, we choose the simple $s_f = \{v_1^\triangle, \ldots, v_{k-1}^\triangle\}$. We start with $\mathcal{T}_n^k = \mathcal{T}^k$. As long as we have more than k vertices left, we repeat the following step, where i is the number of vertices left. With Lemma 3.4.3, we find a chord-free vertex $\tilde{v}_i$ for the forbidden $k{-}1$-simplex s_f in $\mathcal{T}_i^k$. When there are different chord-free vertices in $\mathcal{T}_i^k$, we have to decide which vertex we choose, by the decision the resulting order is determined and therefore might vary. We remove $\tilde{v}_i$ from $\mathcal{T}_i^k$ and get $\mathcal{T}_{i-1}^k$ as a new inner k-triangulation. When we get to $\mathcal{T}_k^k$ we have only one k-simplex left. The k-simplex contains all vertices of s_f and an additional vertex $\tilde{v}_k$. We have found a canonical order: $v_1^\triangle, \ldots, v_{k-1}^\triangle, \tilde{v}_k, \ldots, \tilde{v}_n = v_k^\triangle$. □

3.5. k-Trees

Trees are one of the simplest if not the simplest non-trivial graph class. We can construct a tree by induction. A graph with one vertex is a tree of size 1. Adding one vertex to a tree of size n and connecting it to one of the vertices of the tree, yields a tree of size $n + 1$. k-trees can be constructed similarly, in particular 1-trees are the same graph class as trees.

Definition 3.22 (k-Tree)**.** *We can construct a k-tree with n vertices in the following inductive way, we start with K_{k+1} and repeat the following steps $n - k - 1$ times:*

1. *Choose k vertices that induce a clique in the current k-tree*
2. *Add a new vertex and connect it to all vertices of the chosen clique*

We can enumerate the vertices $v_1, \ldots, v_n$ by the order in which they were added to the tree and call this order on the vertices a construction ordering.

Lemma 3.5.1. *k-trees are chordal graphs.*

Proof. To show that k-trees are chordal, we argue a perfect elimination ordering [21] exists for every k-tree. We call a vertex a simplicial vertex if its neighborhood is a clique. If we can find a decomposition of a k-tree, where in each step we find a simplicial vertex, then we have found a perfect elimination ordering for the k-tree. We know the vertices v_{k+2} to v_n got added in the order $v_{k+2}, \ldots, v_n$. When a vertex v_i got added in the construction, we connect only the new vertex to a clique, so the vertex is in this step a simplicial vertex. Thus the reverse construction order is a perfect elimination ordering for $v_{k+2}, \ldots, v_n$ Since v_1 to v_{k+1} form a clique, every vertex is also simplicial and we find a perfect elimination ordering $v_n, \ldots, v_1$. □

Lemma 3.5.2. *Every perfect elimination ordering of a k-tree is the reverse of a construction ordering.*

Proof. Keep in mind, in every perfect elimination ordering for a k-tree we remove in every step a vertex v_i which is not part of a clique C_j of another vertex v_j. In other words, the vertex

v_i is at the point of the decomposition only connected to its clique, otherwise there are at least two vertices in the neighborhood of v_i, which are not connected. As a result, when we use the perfect elimination ordering, the right neighborhood of the vertex is the clique of size k, we connected it in the construction of the k-tree. □

3.5.1. Simple k-Trees

In the construction of a k-tree we allow that in every step we can choose a clique among any cliques in the current k-tree. We call a k-tree a simple k-tree if in every step of the construction, we choose a clique that was not chosen in a previous step.

Definition 3.23 (Simple k-Tree)**.** *We can construct a simple k-tree ST^k with n vertices by starting with K_{k+1} and the vertices $v_1, \ldots, v_{k+1}$. To track the already connected cliques, we define $C_{k+1} = \{v_1, \ldots, v_k\}$ and $Q_{k+1} = \{C_{k+1}\}$. Now we repeat the following steps $n - k - 1$ times:*

1. *Find a clique $C_i \notin Q_{i-1}$ in ST^k, connect the vertex v_i to the simple k-tree by connecting it to all vertices of C_i*
2. *Add the clique C_i to the covered cliques $Q_i = Q_{i-1} \dot\cup C_i$*

Lemma 3.5.3. *A simple k-tree is a k-triangulation and has a space-partitioning drawing.*

Proof. A simple k-tree has a construction order $v_1, \ldots, v_n$ by its definition. We assign a position in $\mathbb{R}^{k-1}$ to the first k vertices of the construction in such a way that these outer vertices do not lay on a hyperplane. In addition we require that $\chi(v_1, \ldots, v_k) = 1$. If $\chi(v_1, \ldots, v_k)$ is negative we can swap the positions of v_1 and v_2, the resulting chirotope is positive.
These first k vertices form the outer simplex $S_\triangle$. We now assign a position to the remaining vertices in the construction order and show, that after each step the induced subgraph of the points with a position is a k-triangulation and the positioning of the vertices yields a space-partitioning drawing.

We assign v_{k+1} a position inside $\mathrm{conv}^\star(\{v_1, \ldots, v_k\})$, for example $\mathrm{bar}_{\{v_1,\ldots,v_k\}}((\frac{1}{k}, \ldots, \frac{1}{k})^\intercal)$. Now the induced subgraph for the first $k+1$ vertices form a k-triangulation and a space-partitioning drawing. The simplices of the triangulation are $\mathsf{S} = \binom{\{v_1,\ldots,v_{k+1}\}}{k}$ and for all $S_1, S_2 \in \mathsf{S}, S_1 \neq S_2$, the simplices S_1 and S_2 share a k–1-face $S_1 \cap S_2$.
By placing v_{k+1} inside $S_\triangle$ the simplices around v_{k+1} subdivide the outer simplex $S_\triangle$ into k non-intersecting simplices and therefore. The simplices and therefore the drawing of the induced subgraph of the first $k+1$ vertices is space-partitioning.

Using an inductive argument, we add the rest of the vertices. In the step where we add the vertex v_i, the vertices $v_1, \ldots, v_{i-1}$ induce a k-triangulation and a space-partitioning drawing. In the construction, we connect v_i to a clique C_i. These vertices form a k-simplex in the triangulation and the space-partitioning drawing. We subdivide these simplices the same way we subdivided the outer simplex $S_\triangle$ with the vertex v_{k+1}. As a result, we remove the simplex S_i from the triangulation and the space-partitioning drawing and replace it with the k simplices $\binom{S_i \cup \{v_i\}}{k} \setminus S_i$. After we added the vertex v_i to the triangulation and the space-partitioning drawing, they are still a triangulation and a space-partitioning drawing. □

Conjecture 3.24. *We conjecture that simple k-trees are steady k-triangulations.*

For our work later, we need another representation of a simple k-tree. We use a tree with directed colored edges to represent a simple k-tree. For this representation, the k outer vertices $(v_1, \dots, v_k)$ are no vertices in this tree-representation.

Definition 3.25 (k-Color Tree)**.** *We define a k-color tree CT^k as a directed tree and every edge has one of the k colors $c_1, \dots, c_k$. Every vertex has at most one outgoing edge of one color and at most one edge of each color as an incoming edge.*

Remark 3.26. *If we don't restrict the number of incoming edges of one color, the concept of a k-color tree could be generalized for all k-trees not only simple k-trees. Because we only need the definition in this thesis for simple k-trees, we introduce the k-color trees in the way we defined them in Definition 3.25, even though they should be called simple k-color trees and the name k-color trees should be reserved for the corresponding concept for k-trees.*

Definition 3.27 (k-Vertex Color Set)**.** *We define a k-vertex color set as a tuple of k vertices $(v_1, \dots, v_k)$. The vertices get assigned the colors $c_1, \dots, c_k$according to there position in the tuple.*

We define four functions on a k-vertex color set $\nu = (v_1, \dots, v_k)$ the vertex v, the index i, the color c, and the elements e:

$$v_\nu(l) = v_l \qquad i_\nu(v) = \begin{cases} i & v = v_i \\ \text{undefined} & \text{otherwise} \end{cases} \qquad c_\nu(v) = c_{i_\nu(v)} \qquad e(\nu) = \{v_\nu(l) \mid l \in [k]\}$$

When we use k-vertex color sets we always use a map q that maps from the vertices of a k-color tree to k-vertex color sets.

Construction 3.28 (Simple k-Tree to k-Color Tree Transformation)**.**
When we look at a construction of a simple k-tree, we can define a different structure as $Q_{k+1}, \dots, Q_n$. Instead of remembering already used cliques, we can remember which cliques are available. We start with $W_{k+1} = \binom{\{v_1,\dots,v_{k+1}\}}{k}$. When we add a new vertex v_i, we use clique $C_i \notin Q_{i-1}$. We know $C_i \in W_{i-1}$, so we can define the new set of available cliques $W_i = (W_{i-1} \setminus C_i) \cup \{\{v_i\} \cup x \mid x \in \binom{C_i}{k-1}\}$. When we now connect a new vertex v_i to the vertex of the highest index v_m in the construction order, we get a tree with v_{k+1} as the root. We direct the edges every step from v_i to v_m as a result, we get a directed tree. Our alternative representation of a simple k-tree ST^k is a k-color tree on the vertices $v_{k+1}, \dots, v_n$ and a function $q : V(\mathsf{CT}^k) \to V(\mathsf{ST}^k)^k$.
The mapping q maps vertices in CT^k to k-vertex color sets. Part of this mapping is the implicit coloring of our previously described tree to get the k-color tree. When we connect v_i to C_i in our construction, we know v_m is the last vertex of C_i that we added to the simple k-tree. Because of the way, we constructed W_i we know $q(v_m)$ has only one vertex v_o, which is not part of C_i. We denote $o = i_{q(v_m)}(v_o)$ as the index into the tuple $q(v_m) = (t_1, \dots, t_k)$. We assign $q(v_i) := (t_1, \dots, t_{o-1}, v_m, t_{o+1}, \dots, t_k)$ and color the edge with the color c_o. Because we only replace the o-th element of the tuple $q(v_m)$ once, the resulting tree is a k-color tree, since v_m has only one incoming edge of each color.

Lemma 3.5.4. *For a vertex set $V = \{v_1, \ldots, v_n\}$, $V_\triangle = \{v_1, \ldots, v_k\}$, and $V_\perp = \{v_{k+1}, \ldots, v_n\}$, we know a k-color tree CT^k and a map q that maps from $V_\perp$ to k-vertex color sets are equivalent to a simple k-tree if they hold the following properties:*

1. $V_\triangle \cap V(\mathsf{CT}^k) = \emptyset$ *and* $q(v_{k+1}) = (v_1, \ldots, v_k)$
2. $\forall e = (v, w) \in E(\mathsf{CT}^k)$*:*
   a) e *has color* $c_{q(v)}(w)$
   b) $w = v_{q(v)}(i_{q(v)}(w))$
   c) $|e(q(v)) \cap e(q(w))| = k{-}1$

The properties of the Lemma 3.5.4 follow from the above described construction of a k-color tree from a simple k-tree.

Construction 3.29 (k-Color Tree to Simple k-Tree Transformation)**.**
We can reconstruct the k-tree from a k-color-tree and a mapping q by connecting every vertex $v_i \in V_\perp$ to $e(q(v_i))$. In addition, we have to connect the outer vertices to a k-simplex. When we add the edges in the construction order, we build the simple k-tree as described in the definition. After $v_1, \ldots, v_{k+1}$ are added the k-tree is a K_{k+1}. Any other vertices get connected to their k-vertex color set, which corresponds to the clique of the construction.

When we direct the edges from the inner vertices v_i and give all incoming edges to vertices in $e(q(v_i))$ the color of the vertex in the tuple, we can easily see that each inner vertex and outer vertex fulfill part of the properties of a k-color realizer. But we do not have an embedding for the simple k-tree. We will show in the next chapter that we can find a representation in $\mathbb{R}^{k-1}$ for a simple k-tree, such that all conditions are satisfied and every simple k-tree induces a Schnyder realizer with k colors.

Remark 3.30. *We only need the k-vertex color set in the case of the construction of a k-color tree from a simple k-tree or to construct a simple k-tree from a k-color tree. The only important information stored in the mapping q is the tuple of v_{k+1}. The tuple of v_{k+1} determines, which color gets assigned to which outer vertex, but this information is only important for the naming of the vertices and not the structure of the graph itself, since the outer simplex is a K_k.*

With this realization we know every k-color tree induces a simple k-tree, besides the renaming of the outer vertices. This understanding about k-color trees will help us find an encoding of simple k-trees.

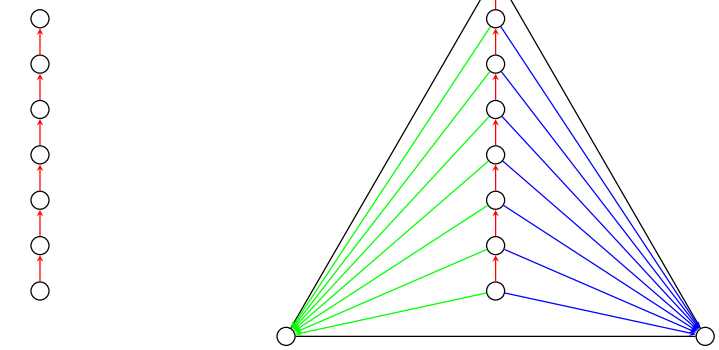
3.5.2. Simple k-Tree Encoding

In this section we will present an intuitive encoding of simple k-trees. This encoding will be useful to describe simple k-trees with a simple notation and even without seeing a representation of the simple k-tree, we can get a rough understanding of its structure. We use the fact that we can convert the representation of a simple k-tree to a k-color tree and vice versa. In order to encode simple k-trees we encode k-color trees and transform them later into simple k-trees.
We start with a set of colors. In our case, we start with four colors, since we will later only look at k-trees with $k \leq 4$. We use the four colors red, green, blue, and lilac. We abbreviate red to r,

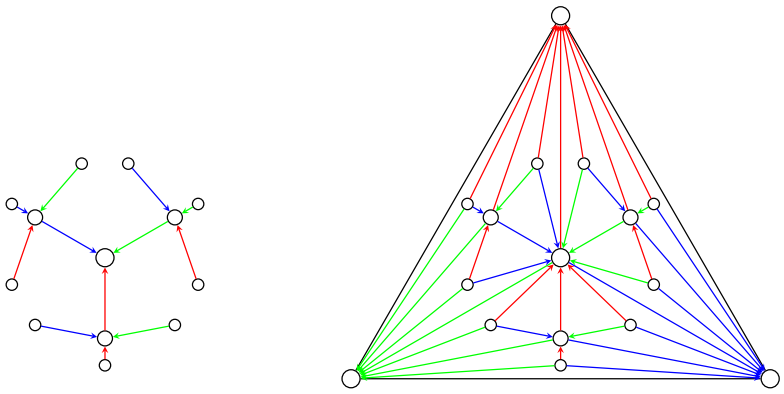
green to g, blue to b and lilac to l. When we look at three colors we only use red, green, and blue.
We decode the encoding as a string. When the string is empty, the k-color tree is a K_{k+1}. In order to construct k-color trees from a string, we keep track, which vertex we added last. We start with v_{k+1} as the last added vertex. When the next character in the string is a color, we add a child under the last added vertex in the k-color tree and color the edge with the color of the character. We remember the new vertex as our last added vertex. If the next character is again a color, we add a new vertex, color the edge and update our last vertex. But with only colors, our k-color tree is a directed colored path. To build a tree, we need another operation. We introduce two new characters " (" and ") ". If the next character is a " (", we push the current last vertex to a stack. If the next character is a ") ", we pop a vertex from the stack and remember it as our last added vertex. This operation introduces a new problem. If we have e.g. the following sequence of characters " (r)r ", we would add two vertices of the red color under v_{k+1}, this would not be a valid k-color tree and therefore no valid simple k-tree. Since we have added two vertices to the same clique of the simple k-tree.

We accommodate this problem, by checking if we already added a vertex of such a color under our last added vertex. In this case, we do not add a new vertex, we only update our last vertex to the child, which is connected with the corresponding color. With these two operations, we can build every possible k-color tree. We only give an intuition on why we can generate all k-color trees. A k-color tree is a tree and for trees, we know that for every two vertices in the tree, there exist a unique path between two vertices. Let us denote the unique paths from v_{k+1} to $v_{k+2}, \ldots, v_n$ as $P^{k+2}, \ldots, P^n$, respectively. For an directed edge $e = a, b$ of CT^k, we can get the color using $c(e) = c_{q(a)}(b)$ and for an path $P = \{p_1, \ldots, p_l\}$ we can get the string that generates this path by appending $c(\{p_1, p_2\}), c(\{p_2, p_3\}), \ldots, c(\{p_{l-1}, p_l\})$. We can get this string with $s(P)$. We can now decode the CT^k with the following string " $(s(P^{k+2}))(s(P^{k+3}))\ldots(s(P^n))$ ".
With only these two operations, it is quite tedious to specify a k-color tree. To remedy this, we introduce two new characters " { " and " } ". Every Operation inside " { " and " } " starts with the last vertex at the point we read " { " in the string. In addition, when we complete an operation inside " { " and " } ", we continue after " } " until we get to the end of the string or reach ") " character, we then continue with the next operation inside the curly brackets. When we evaluated all Operations inside " { " and " } " we jump to the end of the string or after the next ") ".
The next operation that we allow is represented by a number, when we write " 5r " it is the same as " rrrrr ". In the following, we will define a few 3-color trees as CT^k_S with S as a string. We will draw the k-color tree and the simple 3-tree with a Schnyder coloring next to each other.

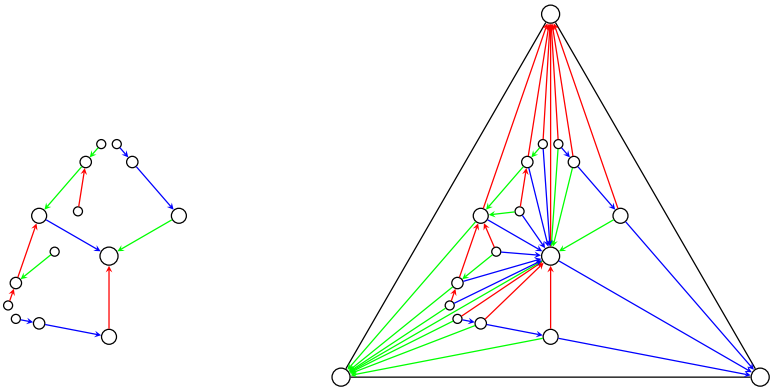
CT^3_{6r}



$\mathsf{CT}^3_{2\{\mathsf{rgb}\}}$



$\mathsf{CT}^3_{(\{\mathsf{rg}\}\mathsf{bb})(\mathsf{b2}\{\mathsf{rg}\})}$



The drawings represent the 3-color trees above such that we can observe that the 3-color trees always are a subgraph of the decoded 3-tree. This holds for all k, not just for $k = 3$. Additional the coloring of the k-color tree is a subset of the colored edges of the Schnyder realizer.

4. Properties of k-Triangulations and Simple k-Trees

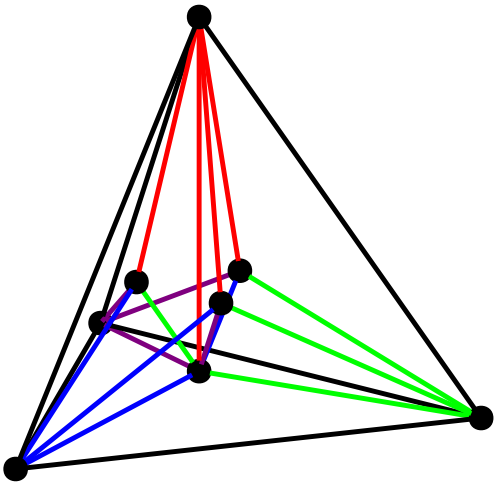
In this chapter, we will show a few properties of k-triangulations and simple k-trees. We start at looking at the number of edges of a k-triangulation. We will notice that for $k = 3$, every triangulation has $3n - 6$ edges, but for $k > 4$, there exists a gap between the minimal and the maximal number of edges. Since we have seen the gap of edges for $k > 3$, we want to know how many simplices are inside an given k-triangulation. After we looked at triangulations, we want to know for which k-triangulations we can find a Schnyder realizer with k colors. We show that a k-triangulation has a k-color realizer if and only if the k-canonical order is $k-1$-surface regular. With this property, we can show that for $k > 3$ minimal triangulations are exactly the same graph class as simple k-trees.

4.1. Number of Edges in a k-Triangulation

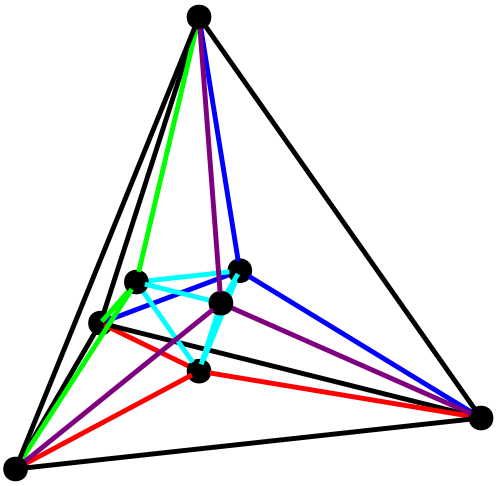
In this section we will look at the number of edges of k-triangulations. When we look at the definition of a k-triangulation, we have defined a k-triangulation as a pure geometric simplicial complex and every $k-1$-face is part of two k-simplices. Note that in the first chapter where we introduced k-triangulations we demanded a connected underlying graph.
As a result for $k = 1$, we cannot find a graph with such a property, since 0-faces are undefined and they would be part of two vertices. Since we have no edges in the underlying graph, it is not even possible to be part of two vertices in any way. For $k = 2$, we get a cycle. A cycle has exactly n edges where n is the number of vertices. This can be shown easily. Every vertex is part of two edges. As a result, we can start walking along the edges. Since every vertex has two edges, we always walk the edge we did not originate from. We cannot visit a vertex multiple times because a vertex only has two edges. In the end we will again arrive at the vertex we have started at, since the number of vertices is finite.

Lemma 4.1.1. *For $k = 3$ a k-triangulation has $3n - 6$ edges.*

Proof. We know for a 3-triangulation $\mathcal{T}^3$ the Euler's formula holds $n - m + f = 2$, where n is the number of vertices, m is the number of edges and f is the number of faces. Because $\mathcal{T}^k$ is a triangulation all faces are triangles. We use the argument of double counting $|\{(e, f) \mid e \in f_2(\mathcal{T}^3) \wedge e \subset t \in f_3(\mathcal{T}^3)\}| = 3f = 2m$, since every edge e is part of two triangles and every triangle t has three edges. When we put this into the Euler's formula we get $2 = n - e + f = n - m + \frac{2}{3}m = n - \frac{1}{3}m$ and therefore $m = 3n - 6$. □



The graph has 8 vertices, 4 edges of color red, green, blue and lilac and 6 edges of black color.



The graph has 8 vertices, 3 edges of color red, green, blue and lilac color and 6 edges each of black and cyan color.

Figure 4.1.: There are two 4-triangulations represented, each with 8 vertices. The graph on the left has 22 edges, while the graph on the right has 24 edges.

Now, we look at the number of edges of a k-triangulation for $k > 3$. For $k \leq 3$ we have shown that the number of edges can be derived from the number of vertices. In the following Figure 4.1, we have two 4-triangulations with 8 vertices, which have a different number of edges.
Due to the fact that the two different 4-triangulations on 8 vertices exist we cannot determine the number of edges for a k-triangulation for $k > 3$ from the number of vertices. To find the number of edges of a k-triangulation for $k > 3$, we have to look at the structure of the k-triangulation. A logical reasoning for this discovery is the lower and upper bound on the number of edges for a k-triangulation with n vertices.

Theorem 4.1. *For every $k > 3$ and any steady k-triangulation $\mathcal{T}^k$ with n vertices, the number of edges is limited by $ST^k(n) \leq |E(\mathcal{T}^k)| \leq \binom{n}{2}$*

The upper bound is clear, since a graph with n vertices has at most $\binom{n}{2}$ edges. We later show for $k = 4$, this bound is tight, but first we will look at the lower bound of the theorem. We start by showing a construction that connects n points with $ST^k(n)$ edges to a k-triangulation.

Lemma 4.1.2. *For every set of points $P = \{p_1, \ldots, p_n\}$ of n points in general position in $\mathbb{R}^{k-1}$ with $n > k$ and exactly k points of P on the convex hull of P, there exists a k-triangulation with vertex set P and $ST^k(n)$ edges.*

Proof. We reorder the points $p_1, \ldots, p_n$, such that $\tilde{p}_1, \ldots, \tilde{p}_k$ are the points $S_\triangle$ that span the convex hull and $\tilde{p}_{k+1}, \ldots, \tilde{p}_n$ are all inside this convex hull. We start with the first $k + 1$ vertices that form a k-triangulation $S_{k+1} = \binom{\{\tilde{p}_1, \ldots, \tilde{p}_{k+1}\}}{k}$. We build a k-triangulation by using a

recursice construction. For the step $i \in [k+2, n]$, S_{i-1} are the simplices of the k-triangulation with the points $\tilde{p}_1, \ldots, \tilde{p}_{i-1}$. We find a simplex $\mathcal{K} \in S_{i-1} \setminus S_\triangle$, such that $\tilde{p}_i \in \text{conv}^\star(\mathcal{K})$. The simplex $\mathcal{K}$ can be subdivided into k simplices $\binom{K \cup \tilde{p}_i}{k} \setminus \mathcal{K}$. We found a new k-triangulation $S_i = (S_{i-1} \cup \binom{K \cup \{\tilde{p}_i\}}{k}) \setminus \mathcal{K}$. Every $k{-}1$-face of $\mathcal{K}$ is now again part of two simplices. The one it was before besides $\mathcal{K}$ and the new one with $\tilde{p}_i$. In addition, the new $k{-}1$-faces which contain $\tilde{p}_i$ are also part of two k-simplices because the new simplices $\binom{K \cup \{\tilde{p}_i\}}{k}$ share pairwise $k{-}1$ vertices.
After the last step where we have added S_n, we have added all vertices to S_n and S_n is a k-triangulation of the points $p_1, \ldots, p_n$. We now count the number of edges. We start with $\binom{k}{2}$ edges and in every step from $i \in [k+1, n]$ we add k edges to the k-triangulation:
$k \cdot (n-k) + \binom{k}{2} = k \cdot n - k^2 + \binom{k}{2} = k \cdot n - \frac{2k^2 - k(k-1)}{2} = k \cdot n - \frac{k(k+1)}{2} = k \cdot n - \binom{k+1}{2} = ST^k(n)$. □

From the way we construct the k-triangulation in the proof of Lemma 4.1.2, we can conclude that the k-triangulation is a simple k-tree because the construction is the same as for a simple k-tree with n vertices and the construction order is $p_1, \ldots, p_n$. This means we can conclude:

Corollary 4.2. *A minimal k-triangulation has at most $ST^k(n)$ edges.*

We can conclude this corollary, because the number of edges of a simple k-tree with n vertices is $ST^k(n)$.

To show the lower bound is tight we need to proof that we cannot find a k-triangulation with less than $ST^k(n)$ edges. This statement is part of the following theorem:

Theorem 4.3. *The following statements for a graph $G = (V, E)$ with $|V| = n$ are equivalent:*

(I) G is a steady k-triangulation and has $ST^k(n)$ edges.

(II) G is a minimal steady k-triangulation

(III) G has a $k{-}1$ surface regular k-canonical ordering

Before we prove Theorem 4.3, we will first show a lemma to simplify the proof of Theorem 4.3.

Lemma 4.1.3. *Every vertex on the surface of an inner k-triangulation has at least $k{-}1$ edges to other vertices on the surface.*

Proof. The surface of an inner k-triangulation is, by the definition, a $k{-}1$-triangulation. Because of Lemma 3.3.1, we know that each vertex of a $k{-}1$-triangulation has at least $k{-}1$ edges. □

The following proof shows Theorem 4.3:

Proof. For any steady k-triangulation, we can find a k-canonical ordering $v_1, \ldots, v_n$. Because of Lemma 4.1.3, we know that in each step $i \in [k+1, n]$ of our decomposition of the k-triangulation we remove at least $k{-}1$ edges to vertices with lower index. The other edges are to vertices with higher index than v_i that are after this step new vertices on the surface of inner k-triangulation. For a k-canonical ordering $v_1, \ldots, v_n$ we denote $\mathcal{T}_i^k = \mathcal{T}^k[\{v_1, \ldots, v_i\}]$ as the inner k-triangulation in step i, we can count the number of edges of $\mathcal{T}^k$:

$$\underbrace{\sum_{i=k+1}^{n} |N^O_{\mathcal{T}_i^k}(v_i)|}_{(1)} + \underbrace{(n-k)}_{(2)} + \underbrace{\binom{k}{2}}_{(3)} \geq \underbrace{\left(\sum_{i=k+1}^{n} k-1\right) + (n-k) + \binom{k}{2}}_{(4)}$$

$$= (k-1)\cdot(n-k) + (n-k) + \binom{k}{2} = k\cdot(n-k) + \binom{k}{2} = ST^k(n)$$

(1) edges from v_i to the surface

(2) edges that brought $v_k, \ldots, v_{n-1}$ to the surface

(3) edges of the k-simplex $\{v_1, \ldots, v_k\}$

(4) edges of a k-triangulation with a $k-1$-surface regular k-canonical ordering

The two sides of the estimation are equal, if in each step $|N^O_{\mathcal{T}_i^k}(v_i)| = k-1$ holds. As a result in this case the k-canonical ordering is $k-1$-surface regular. Additionally, because of Lemma 4.1.3 and Lemma 3.3.1 we know in addition the number of edges is minimal. Equation (4) describes the number of edges of a k-triangulation with a $k-1$-surface regular k-canonical ordering and the number of edges is equal to $ST^k(n)$. □

We now show that the upper bound for the number of edges of a k-triangulation is tight. We will show this upper bound only for $k = 4$.

Theorem 4.4. *For every $n > 4$ there exists a 4-triangulation with n vertices and $\binom{n}{2}$ edges.*

We only have to show that we can construct a tetrahedization, which represents a K_n since a fully connected graph has the maximum number of edges for a graph with n vertices and $|E(K_n)| = \binom{n}{2}$.
In the following, we define the moment curve and a cyclic polytope the same way as described in the book `Triangulations. Structures for algorithms and applications` [18].

Definition 4.5 (Moment Curve)**.** *We define the moment curve as the function* $\mathrm{mom}_d(t) : \mathbb{R} \to \mathbb{R}^d, t \mapsto \{t, t^2, \ldots, t^d\}$.

Definition 4.6 (Cyclic Polytope)**.** *For a set of real numbers $R = r_1, \ldots, r_n$, we define the cyclic polytope $C(n,d)$ as the convex hull of the points $\{\mathrm{mom}_d(r_1), \ldots, \mathrm{mom}_d(r_n)\}$.*

Figure 4.2 illustrates a cyclic polytope with 18 vertices, but from the illustrated angle not all vertices are visible.
To show that we can create a tetrahedization, with $n \geq 5$ vertices we will calculate the coordinates in $\mathbb{R}^3$, but first we will show why a cyclic polytope with n vertices forms a K_n and induces an inner tetrahedization.

Lemma 4.1.4. *For $k = 4$, the cyclic polytope $C(n,3)$ with $n > 4$ is a K_n and in addition an inner k-triangulation.*

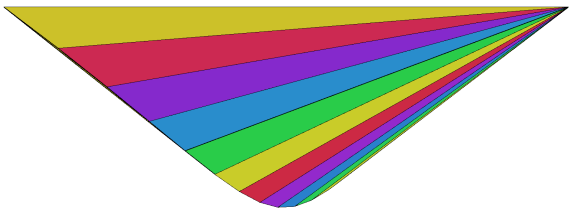


Figure 4.2.: A cyclic polytope with 18 vertices.

Proof. For the sorted numbers $R = \{r_1, \ldots, r_n\} \subset \mathbb{R} : r_1 < r_2 < \cdots < r_n$ we will show how the cyclic polytope on these numbers forms K_n and a inner k-triangulation by induction. We write $P = \{p_1 = \text{mom}_3(r_1), \ldots, p_n = \text{mom}_3(r_n)\}$ as vertices of the cyclic polytope. We start our induction with $\{p_1, p_2, p_3, p_4\}$ as a tetrahedron.
The vertices $\{p_1, p_2, p_4\}$ and $\{p_2, p_3, p_4\}$ form two triangles on the surface of the tetrahedron. Since r_5 is larger than any of the previous numbers r_1, r_2, r_3 and r_4, the point p_5 is outside the convex hull of the tetrahedron. In addition, the corresponding line segments from p_5 to p_1, p_2, p_3, p_4 do not intersect with the convex hull of the tetrahedron. Therefore, we can connect p_5 with the two triangles of the surface of the inner tetrahedization and form a new inner tetrahedization $\mathcal{T}_5^4$.
We can now use an inductive argument for $i \in [6, n]$ to show that each additional point is also outside the convex hull of $\mathcal{T}_{i-1}^4$. The inner tetrahedization defines the triangles $T_1, \ldots, T_{i-3}$ as $T_x = \{p_x, p_{x+1}, p_{i-1}\} : x \in [1, i-3]$ and the corresponding line segments to p_i do not intersect with the convex hull of $\mathcal{T}_{i-1}^4$. The line segments cannot intersect with the convex hull of $\mathcal{T}_{i-1}^4$ since $r_i > r_1, \ldots, r_{i-1}$ and the properties of the moment curve. As a result, we add $i-1$ edges and $i-3$ tetrahedron in each step. In every step, we thereby embed K_i in $\mathbb{R}^3$.

We now summarize this induction again and give an argument why the cyclic polytope has a canonical ordering. The construction order in the this induction is the reverse of a 4-canonical ordering, since every vertex we add in the construction is a chord-free vertex.
When we add a vertex, we do not change any of the vertices, edges, triangles, and tetrahedrons of the $\mathcal{T}_{i-1}^4$, but only cover triangles on the surface of the polytope. The triangles get a new second tetrahedron and are now inside the inner tetrahedization. The new triangles $\tilde{T}_y = \{p_y, p_{i-1}, 4_i\} : y \in [2, i-3]$ are also triangles inside $\mathcal{T}_i^4$ and are part of tetrahedron $\{p_{y-1}, p_y, p_{i-1}, p_i\}$ and $\{p_y, p_{y+1}, p_{i-1}, p_i\}$. All other triangles are part of the surface of the new inner tetrahedization $\mathcal{T}_i^4$. □

Since we now know, we can embed a K_n into an inner tetrahedization, we want to know how we can embed the K_n into a tetrahedization to get a tetrahedron as a convex hull. We will embed a K_n. For $n \geq 5$ into the tetrahedron $(-1, -1, 1)^\intercal, (-1, 1, -1)^\intercal, (1, -1, -1)^\intercal, (1, 1, 1)^\intercal$. Be aware of that $\text{mom}_3(-1) = (-1, 1, -1)^\intercal$ and $\text{mom}_3(1) = (1, 1, 1)^\intercal$, we will use for the following construction only points on the moment curve between $[-1, 1] \subset \mathbb{R}$.

Theorem 4.7. *The graph K_n can be embedded into $\mathbb{R}^3$ such that the embedding is a tetrahedization.*

Lemma 4.1.5. *If a continuoes function f has no root in the interval (x, y), then for all $v \in (x, y)$: $f(v) < 0$ or $f(v) > 0$.*

Proof. This is a direct implication of the intermediate value theorem, since the sign of the values of a function only changes in the root of a function. □

Proof. We use the points in $P = \{p_1, \ldots, p_n\}$ as the vertices of the tetrahedization. We define $map(v, a, b, x, y) := \frac{y-x}{b-a}(v - a) + x$ to map $v \in [a, b]$ to the interval $[x, y]$.

$$p_i = \begin{cases} (-1,-1,1)^\intercal & i = 1 \\ (1,-1,-1)^\intercal & i = 2 \\ (-1,1,-1)^\intercal & i = 3 \\ (1,1,1)^\intercal & i = n \\ \mathrm{mom}_3(map(i,3,n,-1,1)) & \textbf{otherwise} \end{cases}$$

For $n = 5$, the point $p_4 = mom_d(r_4)$ is described by $r_4 = map(4, 3, 5, -1, 1) = (\frac{1--1}{5-3}(4 - 3) + -1) = 1 - 1 = 0 \Rightarrow p_4 = (0, 0, 0)^\intercal$. Obviously, p_4 is inside the outer tetrahedron. We can connect the vertex with the outer tetrahedron to K_5. Now we look for the other cases where $n > 5$.
The points $p_3, \ldots, p_n$ form a cyclic polytope $C(n-2, 3)$ and therefore embed K_{n-2} in an inner tetrahedization as shown in Lemma 4.1.4. We now show that every point $p_4, \ldots, p_{n-1}$ is inside the outer tetrahedron p_1, p_2, p_3, p_n.

Our outer simplex spans 4 planes, we now have to argue, that all points on the moment curve in the interval $(-1, 1)$ are on the inside of the tetrahedron and therefore on the same side of the plane as $(0, 0, 0)^\intercal$:

$\mathsf{E}_{p_1,p_2,p_3} = \{(x, y, z)^\intercal \mid -x - y - z = 1\}$:
We have to show $\forall x \in (-1, 1) : f_{\{p_1,p_2,p_3\}}(x) = -x - x^2 - x^3 < 1$.
The polynomial $f_{p_1,p_2,p_3}(x) - 1$ has only -1 as its real root.

$\mathsf{E}_{p_1,p_2,p_n} = \{(x, y, z)^\intercal \mid x - y + z = 1\}$:
We have to show $\forall x \in (-1, 1) : f_{\{p_1,p_2,p_n\}}(x) = x - x^2 + x^3 < 1$.
The polynomial $f_{p_1,p_2,p_n}(x) - 1$ has only 1 as its real root.

$\mathsf{E}_{p_1,p_3,p_n} = \{(x, y, z)^\intercal \mid -x + y + z = 1\}$:
We have to show $\forall x \in (-1, 1) : f_{\{p_1,p_2,p_n\}}(x) = -x + x^2 + x^3 < 1$.
The polynomial $f_{p_1,p_3,p_n}(x) - 1$ has only -1 and 1 as its root.

$\mathsf{E}_{p_2,p_3,p_n} = \{(x, y, z)^\intercal \mid x + y - z = 1\}$:
We have to show $\forall x \in (-1, 1) : f_{\{p_2,p_3,p_n\}}(x) = x + x^2 + x^3 < 1$.
The polynomial $f_{p_2,p_3,p_n}(x) - 1$ has only -1 and 1 as its root.

To show that the moment curves in the interval $(-1, 1)$ lays inside the tetrahedron, we show that all points of the moment curve lay on the same side of each plain than the point $(0, 0, 0)^\intercal$. In all four cases we know because of Lemma 4.1.5 $f_P(x) - 1 < 0 : P \in \binom{\{p_1,p_2,p_3,p_4\}}{3}$ that

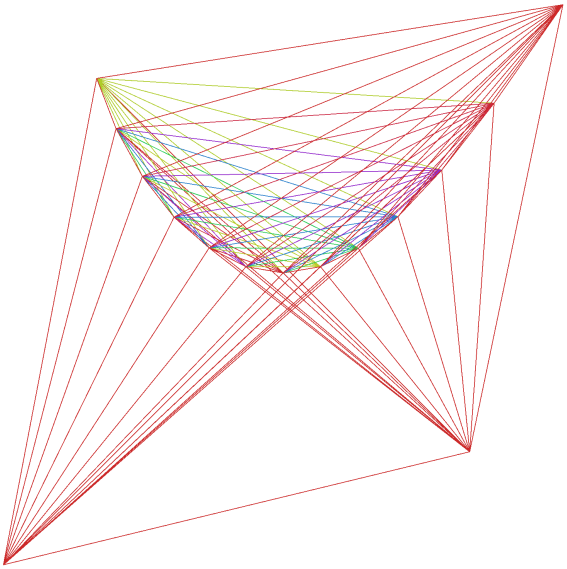


Figure 4.3.: A K_{15} embedded into a 4-triangulation

$f_P(0) - 1 = -1 < 0 : 0 = x \in (-1, 1)$ therefore $f_P(v) < 1 : \forall v \in (-1, 1)$ and therefor the moment curve $\text{mom}_3(x)$ is inside the convex hull of outer simplex for $x \in [1, 1]$.

Now that we know all points inside $[-1, 1]$ of the moment curve are inside the convex hull of the outer tetrahedron, we show that we can now connect p_1 and p_2 two all vertices and the resulting simplicial complex is a tetrahedization. When we connect p_2 pairwise to $p_3, \ldots, p_n$, we know that $p_3, \ldots, p_n$ form a cyclic polytope and every point spans the convex hull. We have choosen p_1 and p_2 such that both of them see half of all surfaces of the cyclic polytope. We now connect first p_2 and after then p_1.
The triangles $T_z = \{p_3, p_z, p_{z+1}, p_n\} : z \in [4, n-1]$ all get connected to p_2 and form a new inner tetrahedization, similar to the way we expand a cyclic polytope. But now, p_1 has in addition to the triangles $T_a = \{p_b, p_{b+1}, p_n\} : b \in [3, n-2]$ the triangles $T_b = \{p_2, p_3, p_a\} : a \in [4, n]$ that are visible. These triangles connected to tetrahedrons with p_1. As a result, the convex hull is a tetrahedron with the vertices p_1, p_2, p_3, p_n. When we add the simplex $\{p_1, p_2, p_3, p_n\}$ to the inner triangulation, the inner triangulation becomes a tetrahedization, since every triangle is part of two tetrahedrons. □

Remark 4.8. *The 4-canonical ordering of this tetrahedization of a K_n is given by $p_1, \ldots, p_n$. For the cyclic polytope, we already know that $p_3, \ldots, p_n$ is a 4-canonical ordering. So we only have to argue for every $p_i : i \in [4, n]$ that the tetrahedrons $\{p_1, p_2, p_{i-1}, p_i\}$, $\{p_1, p_3, p_{i-1}, p_i\}$ and $\{p_2, p_3, p_{i-1}, p_i\}$ contain no chord triangle. Since p_i has only vertices on the surface [18, p. 86] of the inner tetrahedization, it cannot be part of a chord triangle. After removing $p_n, \ldots, p_5$, the only tetrahedron left is $\{p_1, p_2, p_3, p_4\}$ and we can easily find a canonical order for the tetrahedron.*

For $k = 4$, we have shown that the maximal number of edges of a k-triangulation is $\binom{n}{2}$. Figure 4.3 illustrates how a K_{15} can be embedded into a 4-triangulation. But we assume for

$k > 4$, that it is possible to construct a similar k-triangulation with a cyclic polytope $C(n, k-1)$ as shown in the proof of Theorem 4.7. That is why we state the following conjecture:

Conjecture 4.9. *For $k > 4$, we suspect the upper bound of Theorem 4.1 is tight.*

Now that we know and suspect the bounds of the number of edges of a k-triangulation, we want to know how many k-simplices are inside a k-triangulation. We first look at the known case for $k = 3$.

Lemma 4.1.6. *The number of triangles of a triangulation is $2n - 4$.*

Proof. For a triangulation $\mathcal{T}^3$ with n vertices, m edges and f triangles, every edge is part of two triangles and every triangle has three edges. By the argument of double counting $2m = |\{\{e, t\} \mid e = \{v, w\} \subset t \in f_3(\mathcal{T}^3)\}| = 3f \Rightarrow m = \frac{3}{2}f$ We can put this inside Euler's formula $n - m + f = 2$.

$$n - \frac{3}{2}f + f = n - \frac{1}{2}f = 2 \Rightarrow n - 2 = \frac{1}{2}f \Rightarrow f = 2n - 4$$

□

For $k = 4$, we can show a similar result, but be aware of the fact that for $k = 3$ the number of edges was determined, by the number of vertices. For $k = 4$, we do not know the number of edges for a given number of vertices. As a result, the number of tetrahedrons has to be dependent on the number of vertices and edges.

Lemma 4.1.7. *A tetrahedization has $m - n$ tetrahedrons.*

Proof. First we prove that every vertex v in a tetrahedization is surrounded by $\deg(v)$ tetrahedrons.

We draw a small sphere around the vertex v such that no other vertex is inside the sphere and the edges intersect with the sphere. The triangles around v induce with the n intersection points a 3-triangulation on the surface on the sphere. Because of Lemma 4.1.6 we know that this triangluation has $2n - 4$ triangles. The triangles correspond to the tetrahedrons around v, as a result the number of tetrahedrons around v is the same number as triangles on the sphere. We can conclude further that every vertex v in a tetrahedization is part of $2 \cdot \deg(v) - 4$ tetrahedrons, since the number of vertices of the triangulation on the sphere is the number of edges of v.

We shall prove the statement again by the argument of double counting. A tetrahedron has four vertices and a vertex is part of $2 \cdot \deg(v) - 4$ tetrahedrons., therefore:

$$4t = |\{\{v, t\} \mid v \in t \in f_4(\mathcal{T}^k)\}| = \sum_{v \in V}(2 \cdot \deg(v) - 4) = 2 \cdot \sum_{v \in V} \deg(v) - 4 \cdot n \overset{(\star)}{=} 2 \cdot 2m - 4n$$

$$\Rightarrow t = m - n$$

($\star$) handshake lemma: $2m = \sum_{v \in V} \deg(v)$

□

Corollary 4.10. *We can conclude for n vertices, m edges, f triangles and t tetrahedrons that because of Euler's formula $n - m + f - t = 0$, that $n - m + f - (m - n) = 0$ and therefore $f = 2(m - n) = 2t$*

4.2. Minimal k-Triangulations are Simple k-Trees

In this section we show that minimal k-triangulations are simple k-trees under the Conjecture 3.24. We discuss later the implications of this equivalence.

Theorem 4.11. *We assume that the Conjecture 3.24 holds and simple k-trees are steady k-triangulations, the following statements for $k > 3$ about a graph $G = (V, E)$ with n vertices are equivalent:*

(I) G is a steady k-triangulation and has $kn - \binom{k+1}{2}$ edges.

(II) G has a $k-1$ surface regular k-canonical ordering

(III) G has a Schnyder realizer with k colors

(IV) G is a simple k-tree

Proof. (I) $\Rightarrow$ (II)
We have already shown in Theorem 4.3, that (I) and (II) are equivalent.

(II) $\Rightarrow$ (IV)
To proof that a k-triangulation with a $k-1$-surface regular canonical ordering $v_1, \ldots, v_n$ is a simple k-tree, we will build a k-color tree which represents the k-triangulation.

To construct the k-color tree in the k-canonical ordering, we skip the first k vertices, to start with the inner k-triangulation as the k-simplex $\{v_1, \ldots, v_k\}$. Then we start with v_k as the current root of the k-color tree.

We look at v_i as the next vertex of the ordering. The current inner k-triangulation is called $\mathcal{T}_i^k$ and v_i is connected to $k-1$ vertices $\tilde{v}_1, \ldots, \tilde{v}_{k-1}$ on the surface of the inner k-triangulation. The vertex $\tilde{v}_m : m \in [1, k-1]$ is the vertex with the highest position in the k-canonical ordering.

We have to look at a few nested cases, in which we assign one of the colors $c_2, \ldots, c_k$ to each of the $k-1$ surface neighbors, in such a way that we can put them into a k-color vertex set for v_i. In a later step, we will fill in the vertex of color c_1 in $q(v_i)$.
We now look at the possible cases:

(1) v_i covers vertices on the surface of $\mathcal{T}_{i-1}^k$
We need to look at the vertices that are covered by v_i. We look at the vertex v_l, which is lowest in the canonical ordering. We copy the k-vertex color set of v_l and assign it to v_i. In the k-color tree, we attache v_l below v_i as a child with the color c_1 and add v_i as the vertex of color c_1 to the k-vertex color sets of all the covered vertices.

(1.1) v_l was root of CT^k
we are done with v_i.

(1.2) v_l was not the root of the CT^k
Since v_l was not the root of the CT^k, we remember the parent of v_l as v_p and the color of the connecting edge as c_p. We remove v_l as the child of c_p.

Why are the changes to v_l valid? All vertices that are covered by v_i below v_l have no edge of color c_1 in the path from v_i to v_l in the k-color tree. As a result, v_i is a neighbor of each of these vertices and is part of their k-vertex color set at position 1.

(2) v_i does not cover vertices on the surface of $\mathcal{T}_{i-1}^k$
We have to find out, which vertex is our parent and what color the parent has. Since v_i does not cover any vertex on the surface, v_i connects directly to a k−1-face of a k-simplex. As a result, we find the neighbor of v_i called v_p with the highest number in the canonical ordering. The vertex v_p has a k-vertex color set, and we copy it for v_i. The vertex v_p has a neighbor on the surface, let us call it x, not in common with v_i, since we only connected v_i to a k−1-face of the simplex. We replace x with v_p in the k-vertex color set and therefore implicitly assign a color to v_p, called c_p.

In all of these cases, we have found a parent v_p of v_i, except for the case where v_i is the new root of the CT^k. We set v_i as the c_p colored child of v_p. The resulting k-color tree is valid, since when the parent previously had a vertex with the color c_p, we moved it under v_i, so that every vertex of the CT^k does not have two children of the same color.

We do this construction until we have added v_{n-1}. Since v_n is part of the outer simplex and therefore not part of the k-color tree. To finish our mapping and show that the k-color tree has the properties of a k-color tree that encodes the original k-triangulation.
For v_n, we have to do the same as in the previous cases, except that we do not need to change the k-color tree. The vertex v_n covers all inner vertices that are currently on the surface of the inner k-triangulation. This means v_n gets placed at position 1 of all k-vertex color sets of the inner vertices on the surface of $\mathcal{T}_{n-1}^k$.

We now check the properties of Lemma 3.5.4 and that they align with the structure of k-triangulation and the build k-color tree. Let us recall the properties:

Lemma 3.5.4. *For a vertex set $V = \{v_1, \ldots, v_n\}$, $V_\triangle = \{v_1, \ldots, v_k\}$, and $V_\swarrow = \{v_{k+1}, \ldots, v_n\}$, we know a k-color tree CT^k and a map q that maps from $V_\swarrow$ to k-vertex color sets are equivalent to a simple k-tree if they hold the following properties:*

1. $V_\triangle \cap V(\mathsf{CT}^k) = \emptyset$ *and* $q(v_{k+1}) = (v_1, \ldots, v_k)$
2. $\forall e = (v, w) \in E(\mathsf{CT}^k)$*:*
   a) e *has color* $c_{q(v)}(w)$
   b) $w = v_{q(v)}(i_{q(v)}(w))$
   c) $|e(q(v)) \cap e(q(w))| = k$−1

We start with property 1 The first part $V_\triangle \cap V(\mathsf{CT}^k) = \emptyset$ is fullfilled. The vertices $v_1, \ldots, v_{k-1}, v_n$ for the outer simplex are not part of the k-color tree. For the second part of property 1, we look at the root v_r of the CT^k. Note that in the current construction, we have another encoding than in Lemma 3.5.4. We now have to show $q(v_r) = (v_n, v_1, \ldots, v_{k-1})$.
The vertex v_n gets set at the first position of $q(v_r)$ in the last step, since the root of the CT^k is always a vertex on the surface of $\mathcal{T}_k^k, \ldots, \mathcal{T}_{n-1}^k$.

For the other vertices in $q(v_r)$ we look at the case (1.1) of the k-color tree construction. At the beginning of the construction, v_k is the root of the k-color tree and is initalized with the k-vertex color set $q(v_k) = (\bot, v_2, \ldots, v_{k-1})$. This k-vertex color set does not change until another vertex replaces v_k as the root of the k-color tree. The new root v_p gets the same k-vertex color set as v_k and after this step, we alter $q(v_k)$ to $(v_p, v_1, \ldots, v_{k-1})$. This behavior of the k-vertex color set for the root does not change until we have added v_{n-1}. As a result, the other vertices in the k-vertex color set of v_r are the other outer simplices. We conclude after the construction: $q(v_r) = (v_n, v_1, \ldots, v_{k-1})$.

Now we have to assure that property 2 of Lemma 3.5.4 for the other vertices holds. We have to show, that for each edge $e = (v, w) \in E(\mathsf{CT}^k)$ the properties a), b) and c) hold. We now look at two cases:

e has color c_1
: If e has color c_1, at some point in the construction w removed v from the surface. As a result at position 1 of the k-vertex color set of v is w and therefore $c_{q(v)}(w) = c_1$. In addition, property b) holds. Because of the way, we calculated $q(w)$ from $q(v)$ we only set the first position different, $|e(q(v)) \cap e(q(w))| = k - 1$.

e does not have the color c_1
: In this case, we either got w and the color of the edge of the previous child of w (case 1.2 of the construction) or v got connected to an existing k–1-face of a simplex (case 2 of the construction). In the second case, we found w as the vertex of the simplex with the highest position in the canonical ordering and the color by the only non-common neighbor of v and w. In both cases we placed w at the corresponding position of the color c in the vertex set of v (this shows a) and b)).
All the other vertices are eighter directly derived from w, or can be identified because of the previous children that got connected to the same k–1-face of the simplex where w was the vertex with the highest position in the canonical ordering. As a result, if v and w get covered, they have the covering vertex in common at position 1 of the k-vertex color set. In the case that they do not get covered by an inner vertex, they get covered by v_n in the last step of the construction. Regardless, they hold property c), since they only differentiate in the position $i_{q(v)}(w)$.

(IV) $\Rightarrow$ (III)
We know because of Lemma 3.5.3 that a simple k-tree is a k-triangulation and that we can find a corresponding space-partitioning drawing. Now we need to show that we can find for such a space-partitioning drawing a k-color realizer. Note, a space-partitioning drawing constructed by the proof of Lemma 3.5.3 already fulfills the property 1 of a k-color realizer($\chi(v_1, \ldots, v_k) = 1$). To find a k-color realizer, we need to color and direct the inner edges of the k-tree. We do this in the construction order of the vertices and add step by step the vertices by there position defined in the space partitioning drawing. In addition, we will use the corresponding k-color tree CT^k of the simple k-tree. According to the CT^k w use the map q from $V(\mathsf{CT}^k)$ to the k-vertex color sets.
Before we can start with the recursion we have to color and direct the edges of v_{k+1}. The vertex

v_{k+1} is the root of the k-color tree CT^k and therefor connected to the vertices $v_1, \ldots, v_k$. We direct these edges as outgoing edges and color them with the colors $c_1, \ldots, c_k$, respectively.
For a simple k-tree of size $k+1$ we realize that this is a valid Schnyder realizer with k colors.
Property 2 is met, since the only inner vertex v_{k+1} has exactly k outgoing edges of different colors to each of the vertices of the outer simplex.
Property 3 is met, since for all inner vertices there are no incoming edges and for the outgoing edges $e_1 = \{v_{k+1}, w_1\}, \ldots, e_k = \{v_{k+1}, w_k\}$ of colors $c_1, \ldots, c_k$ respectively have the property: $\chi_{v_{k+1}}(e_1, \ldots, e_k) = \chi(w_1 - v_{K+1}, \ldots, w_k - v_{k+1}) = \chi(w_1, \ldots, w_k) = 1$.
Now we start to add vertices recursively. When we add the inner vertex v_i, we add him to the clique $C_i = q(v_i)$ and therefore inside th convex hull of the positions of the vertices of C_i. The k-vertex color set of a vertex contains exactly the vertices of C_i. We now look at the vertex c_m at position $m \in [k]$ of the k-vertex color set $q(v_i)$. We direct the edge from v_i to v_m with the color c_m.

For each vertex v_i we may add a new color to an outer vertex v_o. As a result in the k-color tree the path between v_i and v_{k+1} does not contain an edge of color $c_{q(v_i)}(v_o)$. We can conclude $v_o\ i_{q(v_i)}(v_o) = i_{Q(v_{k+1})}(v_o)$. Therefore, we colored the edge to v_o in the same color as the edge from v_{k+1} to v_o. Therefore, property 2 of a k-color realizer holds.
For all colored edges that do not get connect to an outer vertex we have to show that they fulfill property 3. First, since we only added k outgoing edges of all k colors to v_i All inner vertices have after the step where we added v_i still k colored and outgoing edges.
We call the vertices of the clique $C_i = (\tilde{v}_i, \ldots, \tilde{v}_k) = q(v_i)$ because of Lemma 3.4.2 for all $l \in [k]$: $\chi(\tilde{v}_1, \ldots, \tilde{v}_{l-1}, v_i, \tilde{v}_{l+1}, \ldots \tilde{v}_n) = \chi(\tilde{v}_1, \ldots, \tilde{v}_k)$. Since this holds also for v_k and the clique/outer simplex $v_1, \ldots v_k$ we can use an inductive argument, that this holds for all inner vertices. Since we get this property for v_i from its parent in the k-color tree CT^k and find it for all children in CT^k for v_i. We know $\chi(\tilde{v}_1, \ldots, \tilde{v}_n) = 1$ since $\chi(v_1, \ldots, v_k) = 1$ and of the previous inductive argument. Therefore, we can follow for the outgoing edges of v_i $\chi(\{v_i, \tilde{v}_1\}, \ldots, \{v_i, \tilde{v}_k\}) = \chi(\tilde{v}_1 - v_i, \ldots, \tilde{v}_k - v_i) = 1$. We now have to show that the outgoing edges that induce incoming edges on other vertices hold the second part of property 3. For an outgoing edge $e_l = \{v_i, w\}$ of the color c_l we look at w, e_l is an incoming edge of w. The vertex w has the outgoing edges $f_1, \ldots, f_k$, and we know $\chi(f_1, \ldots, f_k) = 1$. We have to show $\chi(f_1, \ldots, f_{l-1}, v_i, f_{l+1}, \ldots, f_k) = -1$.

The vertex f_l spans with the other vertices $f_1, \ldots, f_{l-1}, f_{l+1}, \ldots, f_k$ of the other outgoing edges of w a cone where $l \in [k]$ is the index of the base point of the cone:

$$C_l(f_1, \ldots, f_k) = \{w \mid \forall \lambda_1, \ldots, \lambda_{l-1}, \lambda_{l+1}, \ldots, \lambda_k \in R_{>0}, w = f_l + \sum_{l \neq i \in [k]} \lambda_i (f_i - f_l)\}$$

For a point $p \in C_l(f_1, \ldots, f_k)$ we know $p = f_l + \sum_{l \neq i \in [k]} \lambda_i (f_i - f_l)$, we can calculate λ_l to find the barycentric coordinates of p:

$$\begin{aligned} p &= f_l + \sum_{l \neq i \in [k]} \lambda_i (f_i - f_l) \\ &= f_l + (\sum_{l \neq i \in [k]} \lambda_i f_i) - (\sum_{l \neq i \in [k]} \lambda_i f_l) \\ &= \underbrace{(1 - \sum_{l \neq i \in [k]} \lambda_i)}_{\lambda_l} f_l + (\sum_{l \neq i \in [k]} \lambda_i f_i) \end{aligned}$$

Since p is outside of the simplex of $f_1, \ldots, f_k$ $\sum_{l \neq i \in [k]} \lambda_i > 1$ and therefore $\lambda_l < 0$ we can conclude
with Lemma 3.4.2 and Lemma 3.4.1 that $\chi_w(\{w, f_1\}, \ldots, \{w, f_{l-1}\}, \{v_i, w\}, \{w, f_{l+1}\}, \ldots, \{w, f_k\}) = -\chi(f_1, \ldots, f_k) = -1$.

This holds for all outgoing edges of v_i and by the inductive argument for all inner vertices.
(III) $\Rightarrow$ (I)
Given a Schnyder realizer with k colors $R_\triangle$ for a k-triangulation $\mathcal{T}^k$, we can separate the vertices in $V_\triangle$ and $V_{\perp}$. We can count the number of edges:

$$\underbrace{\sum_{v \in V_{\perp}} k}_{(1)} + \underbrace{\binom{k}{2}}_{(2)} = k \cdot (n-k) + \binom{k}{2} = kn - \binom{k+1}{2} = ST^k(n)$$

(1) the outgoing edges of the inner vertices

(2) the edges of the outer simplex

Be aware that we have counted every edge in $\mathcal{T}^k$, since the vertices of the outer simplex does not have outgoing edges. For inner vertices, we only count the outgoing edges for each vertex. As a result, we count every edge only once. □

The proof of Theorem 4.11 has a few implications. We only showed the theorem for $k > 3$, even though the minimal number of edges for a 3-triangulation is $ST^k(n)$. The reason for this restriction is based on a simple observation. For $k = 3$, a minimal and a maximal k-triangulation has the same number of edges, but not all triangulations are simple k-trees.
Figure 4.4 shows the embedding of an octahedron in the plane. The graph is a 3-triangulation, which is not a simple 3-tree. Every 3-triangulation has a 2-surface regular canonical ordering, but because we are in the plane, the conditions of minimal edge number are not strong enough to enforce that every triangulation is a simple 3-tree. We need an additional property for the triangulation, so we know it is a simple k-tree. Such a property is for example that the triangulation is chordal. The reason for that is the simple fact that for $k > 3$, we have to connect $k - 1$ edges to the surface, but they have to connect to a k–1-face of a simplex. For $k = 3$, in every step of the canonical ordering, v_i only has to be to two connected vertices on the border of the inner triangulation. In fact, the two vertices on the surface of the triangulation may not have any relation to each other. This brings us to our next corollary:

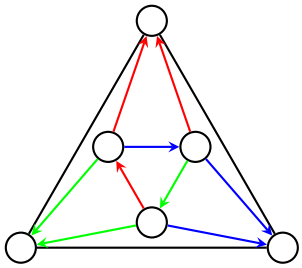


Figure 4.4.: the octahedron graph

Corollary 4.12. *For a* 3*-triangulation* $\mathcal{T}^3$ *and the corresponding canonical ordering* $v_1, \ldots, v_n$*, the vertex* $v_i : i \in [4, n]$ *connects to two vertices* u *and* w *on the surface of* $\mathcal{T}^k[\{v_1, \ldots, v_{i-1}\}]$*. If in each step, the vertices* u *and* w *are connected by an edge, then the* 3*-triangulation is a simple* 3*-tree.*

The most important implication of Theorem 4.11 is the fact that k-triangulations and simple k-trees are the same graph class. If we need a triangulation of higher dimension for our application and our application does not require a specific triangulation we always can use a simple k-tree as the triangulation. This is for example the case if, we need a triangulation of n points in $\mathbb{R}^d$. In such a use case, we can efficiently build a simple $d+1$-tree. This is particularly interesting, since simple k-trees are chordal. For chordal graphs we have efficient algorithms to solve a lot of graph problems [13, 16]. In addition, we can represent the triangulation as a k-color tree. A k-color tree can be used to recursively process the structure of the graph. This might help finding efficent algorithms for minimal triangulations like our construction of contact representations in chapter 5.

5. Contact Representation

For $k > 4$, k-simplices cannot be represented for humans, that is the reason why we will look in the following part at $k = 4$. For $k = 3$ we can find a triangle contact representation for a k-triangulation as described in the paper `On triangle contact graphs` [6]. When we look at $k = 4$, we have previously shown that a k-triangulation has a variable number of edges for $k > 3$, but only the minimal k-triangulations have a k-color realizer. That is the reason, why we will only show show contact representations in the following part for minimal k-triangulations. We use the facts of Theorem 4.11 and Construction 3.28 that minimal k-triangulations are simple k-trees and that we can transform them into k-color trees.

Definition 5.1 (Contact Representation)**.** *In a contact representation we use geometric objects to represent the vertices of a graph. The representation is for a specific dimension, typically the plane or the* 3*-dimensional space. The geometric objects that represent the graph have the same dimension as the representation. To represent the graph we arrange and form the geometric objects in such a way that the connected vertices and the corresponding objects touch, but do not cross each other. IN addition, three objects do not touch each other in a single point.*

Definition 5.2 (Side-Contact Representation)**.** *A side-contact representation is similar to a contact representation. The difference is that for two connected vertices the corresponding objects have to share an area in the representation. It is not sufficient that two objects touch each other in a single point. This means in a side-contact representation on the plane, two objects have to share a line segment and in the* 3*rd dimension they have to share a flat intersection area.*

We call a side-contact representation hole-free, if the geometric objects corresponding to the vertices subdivide the occupied space of the drawing without leaving any gaps.

In this chapter we will look at contact representations of simple 4-trees and in the next chapter we will look at side-contact representations. A contact representation is a drawing of a graph, we will use drawing in the following as a synonym for contact representation and side-contact representation. A drawing Γ is a mapping from vertices to geometric objects.

We will now introduce a notation we need for the rest of the thesis to construct contact representations. For a drawing Γ and for two vertices u, v we write $\mathsf{isc}_\Gamma(O_1, O_2), O_1 = \Gamma(u), O_2 = \Gamma(v)$ as the intersection point between the objects of vertex u and v.

5.1. General Approach

We describe in the following part the general approach on how we construct contact representations for simple k-trees. This construction can be used for all recursive constructions of representations of simple k-trees.

We start a construction by transforming the simple k-tree into a k-color tree. This transformation is always possible because of construction 3.28. Since $v_1, \ldots, v_k$ are not part of CT^k which represents the simple k-tree ST^k. The construction start by handling these outer vertices. This means for the constructions of the following sections, that we will look for a starting configuration for the outer vertices. In addition, we will describe for each construction an invariant so we only need to explain one step of the recursion.

With the starting configuration, the invariant and an arbitrary construction order of the k-tree, we will recursively construct the drawing. We will show that at each step we find a free spot in the drawing where we place a vertex v_i of CT^k and all children of v_i.

5.2. Triangle Contact Representation for Simple 3-Trees

Before we look at the tetrahedron contact representation of a simple 4-tree we will look into another way to construct a triangle contact representation for simple k-trees. This way of constructing triangle contact representations is not as powerful as the previous described method of `On triangle contact graphs` [6], since we can only calculate triangle contact representations only for simple 3-trees and not for all planar triangulations. Even though we have a stronger restriction to the graph we can represent, we use this as an easier motivating application for our results of chapter 3. This will help to understand the more complex construction of the tetrahedron contact representation of the simple 4-trees.

Theorem 5.3. *Every simple 3-tree can be represented as a triangle contact representation in the plane.*

Construction 5.4 (Triangle Contact Representation)**.**
We start with a simple 3-tree ST^3, because of construction 3.28 we know we can construct a 3-color tree CT^3 which represents ST^3. Our constructions start with three arbitrary triangles T_1, T_2, T_3 that are arranged in such a way that they touch each other and together enclose a free space of the shape of a triangle. We call this triangle $S = \{s_1, s_2, s_3\}$. We find another triangle S' such that each line segment of S' contains one vertex of S.
We call the vertices of $S' = \{s'_1, s'_2, s'_3\}$, the center of T_1, T_2, T_3 even though that s_i and $T_i : i \in [1, 3]$ have nothing in common with the centroid of those triangles. We use these new centers, to simulate a specific setup of surrounding triangles, to construct the contact representation, but we only need these points for the construction and can place T_1, T_2 and T_3 around the later constructed triangles of the inner vertices.

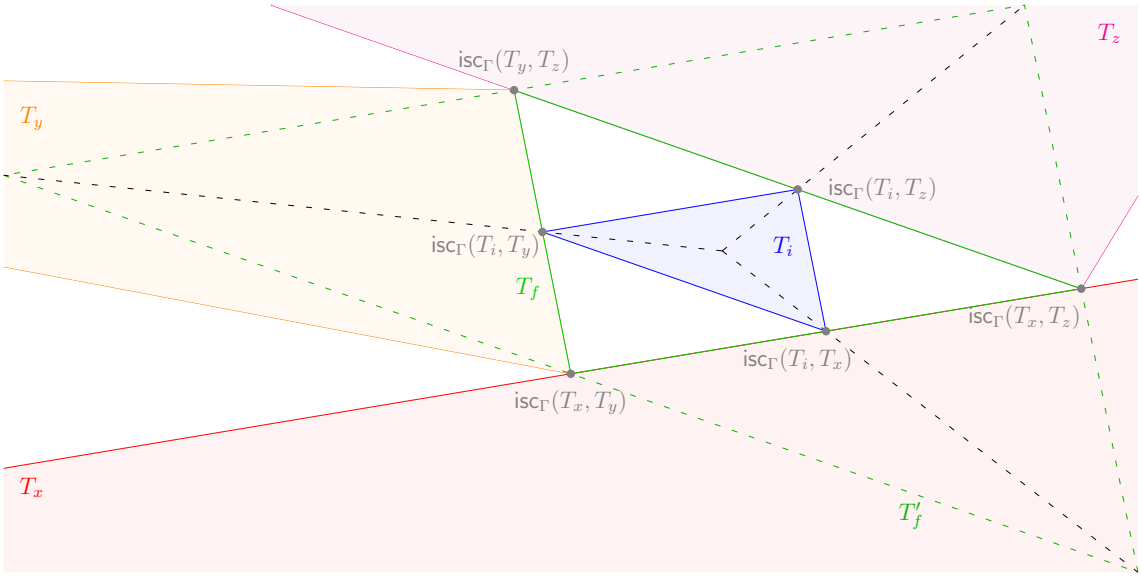


Figure 5.1.: invariant of construction 5.4

The three triangles T_1, T_2, T_3 belong to the corresponding vertices of the outer triangle v_1, v_2, v_3 respectively. The other vertices are sorted by the construction order of the k-tree. We now add the vertices in the construction order. For this we use the mapping q of our k-color tree. Be aware of the fact, that for $v_i : i > 3$ all vertices of $e(q(v_i))$ have a lower position in the construction order than v_i. Therefore, if we add v_i all vertices of $e(q(v_i))$ have already been added to the drawing and we know each triangle they represent.

Now we look at the step of how we add a vertex v_i to the drawing. We start by defining an invariant. We always find for each vertex v_i a triangle T_f in which we want to embed v_i and a triangle T'_f such that each corner of T_f intersect with on edge of T'_f. Figure 5.1 illustrates the invariant.
In the step where we add v_i the corresponding triangles T_o, T_p, T_q of $\{v_o, v_p, v_q\} = e(q(v_i))$ pairwise touch each other and enclose an unoccupied space in shape of a triangle T_f. The triangle $T_f = \{\mathsf{isc}_\Gamma(T_o, T_p), \mathsf{isc}_\Gamma(T_o, T_q), \mathsf{isc}_\Gamma(T_p, T_q)\}$ is created by the intersection points of the surrounding triangles. We now pick a point $p_i \in \mathrm{conv}^\star(T_f)$.

This point induces the shape of the triangle representing v_i. After we have chosen p_i, p_i and the centers of each surrounding triangle form a line. Each line intersects with a line segment of the surrounding triangle which gives us three intersection points S_o, S_p, S_q. These three intersection points form a new triangle T_i inside the triangle T_f. Additionally, since $P_i \in \mathrm{conv}^\star(T_f)$, T_i and each pair of the surrounding triangles form a new free triangle shaped space. We use these three triangles to add the possible children of v_i in CT^3.

We now prove that the Construction 5.4 generates a contact representation and therefor proves Theorem 5.3.

Proof. When we look at the invariant of the construction, it is clearly true that for T_1, T_2 and T_3 and the initial triangle S, since we have chosen a surrounding triangle S' with these properties.

Now we look at the step where we add v_i to the representation. We find T_f and T'_f for v_i and the 3-vertex color set $q(v_i)$ so that the corners of T'_f are the centers of the corresponding triangles T_x, T_y, T_z of the vertices $\{x, y, z\} = e(q(v_i))$. We denote the center of a triangle T as $c(T)$. The triangle $T'_f = \{c(T_x), c(T_y), c(T_z)\}$. Since the $\mathsf{isc}_\Gamma(T_i, T_k) : k \in \{x, y, z\}$ is on the line segment between $c(T_i)$ and $c(T_k)$ we find six new Triangles:

- $T_o = \{\mathsf{isc}_\Gamma(T_x, T_y), \mathsf{isc}_\Gamma(T_i, T_x), \mathsf{isc}_\Gamma(T_i, T_y)\}$, $T'_o = \{c(T_x), c(T_y), c(T_i)\}$
- $T_p = \{\mathsf{isc}_\Gamma(T_x, T_z), \mathsf{isc}_\Gamma(T_i, T_x), \mathsf{isc}_\Gamma(T_i, T_z)\}$, $T'_p = \{c(T_x), c(T_z), c(T_i)\}$
- $T_q = \{\mathsf{isc}_\Gamma(T_y, T_z), \mathsf{isc}_\Gamma(T_i, T_y), \mathsf{isc}_\Gamma(T_i, T_z)\}$, $T'_q = \{c(T_y), c(T_z), c(T_i)\}$

They form for the triangles for the children of v_i in CT^3 the triangles to embed the children. Be aware, if $S = \{s_1, s_2, s_3\}$ is a valid triangle in particular the three corners s_1, s_2, s_3 are not collinear all other constructed triangles are also not collinear. Since we choose $c(T_i)$ from $\mathrm{conv}^\star(T_f)$. For $d \in \{o, p, q\}$ the corners of T_d intersect respectively with one edge of the triangle T'_d. The children of v_i can find a triangle $\tilde{T}_f$ so it is surrounded by the triangles corresponding to the 3-vertex color set of the child, since T_i replaces one of the surrounding triangles T_z, T_y, T_x of T_f in T_o, T_p, T_q respectively. □

Remark 5.5. *The construction 5.4 is very unrestricted and in every step of the construction we could choose a different strategy to choose a point. This also applies to the three starting triangles. We can choose them as similar or completely different triangles. In particular, we can choose them in a way, such that the convex hull of the drawing is a triangle.*

5.3. Tetrahedron Contact Representation for Simple 4-Trees

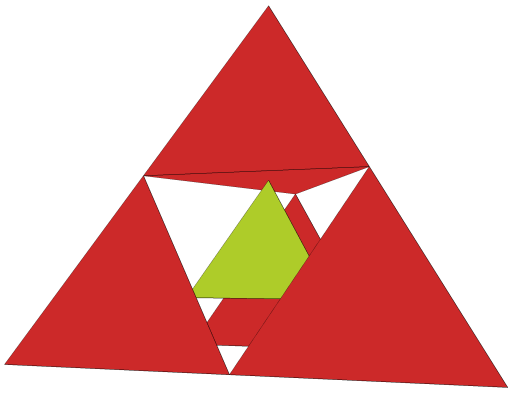
Now that we understand how we can construct a triangle contact representation for simple 3-trees we look at simple 4-trees. We will use a similar construction to find a tetrahedron contact representation for simple 4-trees. But we will use tetrahedrons and octahedrons in the invariant.

Theorem 5.6. *For every simple 4-tree exists a tetrahedron contact representation in $\mathbb{R}^3$.*

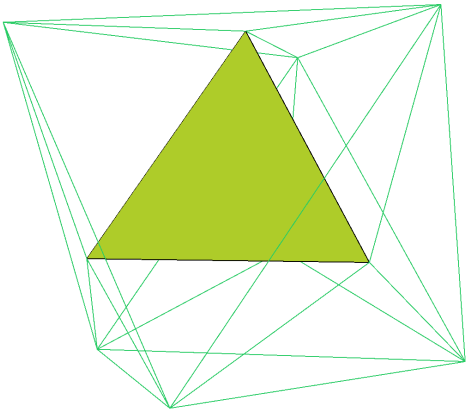
Construction 5.7 (Tetrahedron Contact Representation)**.**
We start with the simple 4-tree ST^4 and the corresponding 4-color tree CT^4 with the colors $C = \{c_1 = r, c_2 = g, c_3 = b, c_4 = l\}$. The colors are orderd: $r < g < b < l$. The construction of the representation Γ starts with one Tetrahedron T, that we subdivide in four tetrahedrons and one free octahedron.
We subdivide the edges of the tetrahedron T, we get $p_1, p_2, p_3, p_4, p_5, p_6$ that form an octahedron O inside T. We color the vertices of $T = \{p_r, p_g, p_b, p_l\}$ we can rename $p_1, p_2, p_3, p_4, p_5, p_6$ to $p_{rg}, p_{rb}, p_{rl}, p_{gb}, p_{gl}, p_{bl}$ in respect to the endpoints of the corresponding endpoints of the edge.

We find the tetrahedrons $T_r = \{p_r, p_{rg}, p_{rb}, p_{rl}\}$, $T_g = \{p_{rg}, p_g, p_{gb}, p_{gl}\}$, $T_b = \{p_{rb}, p_{gb}, p_b, p_{bl}\}$ and $T_l = \{p_{rl}, p_{gl}, p_{bl}, p_l\}$ respectively for the outer vertices v_1, v_2, v_3, v_4 of ST^4. The tetrahedrons T_r, T_g, T_b, T_l and O subdivide T:

$$\left(\mathrm{conv}(O) \cup \bigcup_{c \in C} \mathrm{conv}(T_c)\right) \subset \mathrm{conv}(T)$$



The tetrahedron T_i corresponding to v_i in yellow and the surrounding tetrahedrons $T_r^i, T_g^i, T_b^i, T_l^i$ in red.



The tetrahedron corresponding to v_i in yellow and the green lines form the outlines of the octahedrons O_o, O_p, O_q, O_r for the possible children of v_i

Figure 5.2.: invariant of construction 5.7

The octahedron O intersects pairwise with the tetrahedrons $T_c, c \in C$ on one common faces, we color these face in the color c in respect to T_c. We call these four colored triangles $\triangle_c$. In the octahedron we call the triangles in the opposite position of $\triangle_c$ $\triangle'_c$($\triangle'_c$ has no vertex in common with $\triangle_c$). For the construction below we remember the center of T_c as $c(T_c) = p_c : c \in C$.

We start the construction by adding v_i we find for v_i $q(v_i)$ and therefore the tetrahedrons $T_r^i, T_g^i, T_b^i, T_l^i$ that will touch the next generated tetrahedron T_i representing v_i. The centers of all $T_c^i : c \in C$ form a tetrahedron T_f. In addition, the tetrahedrons intersect pairwise and $p_{c_1c_2} = \mathsf{isc}_\Gamma(T_{c_1}^i, T_{c_2}^i) : c_1, c_2 \in C \wedge c_1 < c_2$ form a free octahedron O_f. This relation between T_f and O_f form the invariant for our construction. Figure 5.2 illustrates the invariant.

We will now choose a point $p_i \in \mathrm{conv}^\star(O_f)$. We now find the intersection point p_c^i of the line segment between p_i and $c(T_c^i)$ and the face of the tetrahedron T_c^i called $\triangle_c^i$. These intersection points describe the new tetrahedron $T_i = \{p_r^i, p_g^i, p_b^i, p_l^i\}$. We remember $c(T_i) = p_i$ as the center of T_i. We occupied a part of the octahedron O_f with T_i and the rest of the space is subdivided into four octahedrons O_o, O_p, O_qO_r by the points $(T_i \setminus p_c^i) \cup \triangle'_c$ for $c \in C$:

- $O_o = \{p_g^i, p_b^i, p_l^i, \mathsf{isc}_\Gamma(T_i, T_g^i), \mathsf{isc}_\Gamma(T_i, T_b^i), \mathsf{isc}_\Gamma(T_i, T_l^i)\}$ $T_o = \{c(T_g^i), c(T_b^i), c(T_l^i), c(T_i)\}$
- $O_p = \{p_r^i, p_b^i, p_l^i, \mathsf{isc}_\Gamma(T_i, T_r^i), \mathsf{isc}_\Gamma(T_i, T_b^i), \mathsf{isc}_\Gamma(T_i, T_l^i)\}$ $T_p = \{c(T_r^i), c(T_b^i), c(T_l^i), c(T_i)\}$
- $O_q = \{p_r^i, p_g^i, p_l^i, \mathsf{isc}_\Gamma(T_i, T_r^i), \mathsf{isc}_\Gamma(T_i, T_g^i), \mathsf{isc}_\Gamma(T_i, T_l^i)\}$ $T_q = \{c(T_r^i), c(T_g^i), c(T_l^i), c(T_i)\}$
- $O_r = \{p_r^i, p_g^i, p_b^i, \mathsf{isc}_\Gamma(T_i, T_r^i), \mathsf{isc}_\Gamma(T_i, T_g^i), \mathsf{isc}_\Gamma(T_i, T_b^i)\}$ $T_r = \{c(T_r^i), c(T_g^i), c(T_b^i), c(T_i)\}$

The octahedrons O_o, O_p, O_q, O_r are inside the tetrahedrons T_o, T_p, T_q, T_r respectively. And we can place the children of v_i in CT^k in a further step of the construction.

Proof. We look at the invariant of the construction. We find for every vertex v_i a tetrahedron T_f and an octahedron O_f. All vertices of O_f lay on the line segment of one edge of T_f as described in the invariant. Since we choose $p_i \in \text{conv}^\star(O_f)$, p_i is also in $\text{conv}^\star(T_f)$. Consider that O_f shares pairwise the triangle with $\triangle_c$ with $T_c^i : c \in C$. Therefore, $\text{conv}^\star(\triangle_c) \subset \text{conv}^\star(T_f)$ which gives us that, $\text{isc}_\Gamma(T_i, T_c^i)$ is inside $\text{conv}^\star(T_f)$. In addition, $c(T_i) = p_i$ is inside $\text{conv}^\star(T_i)$.

Assume not, then p_i would lay outside of the tetrahedron T_i. In this case p_i and $c(T_c^i)$ have to lay with $\text{isc}_\Gamma(T_i, T_c^i)$ on one line, which is only possible if $p_i \notin \text{conv}^\star(T_f)$, which is a contradiction.

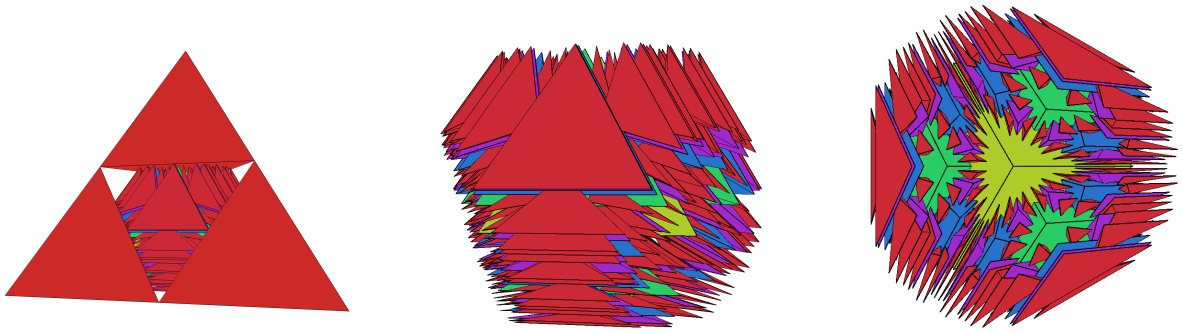
As a result $\text{conv}(T_i) \subset \text{conv}^\star(T_f)$ and the remaining space of O_f gets subdivided into O_o, O_p, O_q, O_r and T_o, T_p, T_q, T_r respectively. These tetrahedrons and octahedrons form the invariant for the children of v_i. With this invariant the construction is valid for all simple 4-trees, if the starting points of T are in general position, since then $\text{conv}^\star(T) \neq \emptyset$. □

Remark 5.8. *Like in Construction 5.4 we could use any configuration of four tetrahedrons that touch each other pairwise. For such a more general starting configuration we need to find a tetrahedron T which fulfills the invariant as T_f. When we have found such a tetrahedron T we could use the vertices of T as the centers of the tetrahedrons representing the outer simplex. That way we can use any starting configuration for the construction analogously to the way we have described it already in Construction 5.4 for simple 3-trees.*

Conjecture 5.9. *We conjecture it is possible to find similar contact representations for $k > 4$ in $\mathbb{R}^{k-1}$ with k-simplices as the geometric objects.*

We base this conjecture based on the fact of the existence of the k-color tree for a simple k-tree. The invariant for $k = 3$ and $k = 4$ can be more generalized, T_f for $k = 3$ and T_f for $k = 4$ are k-simplices. T_f' for $k = 3$ and O_f for $k = 4$ are the line graph of T_f in both cases. In this thesis, we do not prove that we can subdivide a k-simplex into another k-simplex and k geometric objects of the line graphs of a k-simplex, but if it is possible the conjecture holds.

Now we show a few results for this representation, more can be found in the Appendix A:

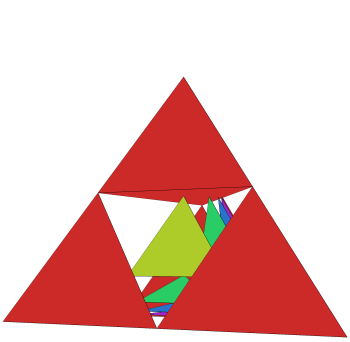


the full graph

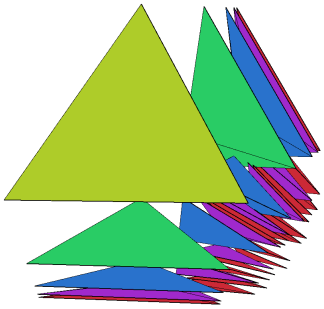
the inner vertices

the inner vertices from above

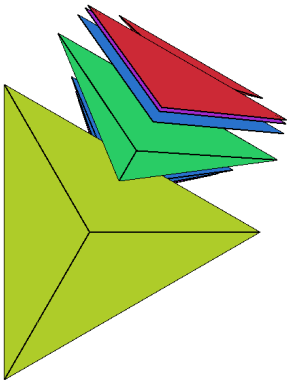
Figure 5.3.: the simple 4-tree $\mathsf{CT}^4_{4\{rgbl\}}$



the full graph

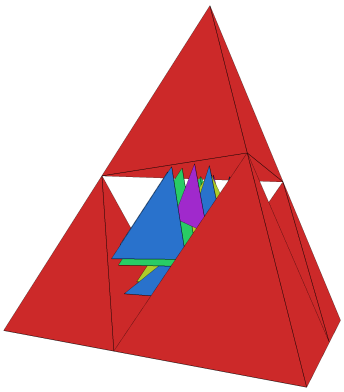


the inner vertices

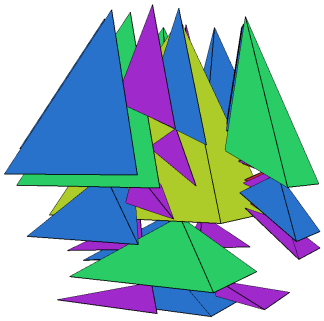


the inner vertices from above

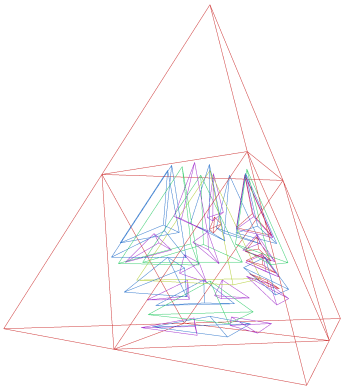
Figure 5.4.: the simple 4-tree $\mathsf{CT}^4_{4\{rg\}}$



the full graph



the graph without the vertices of the outer simplex



the full graph but only the lines are visible

Figure 5.5.: the simple 4-tree $\mathsf{CT}^4_{(r\{rg\}\{bl\})(gblr)(b2\{rg\}g)(l\{gb\}r)(\{rl\}b)}$

5.4. Hexagon Side-Contact-Representation for Simple 3-Trees

Now that we have looked into contact representations of simple 4-trees we want to look into side-contact representations of simple 4-trees. We first start with an easier to understand case of a side-contact representation for simple 3-trees. We can find a hexagon side-contact representation for all planar triangulations as described in `Optimal polygonal representation of planar graphs` [12]. Despite the fact, that we already can find a hexagon side-contact representation we will show another way for simple 3-trees in preparation to find hole-free octahedron side-contact representations for simple 4-trees.

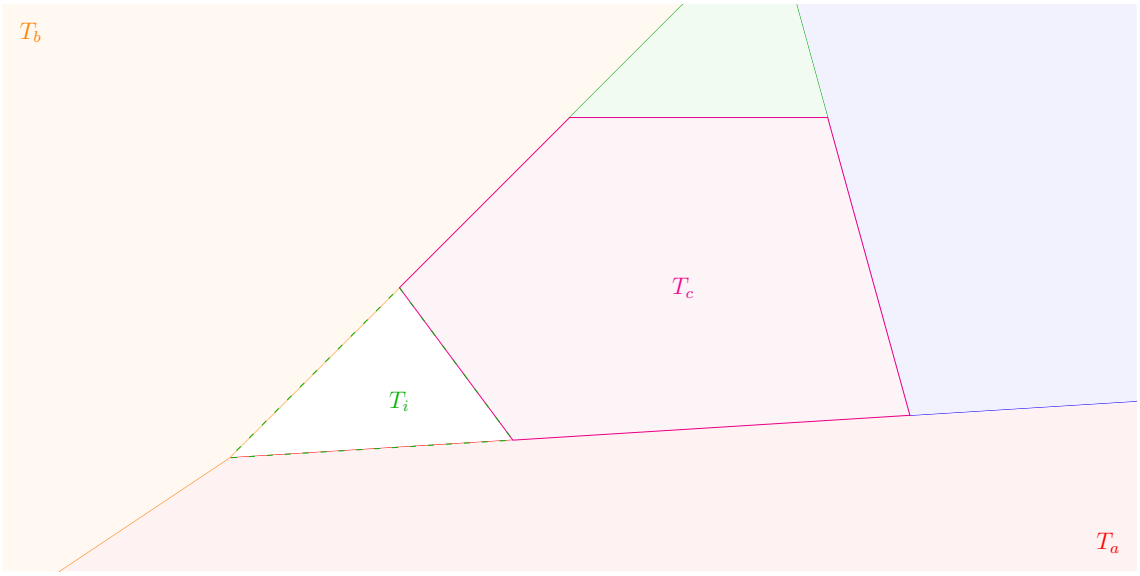


Figure 5.6.: Invariant of construction 5.11

Theorem 5.10. *For every simple* 3*-tree exists a hole-free hexagon side-contact representation in the plane.*

Construction 5.11 (Hexagon Side-Contact Representation)**.**
For this construction we use, that a pentagon, quadrilateral and triangle are all deformed hexagons with $1, 2$ and 3 duplicate vertices respectively. We start the construction by finding a starting configuration for the outer vertices v_1, v_2 and v_3. The outer vertices have to be hexagons, such that they surround a free triangle and share an edge pairwise. We use the free triangle to embed $v_4, \ldots, v_n$ in the construction order of the simple 3-tree.

The invariant for the construction is, that for each vertex $v_i \in \mathsf{CT}^3$ we find a triangle T_i such that the triangle shares an edge with each of the corresponding triangles of the vertices in $e(q(v_i)) = \{T_a, T_b, T_c\}$. The invariant is illustraded in Figure 5.6.

In the step when we place a hexagon H_i for v_i in the drawing Γ, we look at the children of v_i. We encounter four cases, based on the number of children v_i has in CT^3. We write $N^c_{\mathsf{CT}^3}(v)$ as the set of the children of v in CT^3:

$N^c_{\mathsf{CT}^3}(v_i) = \emptyset$
: If v_i has no children, we use T_i as H_i.

$N^c_{\mathsf{CT}^3}(v_i) = \{v_r\}$
: If v_i has one child, we subdivide T_i into one triangle T_r and a quadrilateral H_r. We use T_r to embed the k-color subtree under v_r.

$N^c_{\mathsf{CT}^3}(v_i) = \{v_r, v_s\}$
: If v_i has two children, we subdivide T_i into two triangles T_r, T_s and a pentagon H_i. We embed the subtrees from v_s, v_r into T_r and T_s respectively.

$N^c_{\mathsf{CT}^3}(v_i) = \{v_r, v_s, v_t\}$
: If v_i has all possible children, we cut each corner of the triangle in such a way that we get a hexagon H_i and three triangles T_r, T_s and T_t. We embed the subtree under v_r, v_s, v_t in T_r, T_s, T_t respectively.

In each case for v_i we find a deformed hexagon H_i and leave space so we can embed the children of v_i.

Proof. We have to show that in each case H_i shares a line segment of the edges of T_i. But this is always possible, because we can choose how we subdivide T_i. Therefore, we have to choose for each edge of T_i two points, H_i always connects between the two points to the sides of the surrounding hexagons corresponding to the vertices in $e(q(v_i))$. This results in all hexagons share a line segment with the hexagons corresponding to the vertices in the 3-vertex color set. □

Remark 5.12. *The construction allows a free starting configuration. This gives us the opportunity for example to embed every simple 3-tree inside a hexagon or inside a triangle. The start configurations for this are illustrated in the following Figure 5.7 and 5.8:*

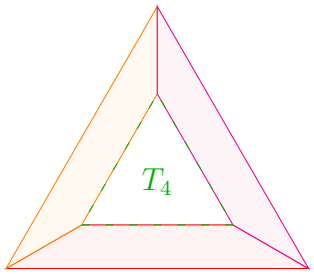


Figure 5.7.: Triangle starting configuration

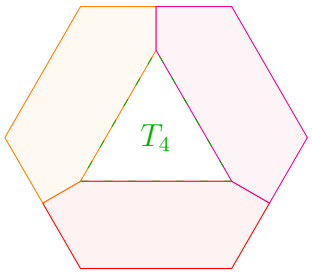


Figure 5.8.: Hexagon starting configuration

5.5. Octahedron Side-Contact Representationf for Simple 4-Trees

Now that we have looked into side-contact representations for simple 3-trees we will look at the side-contact representation for simple 4-trees.

Theorem 5.13. *For every simple 4-tree exists a hole-free octahedron side-contact representation in $\mathbb{R}^3$.*

Construction 5.14.
Octahedron Side-Contact Representation For this construction we use that, we can deform an octahedron into a bipyramid or a tetrahedron by edge-contraction one or two edges respectively. In addition, a frustum is a deformed octahedron, that is the result of a rotated face.
We start our construction again by transforming the simple 4-tree ST^4 into a 4-color tree CT^4 by Construction 3.28. For the outer vertices we find four octahedrons that are arranged in such a way, that the surround a free space of the shape of a tetrahedron.

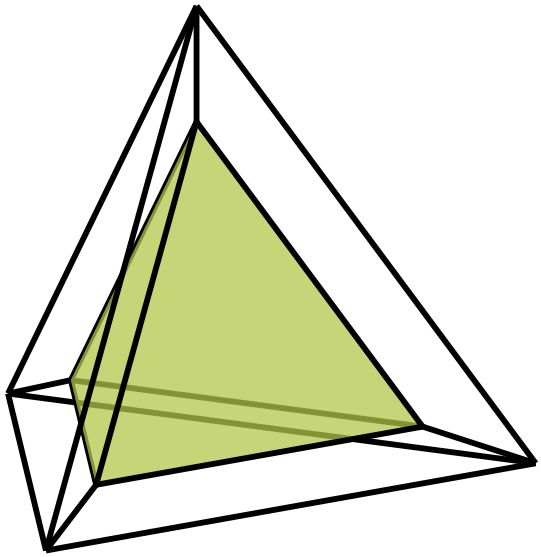


Figure 5.9.: The starting configuration for the construction, the yellow tetraeder is the free inside

This is for example possible when we start with four frustums and arrange them, such that four bases of the frustums create the convex hull of a tetrahedron for all four frustums. The other four bases form the convex hull in the shape of the tetrahedron that surrounds the free space inside the four frustums. This startingconfiguration is illustraded in Figure 5.9.

The invariant for this construction is: For each v_i in the step of the construction where we place v_i, we find a free tetrahedron inside the drawing, where the tetrahedron is surrounded by the octahedrons corresponding to the vertices of $e(q(v_i))$.

We now look at one step of the construction. For v_i we want to find the corresponding octahedron based on the structure of the CT^4. We find T_i as the tetrahedron that is surrounded by the octahedrons corresponding to the vertices in $e(q(v_i))$. We now look at all possible combinations of children, we write $N^c_{\mathsf{CT}^4}(v)$ for the children of v in CT^4. In respect to the children of v_i, we subdivide T_i into an octahedron for v_i and a tetrahedron for each child of v_i. We therefore cut T_i with the $|N^c_{\mathsf{CT}^4}(v_i)|$ cuts into the same number of tetrahedrons and a resulting rest which forms a deformed octahedron. We look at all possible cases, they are each represented in one Figures 5.10, 5.11, 5.12, 5.13, and 5.14.

$N^c_{\mathsf{CT}^4}(v_i) = \emptyset$
: When v_i has no children we use T_i as O_i.

$N^c_{\mathsf{CT}^4}(v_i) = \{v_r\}$
: When v_i has a child v_r, we can subdivide T_i in a tetrahedron T_r and a frustum O_i.

$N^c_{\mathsf{CT}^4}(v_i) = \{v_r, v_s\}$
: We subdivide T_i into a bipyramid O_i and two tetrahedrons T_r and T_s. This subdivision is

not trivial but possible if it holds the following conditions:

(1) $|T_r \cap \mathcal{T}_s| = 1$ (2) $|T_r \cap T_i| = 2$

(3) $|O_i \cap T_i| = 2$ (4) $|T_s \cap T_i| = 2$

These conditions enforce that the two dividing cuts each cut through a vertex of T_i and one common point on the edge of T_i. The result is T_r, T_s and O_i. If the cuts do not hold the conditions, O_i can have more than six vertices and therefore is no octahedron.

$N^c_{\mathsf{CT}^4}(v_i) = \{v_r, v_s, v_t\}$
When v_i has three children, we subdivide T_i in three tetrahedrons T_r, T_s, T_t and an octahedron O_i. The three tetrahedrons share pairwise a vertex. In addition, one of the tetrahedrons share two vertices with T_i, the others only share one vertex with T_i.

$N^c_{\mathsf{CT}^4}(v_i) = \{v_r, v_s, v_t, v_u\}$
We subdivide T_i in four tetrahedrons T_r, T_s, T_t, T_u and the octahedron O_i. This subdivision is possible, since we can cut T_i such that:

$$\forall x \in \{r, s, t, u\} \forall y \in \{r, s, t, u\} \setminus x : |T_x \cap T_y| = 1$$

As a result in each step we find O_i as the deformed octahedron for v_i. In addition, we cut away $|N^c_{\mathsf{CT}^4}(v_i)|$ tetrahedrons from T_i to place the children of v_i.

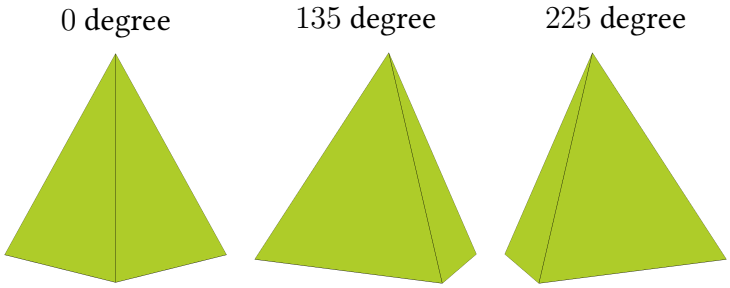


Figure 5.10.: The case of no child, yellow is the octahedron

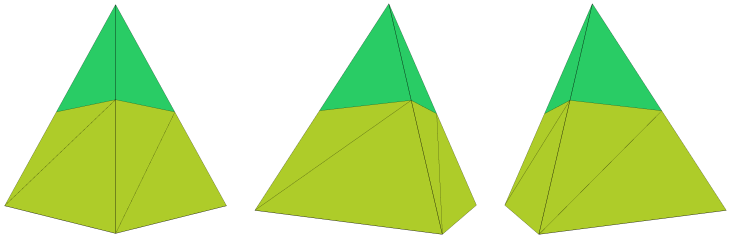


Figure 5.11.: The case of one child, yellow is the octahedron of v_i and the green tetrahedron is the free-space of the child.

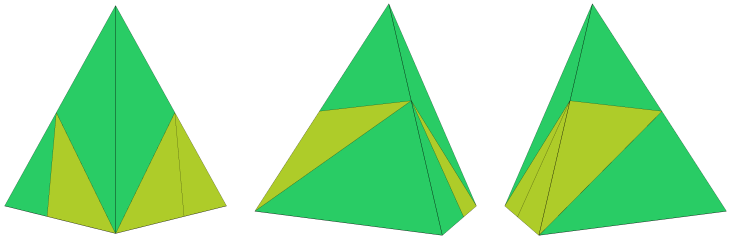


Figure 5.12.: The case of one child, yellow is the octahedron of v_i and the green tetrahedrons are the freespace for the children.

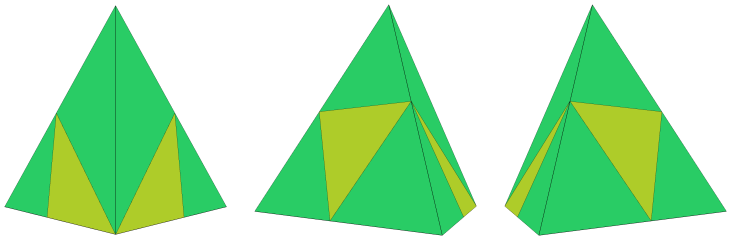


Figure 5.13.: The case of one child, yellow is the octahedron of v_i and the green tetrahedrons are the freespace for the children.

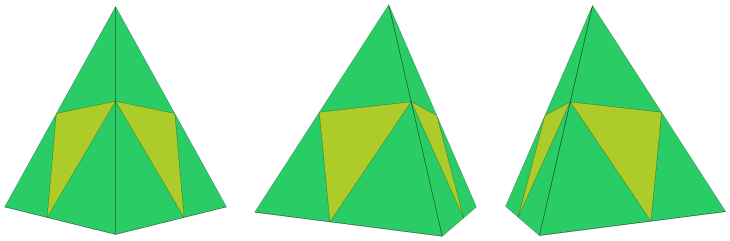


Figure 5.14.: The case of one child, yellow is the octahedron of v_i and the green tetrahedrons are the freespace for the children.

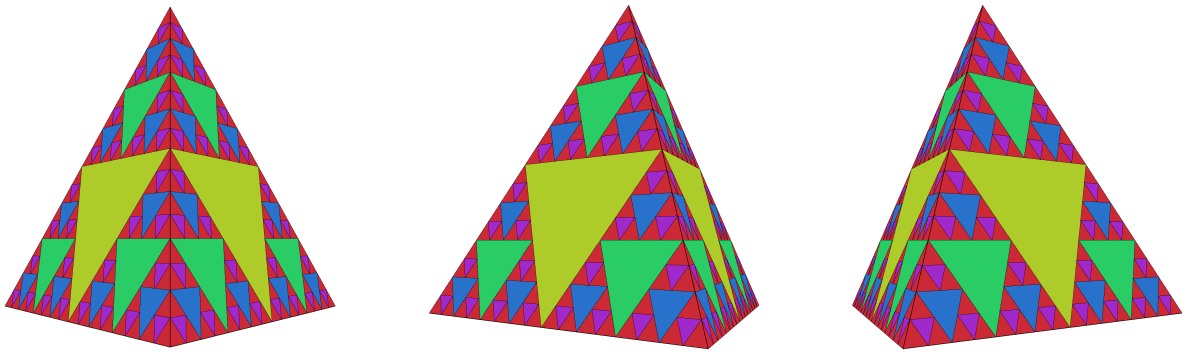
Remark 5.15. *The subdivisions in each of the five cases of Construction 5.14 are all possible if we use the vertices of T_i and all midpoints of all edges of T_i. It is possible that we use an arbitrary point on the edge instead of the midpoint for each edge.*

Proof. We have to show that in each case O_i shares an area with T_i and therefore with the surrounding octahedrons. In addition, we have to show for each children v_d that we find a T_d that shares the faces with the surrounding octahedrons.
When we cut a corner of T_i it is always adjacent to three of the surrounding octahedrons, since T_i has four corners and each has three different surrounding tetrahedrons as neighbors. We find for each possible child of the 4-color tree a corner with the correct adjacent octahedrons. In addition, after the cut O_i is adjacent to a free tetrahedron T_d, so it is always possible to cut the corners of T_i for the children of v_i in every CT^4.

Now we look at the octahedron O_i, O_i is the resulting geometric form of the cut T_i. As long as we do not cut away a complete face of T_i, O_i after the cuts O_i still shares an area with the surrounding octahedrons. With one cut we cannot cut away a complete face of T_i since

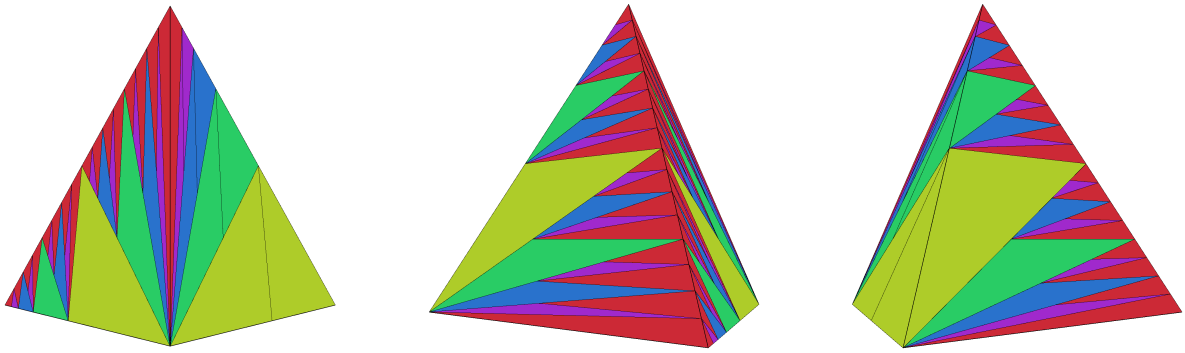
the resulting object would not be a tetrahedron and would not suffice the invariant of the construction. With two or more cuts, it would be possible to cut away a surface of T_i if two cuts share a line segment on the surface of T_i. But we require that two resulting tetrahedrons T_x and T_y for the children v_x and v_y only share one common point and therefore such cuts are also not possible. □

We now show a few results for the octahedron representation, for all representations, we removed theoctahedrons corresponding to the vertices of the outer simplex:



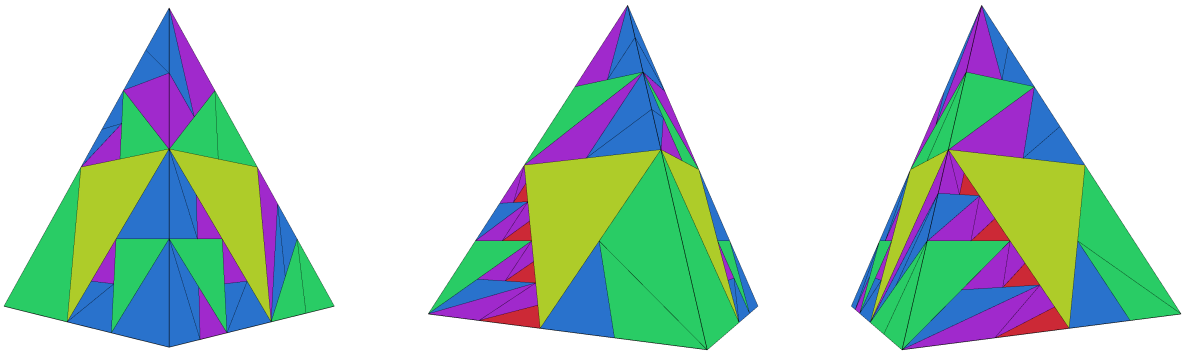
rotated 0 degree | rotated 135 degree | rotated 225 degree

Figure 5.15.: the simple 4-tree $\mathsf{CT}^4_{4\{rgbl\}}$



rotated 0 degree | rotated 135 degree | rotated 225 degree

Figure 5.16.: the simple 4-tree $\mathsf{CT}^4_{4\{rg\}}$



rotated 0 degree rotated 135 degree rotated 225 degree

Figure 5.17.: the simple 4-tree $\mathsf{CT}^4_{(r\{rg\}\{bl\})(gblr)(b2\{rg\}g)(l\{gb\}r)(\{rl\}b)}$

In Figure 5.15 we leafes of the CT^k form the Sierpinski-tetrahedron, all other vertices fill the holes between the tetrahedrons with regular octahedrons. In particular the tetrahedrons in the representation of $\mathsf{CT}^4_{n\{rgbl\}}$ for $n \in \mathbb{N}$ form the tetrahedrons of the Sierpinski-tetrahedron constructed in the n-th step of the recursive construction.

6. Conclusion

In this thesis we looked into a generalized version of Schnyder realizers for higher dimensions. When looking into the structure of triangulations for higher dimension, we recognized a connection between minimal k-triangulations, simple k-trees and Schnyder realizers with k colors. For maximal k-triangulations, we prove for $k = 4$ that K_n can be embedded into a tetrahedization. We assume, but do not prove, that this also holds for even higher dimensions. Furthermore, we believe that this can be proven by a similar construction as ours for 4-triangulations.

We have shown, under the Assumption 3.19 that minimal triangulations have a Schnyder realizer on k colors. We use a generalized version of canonical orderings to find the corresponding simple k-tree and the k-color realizer itself.

Under the Conjecture 3.24 that for $k > 3$ simple k-trees are minimal triangulations has a few implications. If the application does not require a specific structure for the higher dimensional triangulation, we can always use a minimal triangulation. This is also possible if the convex hull is not a k-simplex. In this case where we can use a minimal inner triangulation.
In addition, k-trees are chordal, that is why we can use the efficient chordal graph algorithms to find a vertex coloring, for example.

In the paper [8] the authors describe a combinatorical generalized version of a Schnyder realizer. They are described by k orderings of the vertices. We now look at the Venn diagramm of the k-triangulations, k-realizers and k-color realizers.
Figure 6.1 describes the relations between the graphs which are a k-triangulation, a k-realizer or a k-color realizer. The simple-k-trees are the graphs which are k-triangulations and have a k-realizer and a k-color realizer. They are the only graphs which are k-triangulations and have a k-color realizer. Therefore, no k-triangulation exists which has a k-color realizer but no k-realizer. For k-realizer we can state the same result. Since k-realizers on n vertices have also $ST^k(n)$ edges and a k-triangulation with $ST^k(n)$ vertices is minimal, the only triangulations with a k-realizer are simple k-tree.
Apex triangulations are $k-1$-triangulations with $ST^{k-1}(n)$ vertices and an additional vertex that is fully connected to all other vertices. These graphs have a k-realizer and a k-color realizer, but are no k-triangulation. In addition, some non-simple k-trees have a k-realizer [8]. Non-simple k-trees are all k-trees that are not a simple k-tree. We know, that all k-trees have a k-color realizer. Since we can embed every vertex of the k-tree inside the convex hull of the clique[1] we can show by a similar to the proof Theorem 4.11 (IV) $\Rightarrow$ (III) that k-trees have a

[1]We have to ensure that no edge collides with other edges of other vertices that are connected to the same clique, but this is always possible

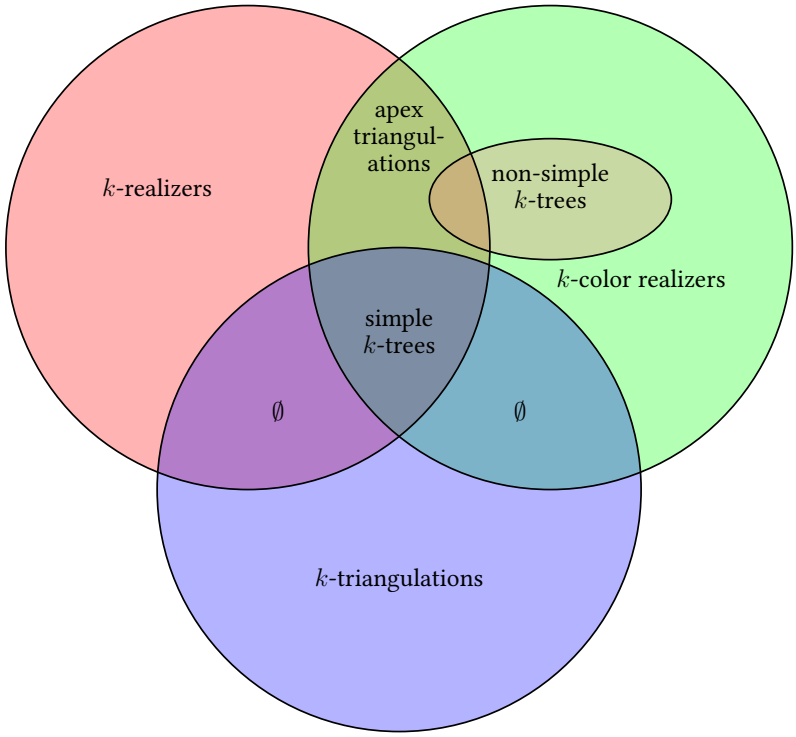


Figure 6.1.: The Venn diagramm of the known graph classes for k-triangulations, k-realizers and k-color realizers.

k-color realizer. This result shows that k-realizers and k-color realizers are not the same concept.

Based on the results on the connections between k-triangulations, simple k-trees, the k-realizers, k-color realizers and the k-realizers of the paper [8], we state a few new questions:
The first and most important, do we need the Assumption 3.19.

Question 1. *Do we need that a k-triangulation is steady, or can we find for all k-triangulation a canonical ordering?*

We have only proven under Conjecture 3.24 that a simple k-tree is a minimal k-traingulation.

Question 2. *Can we prove Conjecture 3.24?*

Besides k-triangulations, simple k-trees and apex traingulations, we do not know many examples, that induce a k-color realizer.

Question 3. *What are other graph classes have a k-color realizer?*

We have shown that minimal k-triangulations have a k-realizer and a k-color realizer and that there exist other graphs that have both realizers, but also graphs that have only a k-color realizer. The difference between the two generalizations of a Schnyder realizers is interesting and leads to the following question.

Question 4. *Do all graphs with a k-realizer also have a k-color realizer or are there graphs with a k-realizer but no k-color realizer?*

Together with the previous two question, the next question arises.

Question 5. *Are the apex triangulations the only graphs that are no k-triangulations which induce a k-realizer and a k-color realizer or are there other graphs with this property?*

For Schnyder realizers we know that the edge set of the planar triangulation decomposes into three trees and the outer triangle. For a graph with a k-color realizer it is unclear if the graph decomposes into k trees and a k-simplex.

Question 6. *Is it possible for a k-color realizer that the graph has a directed cycle of one color?*

In the second part of the thesis we looked into representations of simple k-trees. We have shown for $k = 3$ and $k = 4$ there exists a k-simplex contact representation for all simple k-trees. For $k = 3$ this was already shown in the papers [6, 12] for all planar triangulations. Our main contribution for $k = 3$ is a construction on a new technique for building representations for k-color trees. k-color trees, are compact representation of simple k-trees and are a subgraph of the k-color realizer.
We use this technique to find a tetrahedron contact representation and the octahedron side-contact representations for simple 4-trees.

The tetrahedron contact representation described in this thesis only constructs irregular tetrahedrons. We have a flexibility in constructing the tetrahedrons, but we are limited by the surrounding tetrahedrons. In complex simple k-trees the free-space of a vertex is sometimes degenerated and we think it is not possible to find for all simple k-trees a homothetic tethrahedron contact representation. But with the current knowledge we are not possible to proof this.

The octahedron contact representations for simple 4-trees is hole-free but for a vertex in a k-color tree with two, three and four children the octahedrons are irregular and have no geometric properties. If we look at the case of the k-color tree $\mathsf{CT}^4_{n\{rgbl\}}$ for $n \in \mathbb{N}$. The representation of this special simple k-trees are equivalent to a Sierpiński-tetrahedron if we remove the tetrahedrons of the outer vertices v_1, v_2, v_3, v_4. In the representation only the leaves of the $\mathsf{CT}^4_{n\{rgbl\}}$ form the tetrahedrons, all other vertices fill the gaps between the tetrahedron with regular octahedrons. An image of the representation can be seen in Figure 5.15.

The newly presented representations of simple k-trees raise further questions:

Question 7. *Can we proof that for all simple k-trees there exists no homothetic tetrahedron contact representation based on our construction of irregular tetrahedrons?*

For planar triangulations there are a lot of different contact representations.

Question 8. *Can we find other contact representations that visualize simple k-trees or even graphs with a k-color realizer?*

Bibliography

[1] Muhammad Jawaherul Alam. "Contact representations of graphs in 2D and 3D". PhD thesis. The University of Arizona, 2015.

[2] Luca Castelli Aleardi, Éric Fusy, and Thomas Lewiner. "Schnyder woods for higher genus triangulated surfaces, with applications to encoding". In: *Discrete & Computational Geometry* 42 (2009), pp. 489–516.

[3] Nicolas Bonichon, Stefan Felsner, and Mohamed Mosbah. "Convex drawings of 3-connected plane graphs". In: *Algorithmica* 47.4 (2007), pp. 399–420.

[4] Luca Castelli Aleardi, Éric Fusy, and Thomas Lewiner. "Schnyder woods for higher genus triangulated surfaces". In: *Proceedings of the twenty-fourth annual symposium on Computational geometry*. 2008, pp. 311–319.

[5] Steven Chaplick, Stephen G Kobourov, and Torsten Ueckerdt. "Equilateral L-contact graphs". In: *Graph-Theoretic Concepts in Computer Science: 39th International Workshop, WG 2013, Lübeck, Germany, June 19-21, 2013, Revised Papers 39*. Springer. 2013, pp. 139–151.

[6] Hubert De Fraysseix, Patrice Ossona de Mendez, and Pierre Rosenstiehl. "On triangle contact graphs". In: *Combinatorics, Probability and Computing* 3.2 (1994), pp. 233–246.

[7] Jesús De Loera, Jörg Rambau, and Francisco Santos. *Triangulations: structures for algorithms and applications*. Vol. 25. Springer Science & Business Media, 2010.

[8] William Evans et al. "Graphs admitting d-realizers: spanning-tree-decompositions and box-representations". In: *Proc. of EuroCG* 14 (2014).

[9] Stefan Felsner et al. "Bijections for Baxter families and related objects". In: *Journal of Combinatorial Theory, Series A* 118.3 (2011), pp. 993–1020.

[10] Hubert de Fraysseix, János Pach, and Richard Pollack. "Small sets supporting Fary embeddings of planar graphs". In: *Proceedings of the twentieth annual ACM symposium on Theory of computing*. 1988, pp. 426–433.

[11] Éric Fusy. "Transversal structures on triangulations: A combinatorial study and straight-line drawings". In: *Discrete Mathematics* 309.7 (2009), pp. 1870–1894.

[12] Emden R Gansner et al. "Optimal polygonal representation of planar graphs". In: *LATIN 2010: Theoretical Informatics: 9th Latin American Symposium, Oaxaca, Mexico, April 19-23, 2010. Proceedings 9*. Springer. 2010, pp. 417–432.

[13] Fănică Gavril. "Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph". In: *SIAM Journal on Computing* 1.2 (1972), pp. 180–187.

[14] Xin He and Huaming Zhang. "Schnyder greedy routing algorithm". In: *Theory and Applications of Models of Computation: 7th Annual Conference, TAMC 2010, Prague, Czech Republic, June 7-11, 2010. Proceedings 7*. Springer. 2010, pp. 271–283.

[15] Goos Kant and Xin He. "Regular edge labeling of 4-connected plane graphs and its applications in graph drawing problems". In: *Theoretical Computer Science* 172.1-2 (1997), pp. 175–193.

[16] Philip N Klein. "Efficient parallel algorithms for chordal graphs". In: *[Proceedings 1988] 29th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society. 1988, pp. 150–161.

[17] Yiting Li, Xin Sun, and Samuel S Watson. "Schnyder woods, SLE (16), and Liouville quantum gravity". In: *arXiv preprint arXiv:1705.03573* (2017).

[18] Jesús Loera, Jörg Rambau, and Francisco Santos. *Triangulations. Structures for algorithms and applications*. Vol. 25. Jan. 2010. ISBN: 978-3-642-12970-4. DOI: 10.1007/978-3-642-12971-1.

[19] Walter Schnyder. "Planar graphs and poset dimension". In: *Order* 5 (1989), pp. 323–343.

[20] Walter Schnyder. "Embedding planar graphs on the grid". In: *Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*. 1990, pp. 138–148.

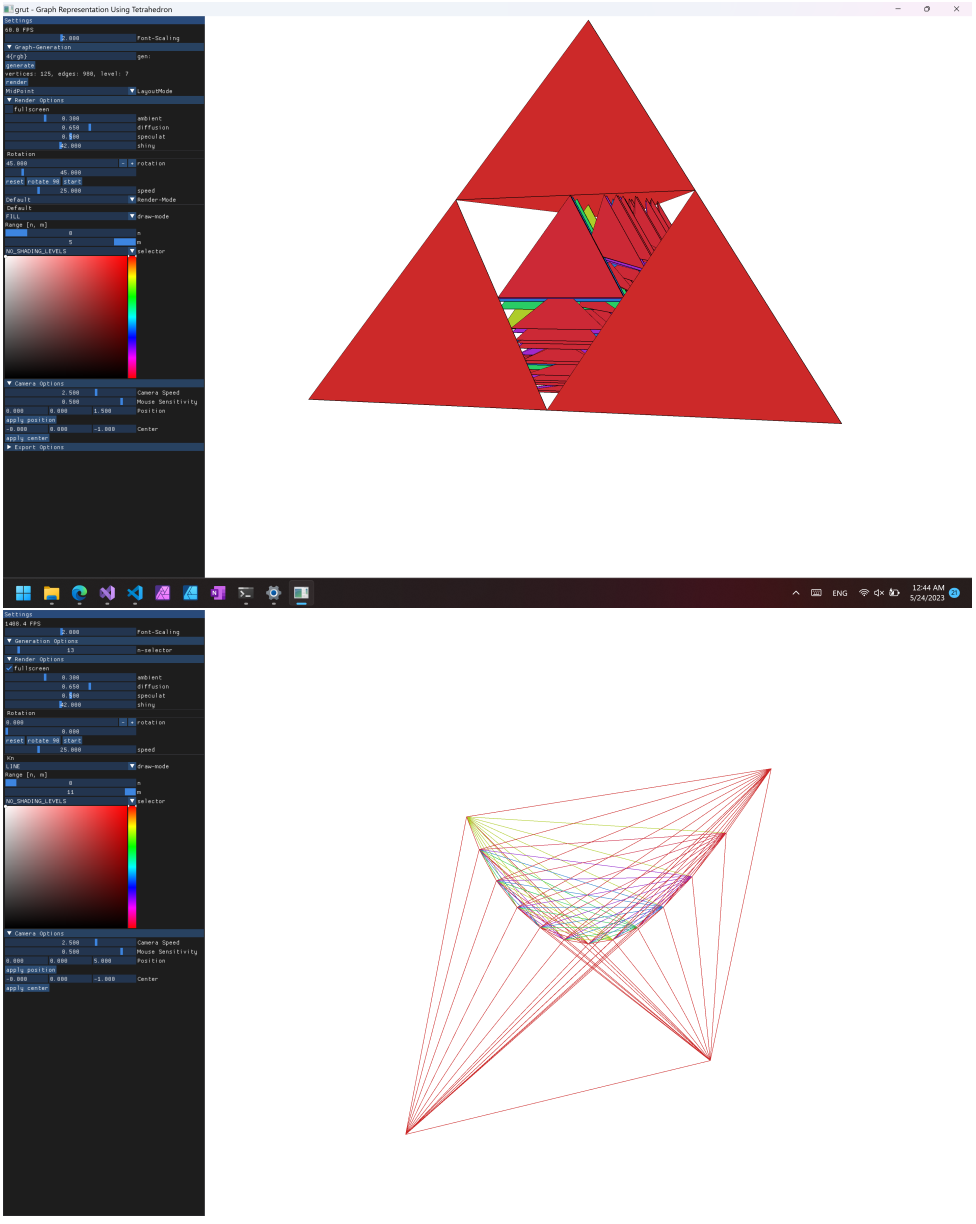
[21] Douglas R Shier. "Some aspects of perfect elimination orderings in chordal graphs". In: *Discrete Applied Mathematics* 7.3 (1984), pp. 325–331.

[22] Jinjin Yang et al. "Average geodesic distance of skeleton networks of Sierpinski tetrahedron". In: *Physica A: Statistical Mechanics and its Applications* 495 (2018), pp. 269–277.

A. Appendix

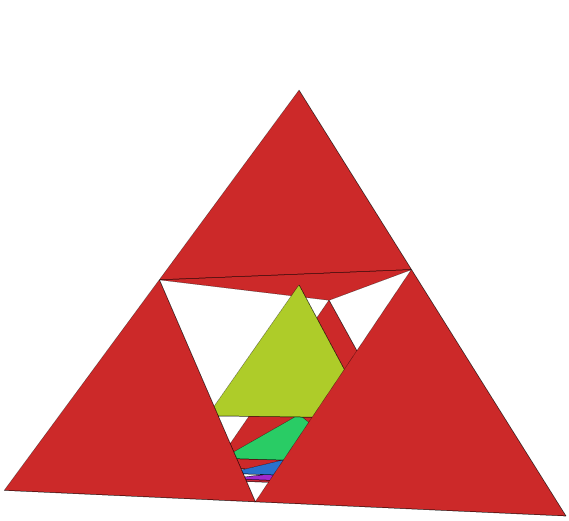
A.1. Grut

The source code implementing the representations of this thesis is open-source on github: `https://github.com/knorrfix/grut`
In the following are a few pictures of the software:

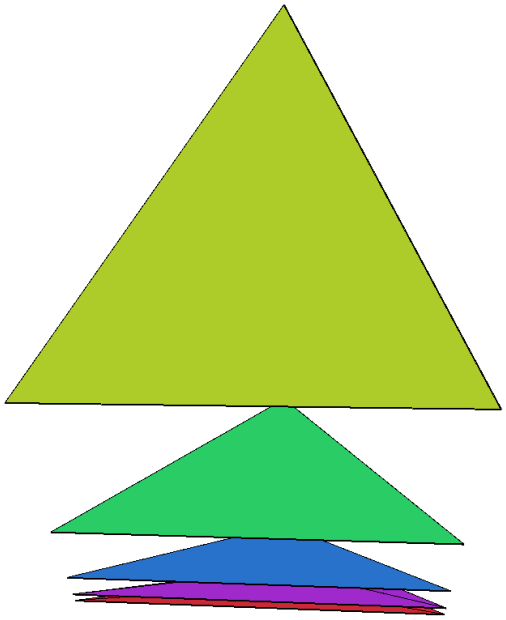


A.2. Additional Representations

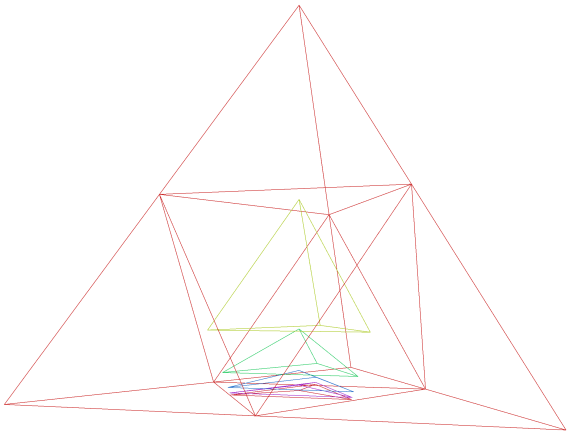
A.2.1. Tetrahedron Contact Representation



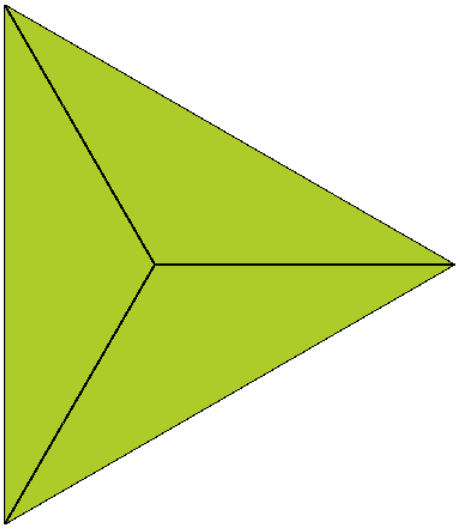
full graph



inner vertices

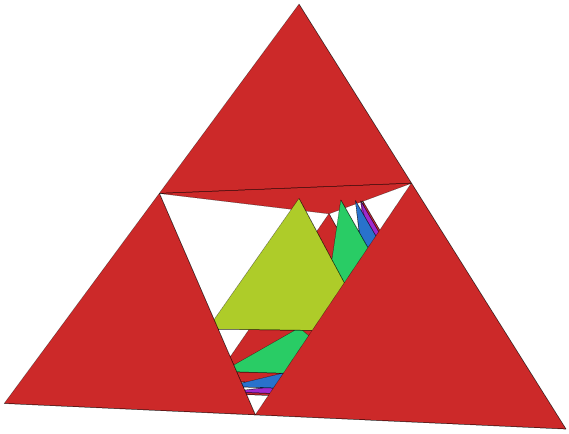


the outline of the full graph

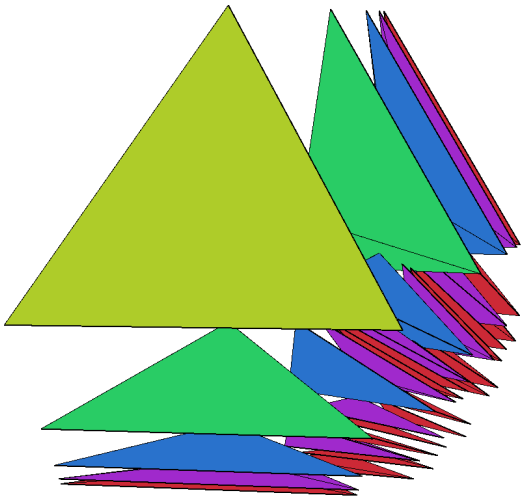


inner vertices from above

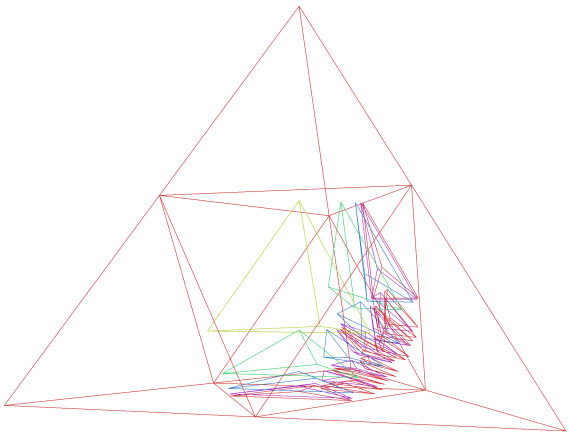
Figure A.1.: The simple 4-tree CT^4_{4r}.



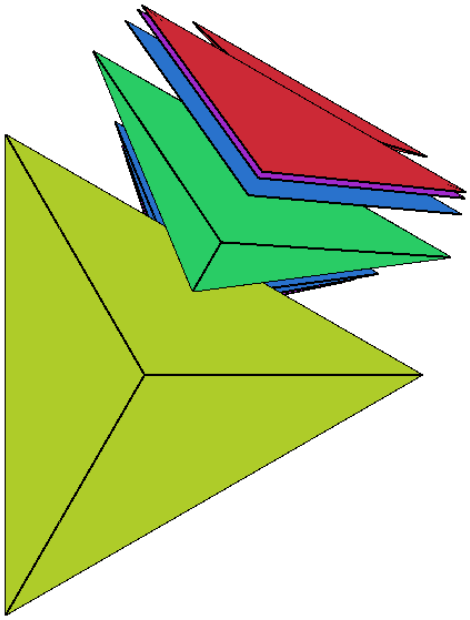
full graph



inner vertices

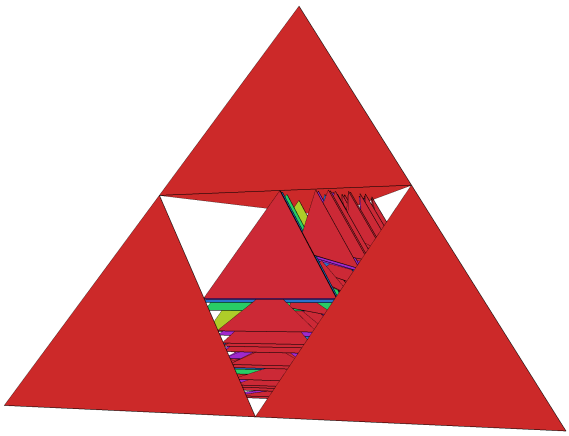


the outline of the full graph

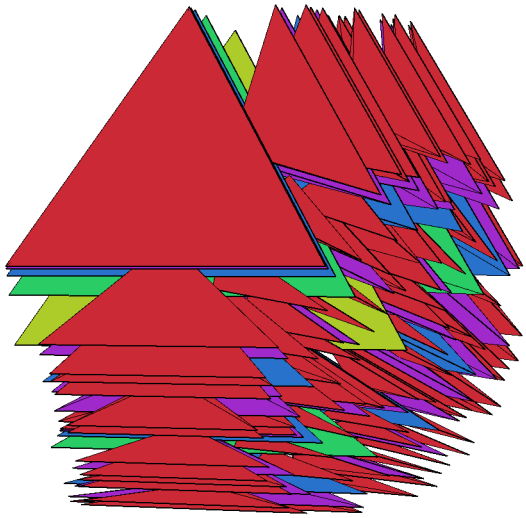


inner vertices from above

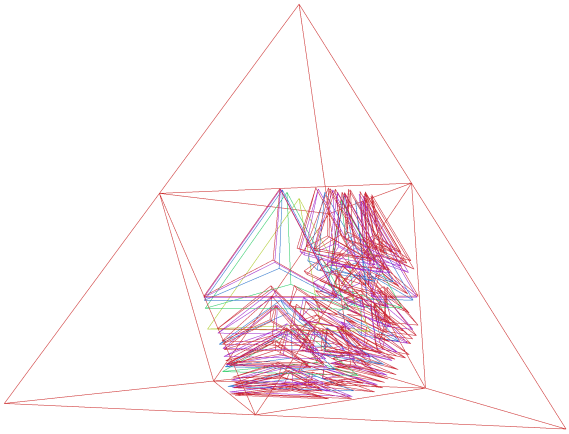
Figure A.2.: The simple 4-tree $\mathsf{CT}^4_{4\{rg\}}$.



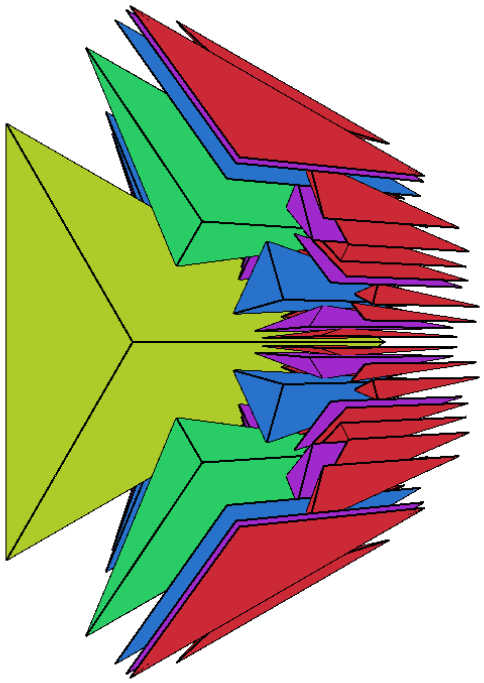
full graph



inner vertices

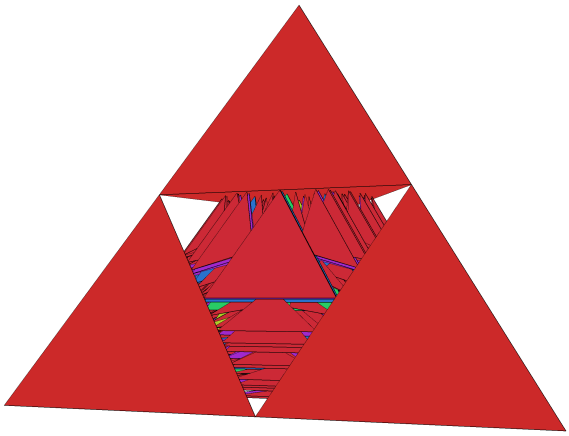


the outline of the full graph

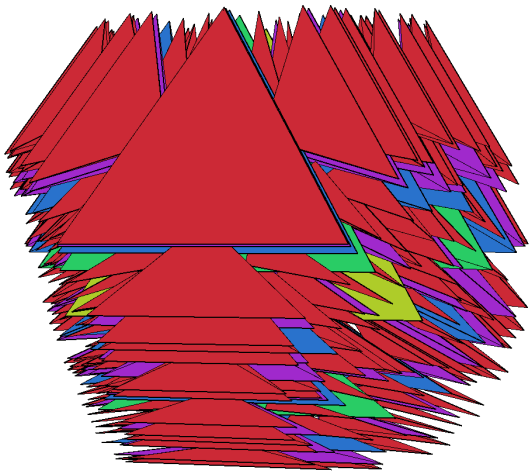


inner vertices from above

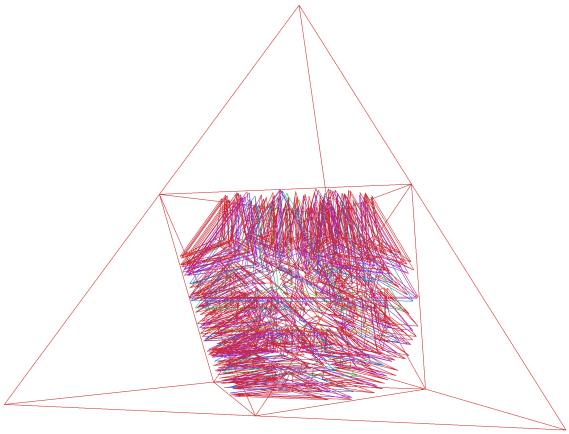
Figure A.3.: The simple 4-tree $\mathsf{CT}^4_{4\{rgb\}}$.



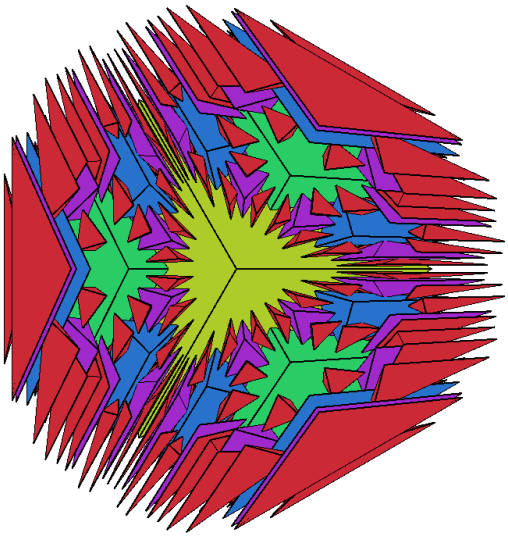
full graph



inner vertices

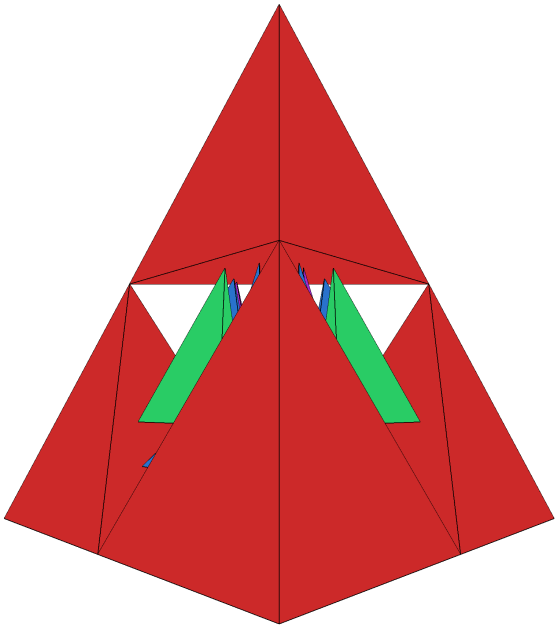


the outline of the full graph

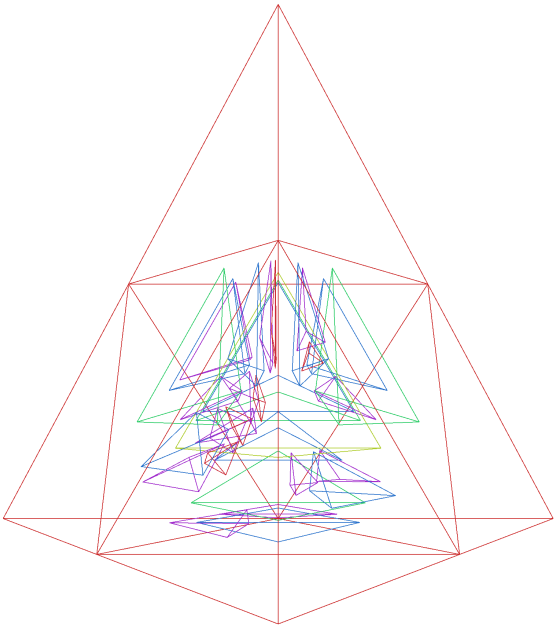


inner vertices from above

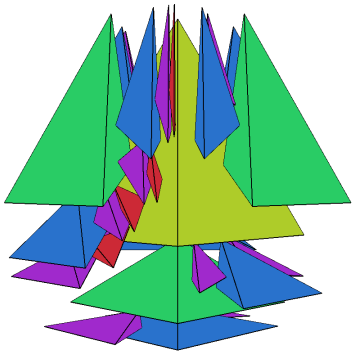
Figure A.4.: The simple 4-tree $\mathsf{CT}^4_{4\{rgbl\}}$.



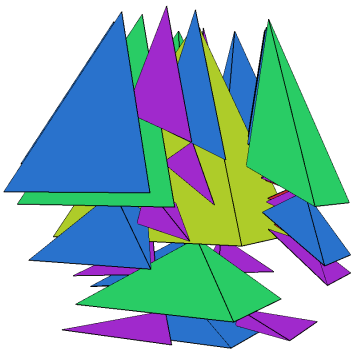
full graph



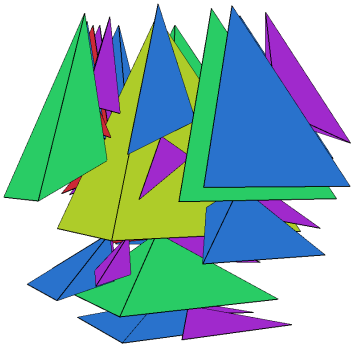
inner vertices from above



inner vertices 0 degree



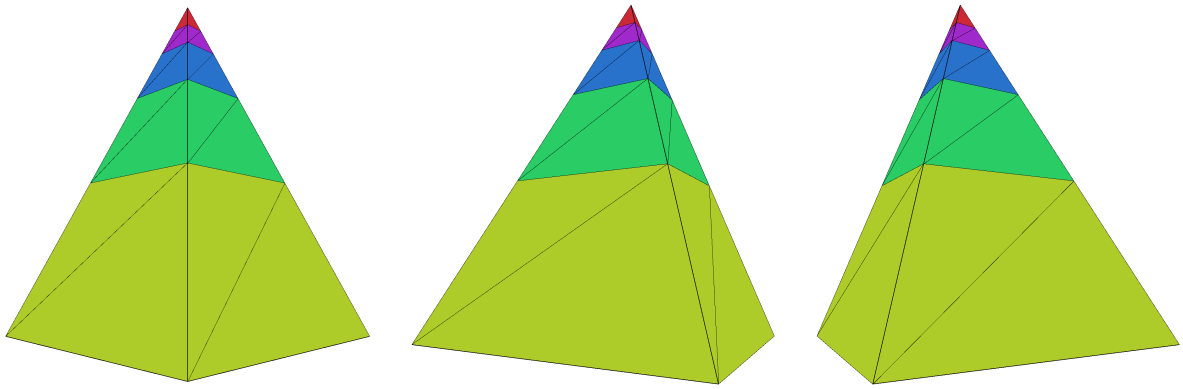
inner vertices 135 degree



inner vertices 225 degree

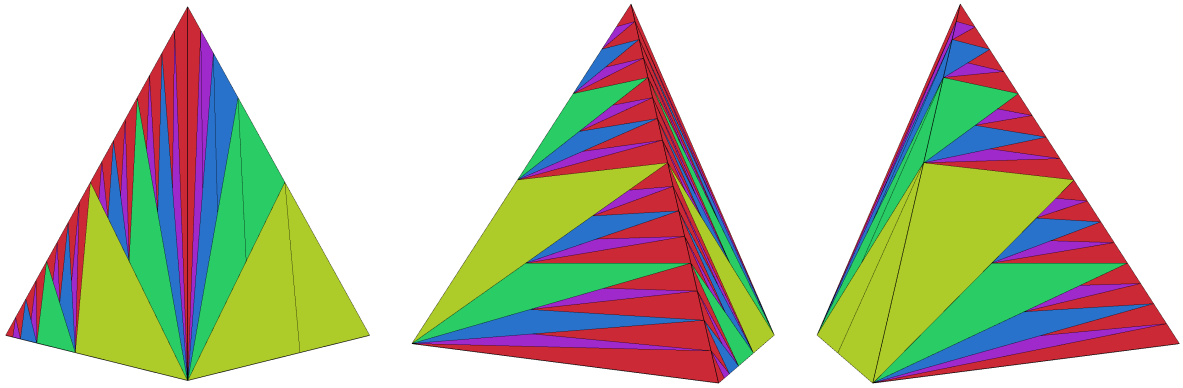
Figure A.5.: The simple 4-tree $\mathsf{CT}^4_{(r\{rg\}\{bl\})(gblr)(b2\{rg\}g)(l\{gb\}r)(\{rl\}b)}$.

A.2.2. Octahedron Side-Contact Representation



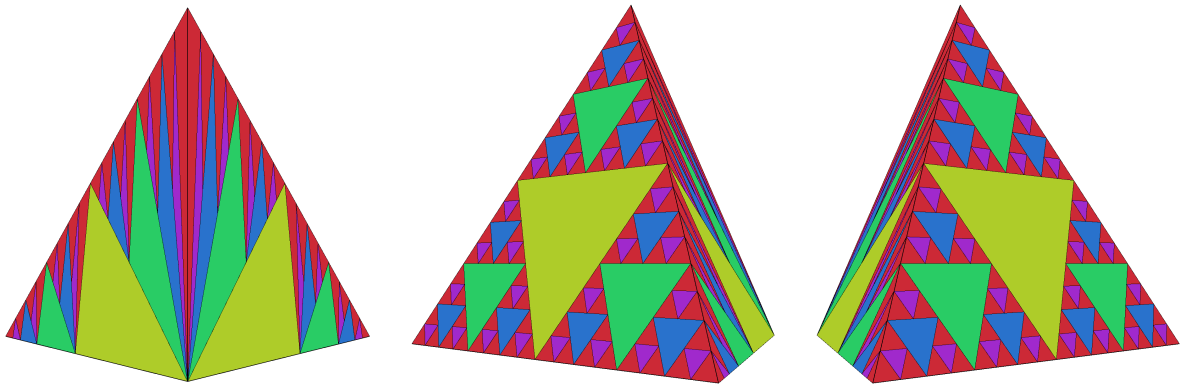
0 degree 135 degree 225 degree

Figure A.6.: The simple 4-tree CT^4_{4r}.



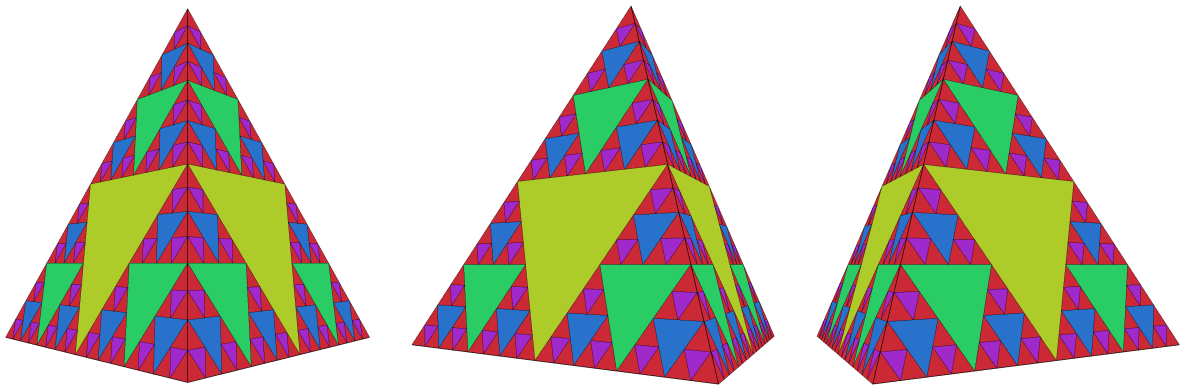
0 degree 135 degree 225 degree

Figure A.7.: The simple 4-tree $\mathsf{CT}^4_{4\{rg\}}$.



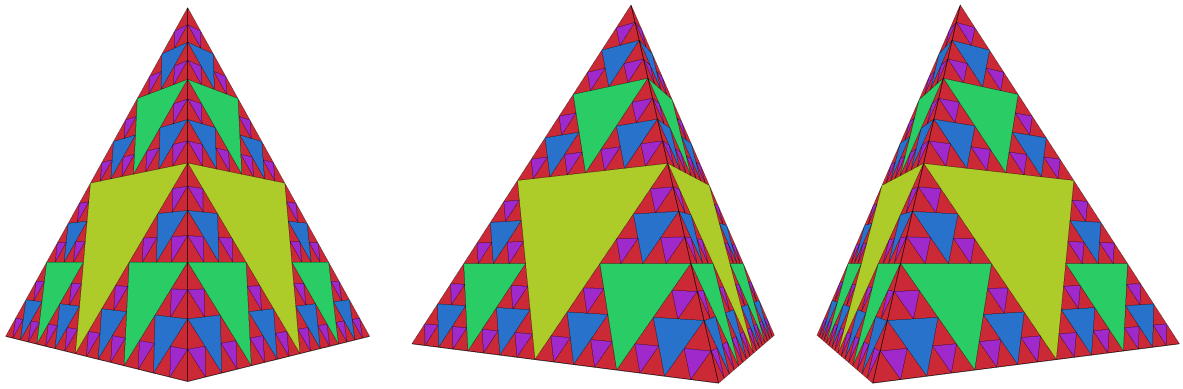
0 degree 135 degree 225 degree

Figure A.8.: The simple 4-tree $\mathsf{CT}^4_{4\{rgb\}}$.



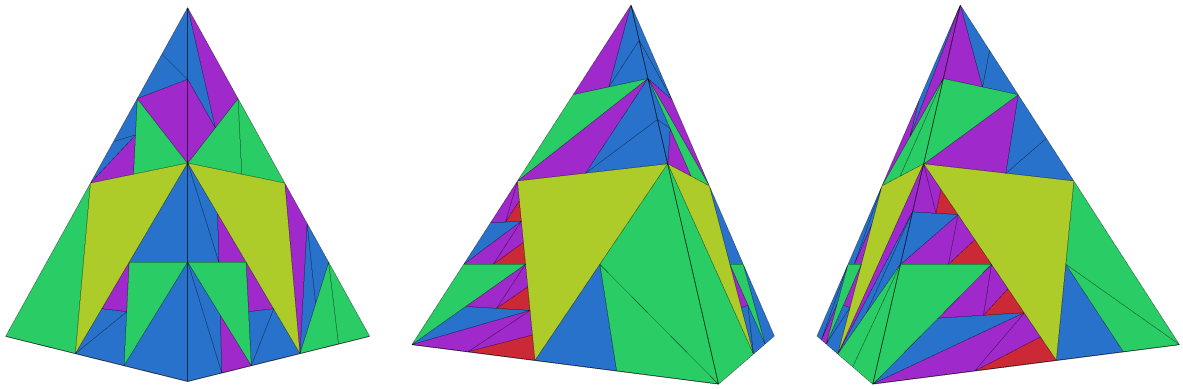
0 degree 135 degree 225 degree

Figure A.9.: The simple 4-tree $\mathsf{CT}^4_{4\{rgbl\}}$.



0 degree 135 degree 225 degree

Figure A.10.: The simple 4-tree $\mathsf{CT}^4_{4\{rgbl\}}$.



0 degree 135 degree 225 degree

Figure A.11.: The simple 4-tree $\mathsf{CT}^4_{(r\{rg\}\{bl\})(gblr)(b2\{rg\}g)(l\{gb\}r)(\{rl\}b)}$.

B. List of Figures