Algorithmic Symplectic Packing

Inaugural-Dissertation

zur

Erlangung des Doktorgrades

der Mathematisch-Naturwissenschaftlichen Fakultät

der Universität zu Köln

vorgelegt von

Greta Fischer

aus Lendersdorf

Köln 2021

Berichterstatter (Gutachter):

Prof. Dr. Michael Jünger,	Universität zu Köln
Prof. Dr. Hansjörg Geiges,	Universität zu Köln

Tag der mündlichen Prüfung: 12.04.2021

Abstract

In this thesis we explore a symplectic packing problem where the targets and domains are $2n$-dimensional symplectic manifolds. We work in the context where the manifolds have first homology group equal to $\mathbb{Z}^n$ and we require the embeddings to induce isomorphisms between first homology groups. In this case, the problem can be related to a combinatorial optimization problem, namely packing certain allowable simplices into a given standard simplex. We design a computational approach to determine the corresponding k-simplex packing widths for up to $k = 13$ simplices in dimension four and $k = 8$ simplices in dimension six.

Acknowledgements

First and foremost, I thank Michael Jünger for giving me the best mixture of freedom, support and encouragement I could have asked for. His eagerness and enthusiasm for computer science will always be an inspiration for me. Many thanks also go to the other two principal investigators of the project: Frank Vallentin and Hansjörg Geiges. Both have always been there for any open questions. I am also indebted to the Deutsche Forschungsgemeinschaft for providing financial support for this project.

To my coworker Jean Gutt I am very thankful for giving me private lectures on differential geometry, for the active exchange of ideas and for the countless table soccer matches he let me win. I feel lucky to have met him and to call him my friend today. I thank Thomas Lange for always having an immediate solution for any technical problems. I also thank my other colleagues Göntje Teuchert, Michael Belling and former colleagues Sven Mallach, Daniel Schmidt and Martin Gronemann for sharing thoughts and for diverting lunch and coffee breaks.

I thank Franz Rendl and Angelika Wiegele for inviting me to Klagenfurt and for the fruitful discussions. Likewise, I am thankful to Felix Schlenk for his encouraging ideas. For proofreading this thesis I want to thank Silvia Gruner, Sven Mallach, Elena Kohlwey, Stefan Krupp, Christian Lange, Anke Paulus and Christian Weiss.

Two years after starting this project I became part of a packing problem myself. I was pregnant with triplets and we had to spend a long time at the hospital. The first months at home also were very exhausting. Overcoming this time and finishing this thesis would have been impossible without the support of my family. I want to thank my parents, my sister and my family in law for their great backup. Last but by no means least, I want to thank Jörg Fischer for being the best husband and loving father to our wonderful kids Emmi, Lotta and Pepe I could imagine.

Contents

List of Figures

List of Tables

Introduction

Symplectic geometry arose as the geometry of classical mechanics two centuries ago and nowadays is a central branch of differential topology. It has connections with quantum mechanics, representation theory, equivariant cohomology, algebraic geometry and partial differential equations, just to name a few.

Symplectic packings lie at the heart of symplectic geometry. Finding explicit symplectic packings is a huge challenge, both theoretically and computationally. In this thesis we will present a computational approach to explore a symplectic packing problem. More precisely, the targets and domains are $2n$-dimensional symplectic manifolds that have first homology group equal to $\mathbb{Z}^n$ and the embeddings induce isomorphisms on first homology.

As this thesis is a symbiosis of symplectic geometry and combinatorial optimization, we will give a short introduction to both in Chapter 1 and Chapter 2. We then shall have the necessary knowledge to design an algorithmic approach for a certain symplectic packing problem, namely the computation of the k-simplex packing width of the four-dimensional prism $s_k\left(P^4(r), \omega_0\right)$. The core of this algorithmic approach is a mixed integer linear program embedded in a branch-and-bound framework. We will describe the algorithm and results in Chapter 3. The algorithmic approach builds up on work of Maley, Mastrangeli and Traynor [MMT00].

Instead of using a mixed integer linear program to model the problem, we can also set up a quadratically constrained quadratic program that can then be relaxed to a semidefinite program. We will describe this approach and further improvement strategies to the algorithm in Chapter 4.

In Chapter 5 we will extend the algorithm to the next higher dimension and compute the k-simplex packing width of the six-dimensional prism $s_k\left(P^6(r), \omega_0\right)$.

The appendix is accompanied by a compact disc that contains the source code of all described computer programs and the corresponding output files.

Chapter 1

Foundations of Symplectic Geometry

In this chapter we will give a short introduction to symplectic geometry. It is mainly based on the books of McDuff and Salamon [MS17] and Cannas da Silva [Sil04].

1.1 Symplectic Manifolds

Symplectic manifolds are the central object in symplectic geometry. Formally, they are defined as follows.

Definition 1.1. A *symplectic manifold* (M, ω) is a pair consisting of a manifold M and a symplectic form ω on M. A *symplectic form* ω on a manifold M is a closed non-degenerate 2-form on M.

A 2-form ω on a manifold M is a bilinear skew-symmetric map on the tangent space T_pM at every point $p \in M$. It is called closed if the exterior derivative vanishes, that is $d\omega = 0$. It is called non-degenerate if for all $p \in M$ and for all nonzero vectors $u \in T_pM$ there exists some other vector $v \in T_pM$ such that $\omega(u, v) \neq 0$.

The non-degeneracy condition enforces a symplectic manifold to be even dimensional. To see this, we write the bilinear skew-symmetric map $\omega : T_pM \times T_pM \to \mathbb{R}$

as

$$\omega(u,v) = u^T A v$$

for each $p \in M$. Here, A is a skew-symmetric matrix of size $n \times n$ with $n = \dim(T_p M) = \dim(M)$. If n is odd, then

$$\begin{aligned}\det(A) &= \det\left(A^T\right) \\ &= \det(-A) \\ &= (-1)^n \det(A)\end{aligned}$$

implies $\det(A) = 0$ and therefore the existence of a nonzero vector $u \in T_p M$ such that $Au = 0$. This in turn leads to the identity

$$\begin{aligned}\omega(u,v) &= -\omega(v,u) \\ &= -v^T A u \\ &= -v^T 0 \\ &= 0\end{aligned}$$

for all $v \in T_p M$, which contradicts the non-degeneracy condition.

As already mentioned in the introduction, symplectic geometry has its historical origin in classical mechanics. The easiest example of a symplectic manifold is the phase space $M = \mathbb{R}^2$ endowed with the symplectic form

$$\omega_0 = dx \wedge dy.$$

Here, x and y describe a particle moving in one dimension with position x and momentum y. The symplectic form ω_0 measures the area of each open region S in the plane by integration:

$$\text{area}(S) = \int_S \omega_0.$$

This area is an important quantity because it is preserved under time evolution. The symplectic form ω_0 is referred to as the canonical symplectic form. It naturally extends to

$$\omega_0 = \sum_{i=1}^{n} dx_i \wedge dy_i$$

in dimension $2n$. Let $J \in \mathbb{R}^{2n\times 2n}$ be the matrix given by

$$J = \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix},$$

where $I \in \mathbb{R}^{n\times n}$ denotes the identity matrix. Then the canonical symplectic form also has the following representation.

Lemma 1.2. *Let $u, v \in \mathbb{R}^{2n}$. Then $\omega_0(u, v) = u^T Jv$.*

Proof. Let $u = \left(\begin{smallmatrix} a \\ b \end{smallmatrix}\right), v = \left(\begin{smallmatrix} c \\ d \end{smallmatrix}\right)$ with $a, b, c, d \in \mathbb{R}^n$. Then

$$\begin{aligned} \omega_0(u, v) &= \sum_{i=1}^{n} dx_i \wedge dy_i(u, v) \\ &= \sum_{i=1}^{n} \det\left(\begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix}\right) \\ &= \sum_{i=1}^{n} a_i d_i - c_i b_i \\ &= \begin{pmatrix} a \\ b \end{pmatrix}^T \begin{pmatrix} d \\ -c \end{pmatrix} \\ &= \begin{pmatrix} a \\ b \end{pmatrix}^T \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix} \begin{pmatrix} c \\ d \end{pmatrix} \\ &= u^T Jv. \end{aligned}$$

□

Given a symplectic manifold (M, ω) and a point $p \in M$, the dual to the tangent space $T_p M$ is called the cotangent space $T_p^\star M$. The disjoint union of all cotangent spaces is called the cotangent bundle $T^\star M$. Local coordinates on $T^\star M$ are of the form $(x, y) = (x_1, \ldots, x_n, y_1, \ldots, y_n)$, where $x = (x_1, \ldots, x_n)$ are the local coordinates on M and $y = (y_1, \ldots, y_n)$ are the coefficients that determine a 1-form on $T_x M$. In these local coordinates one can define a canonical 1-form on $T^\star M$ by

$$\lambda_0 = \sum_{i=1}^{n} y_i dx_i.$$

This gives rise to a well-defined global 1-form. Since

$$\begin{aligned} -d\lambda_0 &= -\sum_{i=1}^{n} dy_i \wedge dx_i \\ &= \sum_{i=1}^{n} dx_i \wedge dy_i \\ &= \omega_0, \end{aligned}$$

the cotangent bundle carries a canonical symplectic structure and thus can be regarded as a symplectic manifold in its own right. In this thesis, we will pay special attention to submanifolds of the cotangent bundle of the n-dimensional torus $\mathbb{T}^n = \mathbb{R}^n / \mathbb{Z}^n$. In this case, the cotangent bundle is a Cartesian product of the form $T^\star \mathbb{T}^n = \mathbb{T}^n \times \mathbb{R}^n$.

1.2 Symplectic Maps

Next, we are going to study maps between two symplectic manifolds.

Definition 1.3. A *symplectic map* $\varphi : (M_1, \omega_1) \to (M_2, \omega_2)$ is a smooth map that satisfies $\varphi^\star \omega_2 = \omega_1$. If φ is injective and if $\varphi(M_1)$ is a submanifold of M_2, then φ is called a *symplectic embedding*. If φ is a diffeomorphism, then φ is called a *symplectomorphism*.

Recall, that the pullback of ω_2 by φ is defined as

$$(\varphi^\star \omega_2)(u, v) = \omega_2(d\varphi(u), d\varphi(v)).$$

As Riemannian geometry is the study of transformations preserving the inner product, symplectic geometry is the study of transformations preserving the symplectic form. The first important theorem in symplectic geometry, which goes back to Darboux in 1882, is that locally all symplectic forms are the same.

Theorem 1.4 (Darboux's Theorem [Dar82])**.**
For any point p on a $2n$-dimensional symplectic manifold (M, ω) there exist an open neighbourhood $U \subseteq M$ of p, an open neighbourhood $V \subseteq \mathbb{R}^{2n}$ of the origin and a symplectomorphism $\varphi : (U, \omega) \to (V, \omega_0)$ such that $\varphi(p) = 0$.

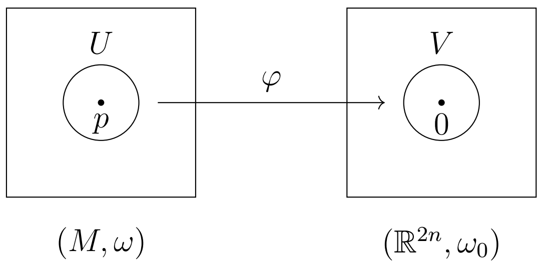


Figure 1.1: (M, ω) is locally symplectomorphic to $(\mathbb{R}^{2n}, \omega_0)$.

This result implies that there are no local invariants in symplectic geometry. This is in great contrast to Riemannian geometry where curvature is an invariant that can be determined locally. So the natural question that arises is: Can we find globally defined symplectic invariants? One of the first striking results in this direction, which lies at the root of symplectic geometry, is due to Gromov. He asked for the biggest radius of a ball that can be symplectically embedded into a given symplectic manifold. To illustrate the nontriviality of this question, he stated the famous Non-squeezing Theorem in 1985. Let

$$B^{2n}(r) = \left\{ (x, y) \in \mathbb{R}^{2n} \;\middle|\; \|x\|^2 + \|y\|^2 < \frac{r}{\pi} \right\},$$
$$Z^{2n}(s) = \left\{ (x, y) \in \mathbb{R}^{2n} \;\middle|\; x_1^2 + y_1^2 < \frac{s}{\pi} \right\}$$

denote the $2n$-dimensional open ball of radius $\sqrt{\frac{r}{\pi}}$ and the $2n$-dimensional open cylinder of radius $\sqrt{\frac{s}{\pi}}$, respectively. Gromov's Non-squeezing theorem states that one cannot symplectically embed $B^{2n}(r)$ into $Z^{2n}(s)$ unless the radius r of the ball is less than or equal to the radius s of the cylinder.

Theorem 1.5 (Gromov's Non-squeezing Theorem [Gro85])**.**
There exsists a symplectic embedding $\varphi : (B^{2n}(r), \omega_0) \to (Z^{2n}(s), \omega_0)$ if and only if $r \leq s$.

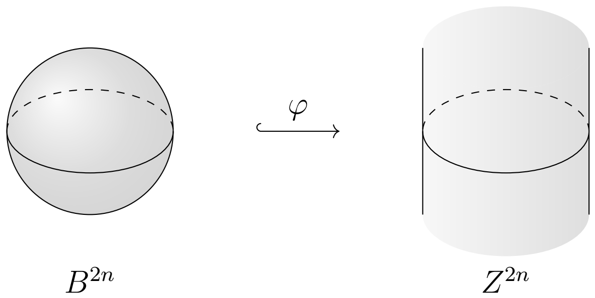


Figure 1.2: One cannot squeeze the ball into the cylinder.

The Non-squeezing Theorem indicates the rigidity of symplectic embeddings as compared to the flexibility of volume preserving diffeomorphisms. Just picture squeezing the ball into the cylinder by a volume-preserving transformation.

In dimension two, symplectomorphisms are precisely area-preserving diffeomorphisms but in dimension greater than two, it is much more restrictive for a map to be symplectic than to be volume-preserving. Physically speaking, the Non-squeezing Theorem says that if a collection of particles initially spread out all over the unit ball, then one cannot squeeze the collection into a statistical state in which the momentum and position in the x_1-y_1-direction spread out less than initially.

1.3 Symplectic Capacities

The invariant found by Gromov is called the ball packing width. Let the expression $\varphi : (M_1, \omega_1) \overset{s}{\hookrightarrow} (M_2, \omega_2)$ denote that the map φ is a symplectic embedding. Then the ball packing width is formally defined as follows.

Definition 1.6. The *ball packing width* of a $2n$-dimensional symplectic manifold (M, ω) is

$$g(M,\omega) = \sup\left\{ r \;\middle|\; \exists \varphi : \left(B^{2n}(r), \omega_0\right) \overset{s}{\hookrightarrow} (M,\omega) \right\}.$$

Instead of studying only one symplectic embedding of a ball of maximum radius, one can also study k symplectic embeddings of a ball with maximum radius such that the embeddings have pairwise disjoint images. The corresponding invariant is called the k-ball packing width.

Definition 1.7. The *k-ball packing width* of a $2n$-dimensional symplectic manifold (M, ω) is

$$g_k(M,\omega) = \sup\left\{ r \;\middle|\; \begin{array}{l} \exists \varphi_1, \ldots, \varphi_k : (B^{2n}(r), \omega_0) \overset{s}{\hookrightarrow} (M,\omega) \text{ with} \\ \varphi_i\left(B^{2n}(r)\right) \cap \varphi_j\left(B^{2n}(r)\right) = \emptyset \quad \forall 1 \leq i < j \leq k \end{array} \right\}.$$

The ball packing width and the k-ball packing width are symplectic invariants. They give information that can be used to distinguish symplectic manifolds. This led to the search for other symplectic invariants. The properties of these invariants were first axiomatized in 1994 by Ekeland and Hofer who introduced the notion of symplectic capacity [HZ12].

Definition 1.8. A *symplectic capacity* is a map

$$c : \{(M,\omega) \mid (M,\omega) \textit{ symplectic manifold}\} \to [0,\infty]$$

that satisfies the following properties:

1. *Monotonicity:* $(M_1, \omega_1) \overset{s}{\hookrightarrow} (M_2, \omega_2) \Rightarrow c(M, \omega_1) \leq c(M_2, \omega_2)$.
2. *Conformality:* $\forall \alpha \in \mathbb{R} \setminus \{0\} : \; c(M, \alpha\omega) = |\alpha| c(M, \omega)$.
3. *Nontriviality:* $c\left(B^{2n}(1), \omega_0\right) > 0$ and $c\left(Z^{2n}(1), \omega_0\right) < \infty$.

The nontriviality condition guarantees that, in dimension greater than two, volume is not a capacity. The search for symplectic capacities and techniques to calculate them are major areas of research in symplectic geometry. Although these invariants are quite easy to define, they are extremely difficult to calculate. For a computational approach, we will replace the ball by its prismification. By

$$P^{2n}(r) = \mathbb{T}^n \times \text{⊾}^n(r)$$

we denote the $2n$-dimensional open prism, which is the Cartesian product of the n-dimensional torus $\mathbb{T}^n = \mathbb{R}^n/\mathbb{Z}^n$ and the n-dimensional open standard simplex of side length r

$$\text{⊾}^n(r) = \left\{ x \in \mathbb{R}^n \;\middle|\; x_i > 0 \quad \forall i \in [n] \text{ and } \sum_{i=1}^{n} x_i < r \right\}.$$

The following theorem shows the relation between the ball B^{2n} and the prism P^{2n}.

Theorem 1.9 (Prismification [MMT00])**.**
For every $\varepsilon > 0$, there exist symplectic embeddings

$$\left(B^{2n}(r-\varepsilon), \omega_0\right) \overset{s}{\hookrightarrow} \left(P^{2n}(r), \omega_0\right) \overset{s}{\hookrightarrow} \left(B^{2n}(r), \omega_0\right).$$

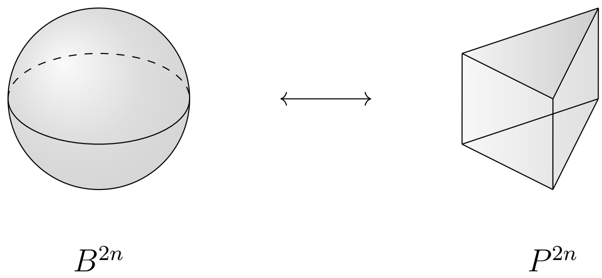


Figure 1.3: Prismification of the ball

Proof. We will start the proof by showing that $(P^{2n}(r), \omega_0)$ symplectically embeds into $(B^{2n}(r), \omega_0)$. To see this, we consider the map $\varphi : (P^{2n}(r), \omega_0) \to (B^{2n}(r), \omega_0)$

given by

$$\varphi(x_1,\ldots,x_n,y_1,\ldots,y_n)=\Bigg(\sqrt{\frac{y_1}{\pi}}\sin(2\pi x_1)\quad,\cdots,\quad\sqrt{\frac{y_n}{\pi}}\sin(2\pi x_n),$$
$$\sqrt{\frac{y_1}{\pi}}\cos(2\pi x_1),\quad,\cdots,\quad\sqrt{\frac{y_n}{\pi}}\cos(2\pi x_n)\Bigg).$$

It is clear, that φ is injective. First, we will show that the image of $P^{2n}(r)$ under φ is contained in $B^{2n}(r)$. For every point $(x_1,\ldots,x_n,y_1,\ldots,y_n)\in P^{2n}(r)=\mathbb{T}^n\times\triangle^n(r)$ we have

$$\begin{aligned}\|\varphi(x)\|^2+\|\varphi(y)\|^2&=\sum_{i=1}^n\left(\sqrt{\frac{y_i}{\pi}}\sin(2\pi x_i)\right)^2+\left(\sqrt{\frac{y_i}{\pi}}\cos(2\pi x_i)\right)^2\\&=\sum_{i=1}^n\frac{y_i}{\pi}\left(\sin^2(2\pi x_i)+\cos^2(2\pi x_i)\right)\\&=\frac{1}{\pi}\sum_{i=1}^n y_i\\&<\frac{r}{\pi}.\end{aligned}$$

Hence, $\varphi\left(P^{2n}(r)\right)\subseteq B^{2n}(r)$. Next, we will show that φ is symplectic, that is $\varphi^\star\omega_0=\omega_0$. We have

$$\begin{aligned}\varphi^\star\omega_0&=\sum_{i=1}^n\varphi^\star dx_i\wedge\varphi^\star dy_i\\&=\sum_{i=1}^n\left(2\pi\sqrt{\frac{y_i}{\pi}}\cos(2\pi x_i)dx_i+\frac{1}{2\sqrt{\pi y_i}}\sin(2\pi x_i)dy_i\right)\\&\qquad\qquad\wedge\left(-2\pi\sqrt{\frac{y_i}{\pi}}\sin(2\pi x_i)dx_i+\frac{1}{2\sqrt{\pi y_i}}\cos(2\pi x_i)dy_i\right)\\&=\sum_{i=1}^n\cos^2(2\pi x_i)dx_i\wedge dy_i-\sin^2(2\pi x_i)dy_i\wedge dx_i\\&=\sum_{i=1}^n dx_i\wedge dy_i\\&=\omega_0.\end{aligned}$$

We finish the proof by showing that $(B^{2n}(r-\varepsilon), \omega_0)$ symplectically embeds into $(P^{2n}(r), \omega_0)$ for arbitrary small $\varepsilon > 0$. For this purpose, we will construct two maps such that their composition will result in the desired symplectic embedding. The construction is shown in the following diagram:

$$\begin{array}{ccc} (B^{2n}(r-\varepsilon), \omega_0) & \xrightarrow{\tilde{\sigma_\varepsilon} \circ \psi} & (P^{2n}(r), \omega_0) \\ & {\scriptstyle \tilde{\sigma_\varepsilon}} \searrow \quad \nearrow {\scriptstyle \psi} & \\ & (B^{2n}(r) \cap A^{2n}(r), \omega_0) & \end{array}$$

We will now specify the maps and show that both are symplectic. Let us define the set

$$A^2(r) = B^2(r) \setminus \left\{ (x, y) \in \mathbb{R}^2 \mid x \geq 0 \ and\ y = 0 \right\}.$$

Geometrically, $A^2(r)$ represents a slit disc. Now consider an area preserving embedding $\sigma_\varepsilon : B^2(r-\varepsilon) \to A^2(r)$ such that $x^2 + y^2 \leq \alpha$ implies $(\sigma_\varepsilon(x, y))^2 \leq \alpha + \varepsilon$ for every $\alpha \geq 0$. A visualization of this map is shown in Figure 1.4.

Let $A^{2n}(r)$ denote the n-times Cartesian product of the set $A^2(r)$. Then the map

$$\tilde{\sigma_\varepsilon} : \underbrace{B^2(r-\varepsilon) \times \cdots \times B^2(r-\varepsilon)}_{n\ times} \to A^{2n}(r)$$

given by

$$\tilde{\sigma_\varepsilon}(x_1, y_1, \ldots, x_n, y_n) = (\sigma_\delta(x_1, y_1), \ldots, \sigma_\delta(x_n, y_n))$$

with $\delta = \frac{\varepsilon}{n\pi}$ symplectically embeds $B^{2n}(r-\varepsilon)$ into $B^{2n}(r) \cap A^{2n}(r)$. To see that the image of $B^{2n}(r-\varepsilon)$ under $\tilde{\sigma_\varepsilon}$ is contained in $B^{2n}(r)$, note that $(x, y) \in B^{2n}(r-\varepsilon)$ implies

$$\sum_{i=1}^{n} x_i^2 + y_i^2 < \frac{r-\varepsilon}{\pi}.$$

Hence, there exist $\alpha_1, \dots, \alpha_n \geq 0$ such that $x_i^2 + y_i^2 \leq \alpha_i$ for every $i \in [n]$ and

$$\sum_{i=1}^{n} \alpha_i < \frac{r - \varepsilon}{\pi}.$$

By definition of σ_δ, we have

$$\begin{aligned}\sum_{i=1}^{n} \left(\sigma_\delta(x_i, y_i)\right)^2 &\leq \sum_{i=1}^{n} (\alpha_i + \delta) \\ &= n\delta + \sum_{i=1}^{n} \alpha_i \\ &< n\frac{\varepsilon}{n\pi} + \frac{r-\varepsilon}{\pi} \\ &= \frac{r}{\pi}.\end{aligned}$$

Hence, $\tilde{\sigma_\varepsilon}\left(B^{2n}(r-\varepsilon)\right) \subseteq B^{2n}(r)$. Now we will show that $(B^{2n}(r) \cap A^{2n}(r), \omega_0)$ symplectically embeds into $(P^{2n}(r), \omega_0)$. For this purpose, we consider the map $\psi : (B^{2n}(r) \cap A^{2n}(r), \omega_0) \to (P^{2n}(r), \omega_0)$ given by

$$\psi(x_1, \dots, x_n, y_1, \dots, y_n) = \begin{pmatrix} \frac{1}{2\pi}\cot^{-1}\left(-\frac{x_1}{y_1}\right) & , \cdots , & \frac{1}{2\pi}\cot^{-1}\left(-\frac{x_n}{y_n}\right), \\ \pi\left(x_1^2 + y_1^2\right) & , \cdots , & \pi\left(x_n^2 + y_n^2\right) \end{pmatrix}.$$

Again, it is easy to see, that this map is injective. First, we will show that the image of $B^{2n}(r) \cap A^{2n}(r)$ under ψ is contained in $P^{2n}(r)$. On the one hand, for $(x, y) \in B^{2n}(r) \cap A^{2n}(r)$ the inequality

$$0 < \sum_{i=1}^{n} \pi(x_i^2 + y_i^2) = \pi\left(\|x\|^2 + \|y\|^2\right) < \pi\frac{r}{\pi} = r$$

holds. Therefore, $\psi\left(B^{2n}(r) \cap A^{2n}(r)\right) \subseteq \mathbb{R}^n \times \lrtriangle^n(r)$. On the other hand, for every $z \in \mathbb{R}$, the function $\cot^{-1}(z)$ is bounded by $0 < \cot^{-1}(z) < \pi$. Thus, $\psi\left(B^{2n}(r) \cap A^{2n}(r)\right) \subseteq \mathbb{T}^n \times \lrtriangle^n(r)$. It remains to show that ψ is symplectic. We have

$$\begin{aligned}
\psi^\star\omega_0 &= \psi^\star\left(\sum_{i=1}^n dx_i \wedge dy_i\right)\\
&= \sum_{i=1}^n \psi^\star dx_i \wedge \psi^\star dy_i\\
&= \sum_{i=1}^n \left(\frac{1}{2\pi y_i}\frac{1}{1+\left(\frac{x_i}{y_i}\right)^2}dx_i - \frac{x_i}{2\pi y_i^2}\frac{1}{1+\left(\frac{x_i}{y_i}\right)^2}dy_i\right) \wedge (2\pi x_i dx_i + 2\pi y_i dy_i)\\
&= \sum_{i=1}^n \frac{1}{1+\left(\frac{x_i}{y_i}\right)^2}dx_i \wedge dy_i - \frac{\left(\frac{x_i}{y_i}\right)^2}{1+\left(\frac{x_i}{y_i}\right)^2}dy_i \wedge dx_i\\
&= \sum_{i=1}^n dx_i \wedge dy_i\\
&= \omega_0.
\end{aligned}$$

Finally, the function composition $\tilde{\sigma}_\varepsilon \circ \psi$ symplectically embeds $(B^{2n}(r-\varepsilon), \omega_0)$ into $(P^{2n}(r), \omega_0)$, which completes the proof. □

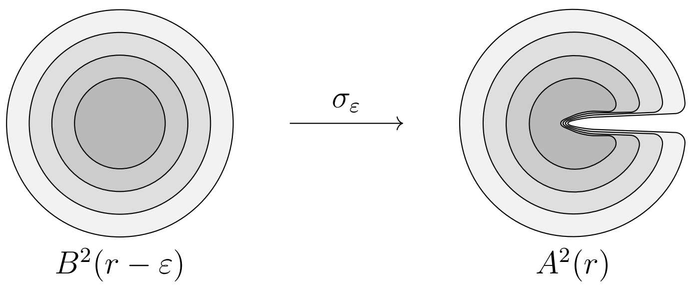


Figure 1.4: The map σ_ε sends the disc $B^2(r-\varepsilon)$ to the slightly larger slit disc $A^2(r)$.

By Theorem 1.9, embeddings of B^{2n} into a symplectic manifold give rise to embeddings of P^{2n} and vice versa. For this reason, there is no quantitative difference between looking at symplectic packings of B^{2n} or P^{2n}. The advantage of looking at packings of P^{2n} is, that when we restrict to symplectic manifolds that have first homology equal to $\mathbb{Z}^n$, it is possible to add the condition that the symplectic embeddings induce isomorphisms on the level of first homology. We call those maps

1-isomorphic. With this additional property we can define a slight modification of the k-ball packing width that allows for a computational approach.

Definition 1.10. A continuous map $\varphi : (M_1, \omega_1) \to (M_2, \omega_2)$ is called 1-*isomorphic* if it induces an isomorphism $\varphi_\star : H_1(M_1, \mathbb{Z}) \to H_1(M_2, \mathbb{Z})$ on the level of first homology.

Definition 1.11. The *k-simplex packing width* of a $2n$-dimensional symplectic manifold (M, ω) is

$$s_k(M,\omega) = \sup \left\{ r \;\middle|\; \begin{array}{l} \exists \varphi_1, \dots, \varphi_k : (P^{2n}(r), \omega_0) \xrightarrow[1-isomorphic]{s} (M, \omega) \text{ with} \\ \varphi_i \left(P^{2n}(r)\right) \cap \varphi_j \left(P^{2n}(r)\right) = \emptyset \quad \forall 1 \le i < j \le k \end{array} \right\}.$$

The k-simplex packing width has been introduced by Maley, Mastrangeli and Traynor [MMT00]. It is a symplectic invariant and satisfies a set of axioms analogous to the capacity axioms. Since we reduce the set of symplectic embeddings to those that are 1-isomorphic, the k-simplex packing width gives a lower bound on the k-ball packing width: $s_k \le g_k$. Theorem 1.12 shows the values of $g_k\left(P^4(1), \omega_0\right)$ and $s_k\left(P^4(1), \omega_0\right)$ for $k \in [20]$.

Theorem 1.12 ($g_k\left(P^4(1), \omega_0\right)$ versus $s_k\left(P^4(1), \omega_0\right)$ [MMT00])**.**

k	1	2	3	4	5	6	7	8	9	10
g_k	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{2}{5}$	$\frac{2}{5}$	$\frac{3}{8}$	$\frac{6}{17}$	$\frac{1}{3}$	$\frac{1}{\sqrt{10}}$
s_k	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{2}{5}$	$\frac{6}{17}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{3}{10}$
k	11	12	13	14	15	16	17	18	19	20
g_k	$\frac{1}{\sqrt{11}}$	$\frac{1}{\sqrt{12}}$	$\frac{1}{\sqrt{13}}$	$\frac{1}{\sqrt{14}}$	$\frac{1}{\sqrt{15}}$	$\frac{1}{4}$	$\frac{1}{\sqrt{17}}$	$\frac{1}{\sqrt{18}}$	$\frac{1}{\sqrt{19}}$	$\frac{1}{\sqrt{20}}$
s_k	$\frac{2}{7}$	$\frac{15}{56}$	$\ge \frac{6}{23}$	$\ge \frac{20}{79}$	$\ge \frac{1}{4}$	$\frac{1}{4}$	$\ge \frac{4}{17}$	$\ge \frac{3}{13}$	$\ge \frac{2}{9}$	$\ge \frac{21}{97}$

Biran proved in 1996 that $g_k(P^4(1), \omega) = \frac{1}{\sqrt{k}}$ for all $k \ge 9$ [Bir96]. For smaller values of k, the k-ball packing width can be computed using pseudoholomorphic curves [MP94]. Maley et. al obtained the simplex packing widths by using a computer program [MMT00]. The values of the k-simplex packing widths are always rational as we will see later.

For $k > 12$ the values of $s_k\left(P^4(1), \omega_0\right)$ are known lower bounds that are conjectured to be optimal. Our goal is to compute the exact values of $s_k\left(P^4(1), \omega_0\right)$ and $s_k\left(P^6(1), \omega_0\right)$ and also to find out what the corresponding explicit packings look like. Whereas algebraic geometry is a crucial tool to calculate the ball packing widths, the main tool to calculate the simplex packing widths is the following theorem.

Theorem 1.13 (Packing Theorem [MMT00])**.**
Let V be an open, connected subset of $\mathbb{R}^n$ with $H_1(V, \mathbb{Z}) = 0$. Then

$$s_k\left(\mathbb{T}^n \times V, \omega_0\right) = \sup \left\{ r \;\middle|\; \begin{array}{l} \exists A_1, \ldots, A_k \in \mathrm{GL_n}(\mathbb{Z}) \; \exists t_1, \ldots, t_k \in \mathbb{R}^n : \\ A_i\left(\measuredrightangle^n(r)\right) + t_i \subseteq V \quad \forall i \in [k] \\ \left(A_i\left(\measuredrightangle^n(r)\right) + t_i\right) \cap \left(A_j\left(\measuredrightangle^n(r)\right) + t_j\right) = \emptyset \\ \forall 1 \leq i < j \leq k \end{array} \right\}.$$

Here, $\mathrm{GL_n}(\mathbb{Z})$ denotes the set of all matrices in $\mathbb{Z}^{n \times n}$ that are invertible over $\mathbb{Z}$, together with matrix multiplication as the group operation.

The remainder of this chapter is devoted to proving Theorem 1.13. Along the proof of Theorem 1.13 we will construct a matrix $A_i \in \mathrm{GL_n}(\mathbb{Z})$ and a vector $t_i \in \mathbb{R}^n$ that corresponds to the symplectic embedding $\varphi_i : \left(P^{2n}(r), \omega_0\right) \to \left(\mathbb{T}^n \times V, \omega_0\right)$ for every $i \in [k]$. For these matrices $A_1, \ldots, A_n$ and vectors $t_1, \ldots, t_n$ we will then have to verify the containment condition

$$A_i\left(\measuredrightangle^n(r)\right) + t_i \subseteq V$$

for all $i \in [k]$ and the disjointness condition

$$\left(A_i\left(\measuredrightangle^n(r)\right) + t_i\right) \cap \left(A_j\left(\measuredrightangle^n(r)\right) + t_j\right) = \emptyset$$

for all $1 \leq i < j \leq k$.

The ingredients for verifying the containment condition are the notion of strongly exact symplectic embeddings and the subsequent Theorem 1.15. Both were introduced by Sikorav in 1989 [Sik89].

Definition 1.14. Let U be an open subset of $\mathbb{R}^n$. A symplectic embedding $\varphi : (\mathbb{T}^n \times U, \omega_0) \to (\mathbb{T}^n \times \mathbb{R}^n, \omega_0)$ is called *strongly exact* if the following two conditions hold:

1. The 1-form $\varphi^\star\lambda_0 - \lambda_0$ is exact on $\mathbb{T}^n \times U$, that is there exists a function $f : \mathbb{T}^n \times U \to \mathbb{R}$ such that $\varphi^\star\lambda_0 - \lambda_0 = df$.

2. The map $\varphi^\star : H^1(\mathbb{T}^n \times \mathbb{R}^n, \mathbb{R}) \to H^1(\mathbb{T}^n \times U, \mathbb{R})$ is equal to $i^\star$, where $i : \mathbb{T}^n \times U \to \mathbb{T}^n \times \mathbb{R}^n$ is the inclusion map.

Theorem 1.15 (Sikorav [Sik89]).
Let U, V be open subsets of $\mathbb{R}^n$. If there exist a strongly exact symplectic embedding $\varphi : (\mathbb{T}^n \times U, \omega_0) \to (\mathbb{T}^n \times V, \omega_0)$, then U is a subset of V.

The ingredients for verifying the disjointness condition are the notion of exact Lagrangian submanifolds and the subsequent Theorem 1.17, Lemma 1.18 and Lemma 1.19.

Definition 1.16. An n-dimensional submanifold L of a $2n$-dimensional symplectic manifold (M, ω) is called *Lagrangian* if $\omega(u, v) = 0$ for all $u, v \in T_pL$ at every point $p \in L$. The Lagrangian submanifold is called *exact* if there exists a function $f : L \to \mathbb{R}$ such that $\lambda_0\big|_L = df$.

Theorem 1.17 (Lalonde-Sikorav [LS91]).
If L, L' are two closed exact Lagrangian submanifolds of $\mathbb{T}^n \times \mathbb{R}^n$, then $L \cap L' \neq \emptyset$.

The connection between strongly exact symplectic embeddings and strongly exact Lagrangian submanifolds is that the latter are preserved under the former.

Lemma 1.18. *If L is an exact Lagrangian submanifold and if φ is a strongly exact symplectomorphism, then $\varphi(L)$ is an exact Lagrangian submanifold.*

Proof. Let L be an exact Lagrangian submanifold of M and let $\varphi : (M, d\lambda_0) \to (M, d\lambda_0)$ be a strongly exact symplectomorphism. Since L is exact, there exists a function $f : L \to \mathbb{R}$ such that $\lambda_0\big|_L = df$. Since φ is strongly exact, there exists a function $g : L \to \mathbb{R}$ such that $\varphi^\star\lambda_0 - \lambda_0 = dg$. By combination of these two equations we obtain

$$\begin{aligned}\varphi^\star\lambda_0 &= dg + \lambda_0 \\ &= dg + df \\ &= d(g+f).\end{aligned}$$

Therefore, the function $h := (g+f)\circ\varphi^{-1}$ satisfies $\lambda_0\big|_{\varphi(L)} = dh$, which shows that $\varphi(L)$ is an exact Lagrangian submanifold. □

Another useful fact about strongly exact symplectic embeddings is that they are preserved under conjugation. In the proof of Theorem 1.13 we will consider the conjugation by a translation in the fiber of $T^\star\mathbb{T}^n = \mathbb{T}^n \times \mathbb{R}^n$.

Lemma 1.19. *Let $\varphi : \mathbb{T}^n \times U \to \mathbb{T}^n \times \mathbb{R}^n$ be a strongly exact embedding and let $\tau_u : \mathbb{T}^n \times U \to \mathbb{T}^n \times \mathbb{R}^n$ given by*

$$\tau_u(x,y) = (x, y+u)$$

be the translation by u in the fiber for some $u \in \mathbb{R}^n$. Then $\tau_u^{-1}\circ\varphi\circ\tau_u$ is a strongly exact symplectic embedding.

Proof. Let $\varphi : \mathbb{T}^n \times U \to \mathbb{T}^n \times \mathbb{R}^n$ be a strongly exact embedding. By definition there exists a function $f : \mathbb{T}^n \times U \to \mathbb{R}$ such that $\varphi^\star\lambda_0 - \lambda_0 = df$. Furthermore, $\varphi^\star : H^1(\mathbb{T}^n \times \mathbb{R}^n, \mathbb{R}) \to H^1(\mathbb{T}^n \times U, \mathbb{R})$ is equal to $i^\star$. The second property implies

$$\begin{aligned}\left(\tau_u^{-1}\circ\varphi\circ\tau_u\right)^\star &= \tau_u^\star \circ \varphi^\star \circ \left(\tau_u^{-1}\right)^\star \\ &= \tau_u^\star \circ i^\star \circ \left(\tau_u^{-1}\right)^\star \\ &= \tau_u^\star \circ \left(\tau_u^{-1}\right)^\star \\ &= i^\star.\end{aligned}$$

Together with the first property, we deduce

$$\begin{aligned}\left(\tau_u^{-1} \circ \varphi \circ \tau_u\right)^\star \lambda_0 - \lambda_0 &= i^\star \lambda_0 - \lambda_0 \\ &= \varphi^\star \lambda_0 - \lambda_0 \\ &= df.\end{aligned}$$

Thus, $\tau_u^{-1} \circ \varphi \circ \tau_u$ is a strongly exact symplectic embedding. □

Now we have the necessary preliminaries to prove Theorem 1.13.

Proof of Theorem 1.13. Let V be an open, connected subset of $\mathbb{R}^n$ with $H_1(V, \mathbb{Z}) = 0$. Let

$$s^\star := s_k\left(\mathbb{T}^n \times V, \omega_0\right)$$

denote the left hand side in the equation from Theorem 1.13 and let

$$r^\star := \sup \left\{ r \;\middle|\; \begin{array}{l} \exists A_1, \ldots, A_k \in \mathrm{GL_n}(\mathbb{Z})\ \exists t_1, \ldots, t_k \in \mathbb{R}^n : \\ A_i\left(\triangle^n(r)\right) + t_i \subseteq V \quad \forall i \in [k] \\ \left(A_i\left(\triangle^n(r)\right) + t_i\right) \cap \left(A_j\left(\triangle^n(r)\right) + t_j\right) = \emptyset \\ \forall 1 \leq i < j \leq k \end{array} \right\}$$

denote the right hand side in the equation from Theorem 1.13.

In the first part of the proof, we show that $r^\star$ is less than or equal to $s^\star$. Suppose that $A_1, \ldots, A_k \in \mathrm{GL_n}(\mathbb{Z})$ and $t_1, \ldots, t_k \in \mathbb{R}^n$ are the corresponding matrices and vectors satisfying

$$A_i\left(\triangle^n(r^\star)\right) + t_i \subseteq V$$

for all $i \in [k]$ and

$$\left(A_i\left(\triangle^n(r^\star)\right) + t_i\right) \cap \left(A_j\left(\triangle^n(r^\star)\right) + t_j\right) = \emptyset$$

for all $1 \leq i < j \leq k$. Let us define the map $\varphi_{A_i,t_i} : (P^{2n}(r^\star), \omega_0) \to (\mathbb{T}^n \times V, \omega_0)$

given by

$$\varphi_{A_i,t_i}(x,y) = \left(\left(A_i^{-1}\right)^T x, A_i y + t_i\right)$$

for every $i \in [k]$. Clearly, these maps are 1-isomorphic, injective and have pairwise disjoint images. If we show that these maps are also symplectic, they satisfy all conditions from the definition of the simplex-packing width and thus give the lower bound $r^\star$ on $s^\star$.

The Jacobian matrix of φ_{A_i,t_i} is given by

$$d\varphi_{A_i,t_i} = \begin{pmatrix} \left(A_i^{-1}\right)^T & 0 \\ 0 & A_i \end{pmatrix}.$$

Hence, we have the identity

$$\left(d\varphi_{A_i,t_i}\right)^T J d\varphi_{A_i,t_i} = \begin{pmatrix} A_i^{-1} & 0 \\ 0 & A_i^T \end{pmatrix} \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix} \begin{pmatrix} \left(A_i^{-1}\right)^T & 0 \\ 0 & A_i \end{pmatrix} = J.$$

Together with Lemma 1.2 we deduce that

$$\begin{aligned} \varphi^\star_{A_i,t_i}\omega_0(u,v) &= \omega_0\left(d\varphi_{A_i,t_i}u, d\varphi_{A_i,t_i}v\right) \\ &= \left(d\varphi_{A_i,t_i}u\right)^T J d\varphi_{A_i,t_i}v \\ &= u^T d\varphi^T_{A_i,t_i} J d\varphi_{A_i,t_i} v \\ &= u^T J v \\ &= \omega_0(u,v) \end{aligned}$$

for all $u, v \in \mathbb{R}^{2n}$.

In the second part of the proof, we will show that $s^\star$ is less than or equal to $r^\star$. Suppose that $\varphi_1, \ldots, \varphi_k$ are the corresponding 1-isomorphic symplectic embeddings from $(P^{2n}(s^\star), \omega_0)$ into $(\mathbb{T}^n \times V, \omega_0)$ having pairwise disjoint images. For each map φ_i we will now construct an associating matrix $A_i \in \mathrm{GL_n}(\mathbb{Z})$ and an associating vector $t_i \in \mathbb{R}^n$ that satisfy the conditions of Theorem 1.13.

First, we will construct the matrices $A_1, \ldots, A_k \in \mathrm{GL_n}(\mathbb{Z})$. Since the map φ_i is 1-isomorphic, it induces an isomorphism

$$\varphi_{i\star} : H_1(P^{2n}(s^\star), \mathbb{Z}) \to H_1(\mathbb{T}^n \times V, \mathbb{Z})$$

on the level of first homology. By combining the universal coefficient theorem for cohomology and the five lemma one can show that the induced map on the level of first cohomology

$$\varphi_i^\star : H^1(\mathbb{T}^n \times V, \mathbb{Z}) \to H^1(P^{2n}(s^\star), \mathbb{Z})$$

is also an isomorphism. Since $H_1(⊾^n(s^\star), \mathbb{Z}) = 0$ and $H_1(V, \mathbb{Z}) = 0$, the Künneth theorem implies $H_1(P^{2n}(s^\star), \mathbb{Z}) = H_1(\mathbb{T}^n \times ⊾^n(s^\star), \mathbb{Z}) = H_1(\mathbb{T}^n, \mathbb{Z})$ and $H_1(\mathbb{T}^n \times V, \mathbb{Z}) = H_1(\mathbb{T}^n, \mathbb{Z})$, respectively. By the universal coefficient theorem of cohomology we have

$$H^1(\mathbb{T}^n, \mathbb{Z}) = \mathrm{Hom}(H_1(\mathbb{T}^n), \mathbb{Z}),$$

which shows $\varphi_i^\star \in \mathrm{Aut}(H^1(\mathbb{T}^n, \mathbb{Z}))$. We choose a fixed identification of the automorphism group $\mathrm{Aut}(H^1(\mathbb{T}^n, \mathbb{Z}))$ with the general linear group $\mathrm{GL_n}(\mathbb{Z})$. Under this identification we define the matrix $A_i \in \mathrm{GL_n}(\mathbb{Z})$ as $\varphi_i^\star \in \mathrm{Aut}(H^1(\mathbb{T}^n, \mathbb{Z}))$ for every $i \in [k]$.

Next, we construct the vectors $t_1, \ldots, t_k \in \mathbb{R}^n$. Consider the 1-form $\varphi_i^\star \lambda_0 - \lambda_0$. This 1-form is closed, because

$$\begin{aligned} d(\varphi_i^\star \lambda_0 - \lambda_0) &= \varphi_i^\star d\lambda_0 - d\lambda_0 \\ &= \varphi_i^\star \omega_0 - \omega_0 \\ &= 0, \end{aligned}$$

where the last equality holds due to φ_i being symplectic. Thus, $\varphi_i^\star \lambda_0 - \lambda_0$ represents a cohomology class in $H^1(P^{2n}(s^\star), \mathbb{R})$ and can be written in its standard basis:

$$\varphi_i^\star \lambda_0 - \lambda_0 = \sum_{j=1}^n a_j dx_j.$$

We define the vector t_i as the coefficient vector $t_i = (a_1 \cdots a_n)^T \in \mathbb{R}^n$ for every $i \in [k]$. Now we will show that these matrices $A_1, \ldots, A_k$ and vectors $t_1, \ldots, t_k$ indeed satisfy the containment condition and disjointness condition of Theorem 1.13. We start with the former.

Consider the map

$$\psi_i = \varphi_i \circ \varphi_{A_i,t_i}^{-1} : \mathbb{T}^n \times A_i(⊾^n(s^\star)) + t_i \to \mathrm{Im}(\varphi_i) \subseteq \mathbb{T}^n \times V,$$

where the map φ_{A_i,t_i} is defined as in the first part of the proof. If we show that this map is strongly exact, then Theorem 1.15 implies that $A_i(⊾^n(s^\star)) + t_i$ is contained in V.

On the one hand, we have

$$\phi_i^\star \lambda_0 - \lambda_0 = \sum_{j=1}^n a_j dx_j = t_i dx.$$

Thus, there exists a function $f : P^{2n}(s^\star) \to \mathbb{R}$ such that $\phi_i^\star \lambda_0 - \lambda_0 - t_i dx = df$. By rearranging this equation to $\varphi^\star \lambda_0 = \lambda_0 + t_i dx + df$ we can perform the following calculation:

$$\begin{aligned}
\psi_i^\star \lambda_0 &= \left(\varphi_i \circ \varphi_{A_i,t_i}^{-1}\right)^\star \lambda_0 \\
&= \left(\varphi_{A_i,t_i}^{-1}\right)^\star \circ \varphi_i^\star \lambda_0 \\
&= \left(\varphi_{A_i,t_i}^{-1}\right)^\star (\lambda_0 + t_i dx + df) \\
&= \left(\varphi_{A_i,t_i}^{-1}\right)^\star \lambda_0 + \left(\varphi_{A_i,t_i}^{-1}\right)^\star t_i dx + \left(\varphi_{A_i,t_i}^{-1}\right)^\star df \\
&= \lambda_0 + d\left(\left(\varphi_{A_i,t_i}^{-1}\right)^\star t_i x + \left(\varphi_{A_i,t_i}^{-1}\right)^\star f\right).
\end{aligned}$$

Hence, the function $g : P^{2n}(s^\star) \to \mathbb{R}$ given by $g := \left(\varphi_{A_i,t_i}^{-1}\right)^\star t_i x + \left(\varphi_{A_i,t_i}^{-1}\right)^\star f$ satisfies

$\psi_i^\star \lambda_0 - \lambda_0 = dg$. On the other hand, we have

$$\psi_i^\star = \left(\varphi_{A_i,t_i}^{-1}\right)^\star \circ \varphi_i^\star = i^\star.$$

Hence, ψ_i satisfies both properties of a strongly exact symplectic embedding. In consequence of Theorem 1.15, the matrices $A_1, \ldots, A_k$ and vectors $t_1, \ldots, t_k$ satisfy the containment condition.

We finish this proof by showing that the matrices $A_1, \ldots, A_k$ and vectors $t_1, \ldots, t_k$ also satisfy the disjointness condition. Suppose there exists indices $1 \leq i < j \leq k$ such that the sets $A_i\left(⊾^n(s^\star)\right) + t_i$ and $A_j\left(⊾^n(s^\star)\right) + t_j$ are not disjoint. Hence, there exists an element $c \in \left(A_i\left(⊾^n(s^\star)\right) + t_i\right) \cap \left(A_j\left(⊾^n(s^\star)\right) + t_j\right)$. Consider the translation by c in the fiber $\tau_c : \mathbb{T}^n \times \mathbb{R}^n \to \mathbb{T}^n \times \mathbb{R}^n$ given by

$$\tau_c(x, y) = (x, y + c).$$

Then the sets

$$L_i := \tau_c^{-1} \circ \psi_i \circ \tau_c\left(\mathbb{T}^n \times \{0\}\right),$$
$$L_j := \tau_c^{-1} \circ \psi_j \circ \tau_c\left(\mathbb{T}^n \times \{0\}\right)$$

are exact Lagrangian submanifolds in $\mathbb{T}^n \times \mathbb{R}^n$ as a consequence of Lemma 1.18 and Lemma 1.19. Theorem 1.17 implies that the intersection of L_i and L_j is nonempty. Hence, the intersection of $\tau_c(L_i)$ and $\tau_c(L_j)$ is also nonempty. However, since

$$\tau_c(L_i) = \psi_i \circ \tau_c\left(\mathbb{T}^n \times \{0\}\right) = \psi_i\left(\mathbb{T}^n \times \{c\}\right) \subseteq \mathrm{Im}(\varphi_i),$$
$$\tau_c(L_j) = \psi_j \circ \tau_c\left(\mathbb{T}^n \times \{0\}\right) = \psi_j\left(\mathbb{T}^n \times \{c\}\right) \subseteq \mathrm{Im}(\varphi_j)$$

this is a contradiction to φ_i and φ_j having disjoint images. Therefore, the matrices $A_1, \ldots, A_k \in \mathrm{GL_n}(\mathbb{Z})$ and vectors $t_1, \ldots, t_k \in \mathbb{R}^n$ satisfy both the containment condition and the disjointness condition and thus give the lower bound $s^\star$ on $r^\star$.

□

The meaning of Theorem 1.13 is, that for symplectic manifolds of the form $\mathbb{T}^n \times V$, where V is an open connected subset of $\mathbb{R}^n$ with first homology equal to zero, we can compute the k-simplex packing width $s_k\left(\mathbb{T}^n \times V, \omega_0\right)$ by computing an optimal packing of V by copies of $\llcorner^n(r)$ under integral affine transformations while maximizing the sidelength r. This not only reduces the dimension of the problem space from $2n$ to n but also converts the calculation of the k-simplex packing width into a classical combinatoral packing problem. In the next chapter we will build up the necessary background knowledge in combinatoral optimization for being able to solve this problem.

Chapter 2

Foundations of Combinatorial Optimization

In this chapter we will give a short introduction to combinatorial optimization. It is mainly based on the books of Alexander Schrijver [Sch98] and Vandenberghe and Boyd [VB96].

2.1 Linear Optimization

Linear optimization is a major field in operations research and concerns the problem of maximizing a linear function over a polyhedron.

Definition 2.1. A *linear program* is of the form

$$\begin{aligned} \max \quad & c^T x \\ & Ax \leq b \\ & x \in \mathbb{R}^n \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$ is a matrix and $b \in \mathbb{R}^m$ and $c \in \mathbb{R}^n$ are vectors.

Geometrically, the set of feasible solutions describes a polyhedron

$$\mathcal{P} = \{x \in \mathbb{R}^n : Ax \leq b\}.$$

Every row $a_i^T x \leq b_i$ for $i \in [m]$ of the system of linear inequalities $Ax \leq b$ represents a half-space and $\mathcal{P}$ is formed by the intersection of all those half-spaces. Maximizing the objective function $c^T x$ over $\mathcal{P}$ corresponds to shifting the hyperplane $\left\{x \in \mathbb{R}^n \;\middle|\; c^T x = 0\right\}$ along the direction of the vector c as long as it contains points in $\mathcal{P}$. The optimal solution $x^\star$ is then given by an intersection point as visualized in Figure 2.1.

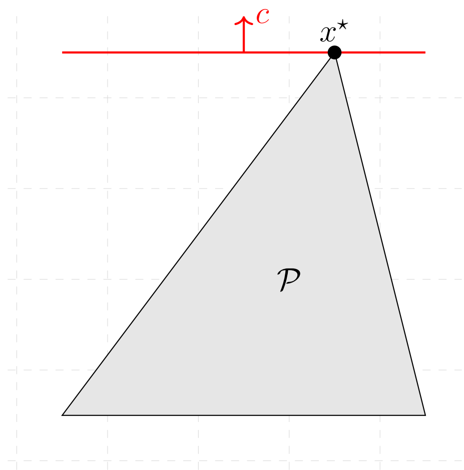


Figure 2.1: Geometric interpretation of a linear program

Of course, there could also happen to be several optimal solutions (in case c is parallel to a vector a_i for some $i \in [m]$), no solution at all (in case $\mathcal{P}$ is empty), or the optimum value might be infinity (in case c points into a direction in which $\mathcal{P}$ is unbounded).

Solving a linear problem to arbitrary precision can be done in polynomial time using the ellipsoid method or the interior point method. However, for many real world problems the simplex algorithm is the method of choice, even though its runtime is exponential in theory. The simplex algorithm was designed by Dantzig in 1951 and has a strong geometric intuition [Dan51]. It starts at one vertex of the polytope and runs along its edges until it finds a vertex that is optimal for the problem. The simplex algorithm also solves another linear problem that is given by

$$\begin{aligned} \min \quad & y^T b \\ & y^T A = c^T \\ & y \geq 0 \\ & y \in \mathbb{R}^m. \end{aligned}$$

The problems are called primal and dual pair. The dual problem of the dual problem results in the primal problem again. For any feasible solution x to the primal problem and any feasible solution y to the dual problem, the inequality

$$c^T x = \left(y^T A\right) x = y^T (Ax) \leq y^T b$$

holds. This means, each feasible solution of the dual problem gives an upper bound on the optimal value of the primal problem and each feasible solution of the primal problem gives a lower bound on the optimal value of the dual problem. This property is referred to as weak duality. It can even be sharpened: If one of the problems has an optimal solution, then so has the other and both values coincide. This property is referred to as strong duality.

Theorem 2.2 (Strong Duality Theorem [Neu47])**.**
Let $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ and $c \in \mathbb{R}^n$. Then

$$\begin{aligned} \max \quad & c^T x & \quad = \quad & \min \quad & y^T b \\ & Ax \leq b & & & y^T A = c^T \\ & x \in \mathbb{R}^n & & & y \geq 0 \\ & & & & y \in \mathbb{R}^m \end{aligned}$$

provided that both problems have feasible solutions.

There is a simple criterion to check whether two given feasible solutions are optimal. If x is feasible for the primal problem and if y is feasible for the dual problem, then both solutions are optimal if and only if $y^T(b - Ax) = 0$. This criterion is called complementary slackness condition. The name becomes clear by writing the primal problem in a different way using a slack variable s.

$$
\begin{array}{ll} \max & c^T x \\ & Ax \le b \\ & x \in \mathbb{R}^n \end{array}
\quad = \quad
\begin{array}{ll} \max & c^T x \\ & Ax + s = b \\ & s \ge 0 \\ & x \in \mathbb{R}^n, \quad s \in \mathbb{R}^m \end{array}
$$

The complementary slackness condition now becomes $y^T s = 0$. So, if the i^{th} component of the dual variable y is not zero, then the i^{th} component of the primal slack variable s must be equal to zero and vice versa.

2.2 Mixed Integer Linear Optimization

Many real world problems cannot be modeled using continuous variables only but require some of the variables to be integer.

Definition 2.3. An *integer linear program* is of the form

$$
\begin{array}{ll} \max & c^T x \\ & Ax \le b \\ & x \in \mathbb{Z}^n \end{array}
$$

where $A \in \mathbb{R}^{m \times n}$ is a matrix and $b \in \mathbb{R}^m$ and $c \in \mathbb{R}^n$ are vectors. If only some of the variables are integer, we speak of a *mixed integer linear program*.

For convenience, we will consider the purely integer case from now on but the subsequent results also apply to the mixed integer case.

Restricting the variables to integers has a huge impact on the tractability of the problem. In contrast to linear programs, polynomial time algorithms for integer linear programs are neither known nor believed to exist, since the problem is $\mathcal{NP}$-complete [GJ79].

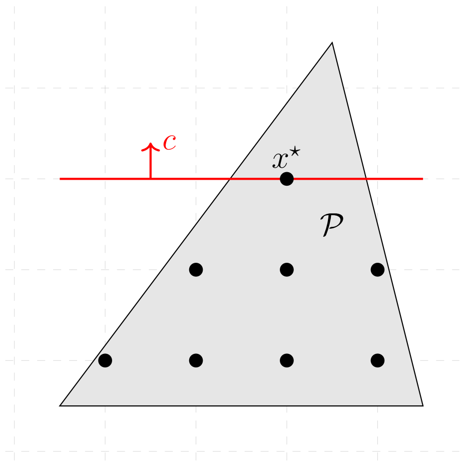


Figure 2.2: Geometric interpretation of an integer linear program

By comparison of Figure 2.1 and Figure 2.2 one can see that the optimal values of the two problems can differ quite a lot, even though the underlying polyhedron $\mathcal{P} = \{x \in \mathbb{R}^n : Ax \leq b\}$ is the same. There are two main solution strategies for solving integer linear programs: the cutting plane method and the branch-and-bound method. We will briefly explain the two, since they are part of our computer algorithm for calculating the simplex packing widths.

The cutting plane method was first utilized in 1954 by Dantzig, Fulkerson and Johnson to solve a large instance of the Traveling Salesman Problem [DFJ54]. Without knowledge of this result, Gomory developed the first general cutting plane approach [Gom58] in 1958. The idea is to first solve the linear relaxation of the integer linear program, which is the problem that arises by removing the integrality constraint of each variable. In general, the corresponding solution is not integer but delivers an upper bound on the solution of the integer linear program. The goal is to gradually improve this upper bound by adding further inequalities to the system. These inequalities must be valid for all feasible solutions of the integer linear program but not valid for the current optimal solution of the linear relaxation. Geometrically, they cut off the optimal solution of the linear relaxation, which is the reason they are called cutting planes. One possible cutting plane for the previous integer linear problem is shown in Figure 2.3.

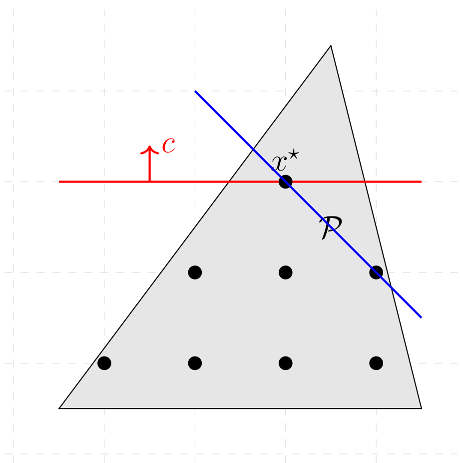


Figure 2.3: Cutting plane for an integer linear program

In most cases cutting planes on their own do not yield promising results. Instead, they are combined with the branch-and-bound method. The branch-and-bound method for solving integer linear programs was first proposed by Land and Doig in 1960 [LD60]. Just like the cutting plane approach, the branch-and-bound method starts by calculating the optimal solution $\hat{x}$ of the linear relaxation. The corresponding optimal value gets stored as the global upper bound. If $\hat{x}$ is integer, the problem is solved, otherwise there exists a non-integer component $\hat{x}_i$. In this case a branching on x_i is performed by creating two subproblems

$$\begin{aligned} \max \quad & c^T x \\ & Ax \leq b \\ & x_i \leq \lfloor \hat{x}_i \rfloor \\ & x \in \mathbb{Z}^n \end{aligned} \qquad \text{and} \qquad \begin{aligned} \max \quad & c^T x \\ & Ax \leq b \\ & x_i \geq \lceil \hat{x}_i \rceil \\ & x \in \mathbb{Z}^n. \end{aligned}$$

In the next step, the linear relaxation of both subproblems is solved. Now for each subproblem, one of three things can happen:

1. There exist no feasible solutions.
2. The optimal solution is integer.
3. The optimal solution is not integer.

In the first case the problem is discarded. In the second case the optimal value gets stored as the new global lower bound. In the third case the optimal value gets stored as the local upper bound and two new subproblems are created by branching on one of the non-integer variables. The branch-and-bound algorithm terminates prematurely if an integer solution with optimal value equal to the global upper bound has been detected. Otherwise, it terminates when there are no open subproblems left. The optimal solution is then given by the global lower bound. If no global lower bound exists, the problem is infeasible.

For example, consider the integer linear program from Figure 2.2:

$$\begin{aligned} \max \quad & \begin{pmatrix} 0 \\ 1 \end{pmatrix}^T x \\ & \begin{pmatrix} -8 & 6 \\ 8 & 2 \\ 0 & 2 \end{pmatrix} x \leq \begin{pmatrix} -1 \\ 37 \\ -1 \end{pmatrix} \\ & x \in \mathbb{Z}^2. \end{aligned}$$

The corresponding branch-and-bound search tree is shown in Figure 2.4. The optimal solution of the linear relaxation P_0 is given by $\begin{pmatrix} \frac{7}{2} \\ \frac{9}{2} \end{pmatrix}$ with optimal value $\frac{9}{2}$. Branching on the variable x_1 yields two new subproblems P_1 and P_2. Subproblem P_1 has optimal solution $\begin{pmatrix} 3 \\ \frac{23}{6} \end{pmatrix}$ with optimal value and new local upper bound $\frac{23}{6}$. Subproblem P_2 has optimal solution $\begin{pmatrix} 4 \\ \frac{5}{2} \end{pmatrix}$ with optimal value and new local upper bound $\frac{5}{2}$. Branching on the variable x_2 in P_1 yields the subproblems P_3 and P_4. Subproblem P_3 has an integer optimal solution $\binom{3}{3}$ with optimal value 3 that is stored as the new global lower bound. Subproblem P_4 is infeasible. The only open problem left is P_2 that is fathomed due to the global lower bound exceeding the local upper bound.

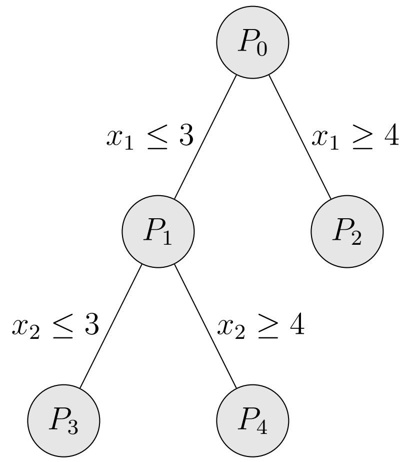


Figure 2.4: Branch-and-bound search tree for an integer linear program

2.3 Quadratically Constrained Quadratic Optimization

If we replace the linear objective function by a quadratic objective function, we speak about a quadratic program. If we additionally allow quadratic functions in the constraints, we call this a quadratically constrained quadratic program.

Definition 2.4. A *quadratically constrained quadratic program* is of the form

$$\begin{aligned} \min \quad & x^T C x + 2c^T x \\ & x^T A_i x + 2a_i^T x = b_i \quad \forall i \in [m] \\ & x \in \mathbb{R}^n \end{aligned}$$

where $C, A_1, \dots, A_m \in \mathbb{S}^n$, $c, a_1, \dots, a_m \in \mathbb{R}^n$ and $b_1, \dots, b_m \in \mathbb{R}$.

Here, $\mathbb{S}^n$ denotes the set of symmetric matrices in $\mathbb{R}^{n \times n}$. Quadratically constrained quadratic programs include linear programs as a special case by taking

$C = 0$ and $A_i = 0$ for $i \in [m]$. Furthermore, they include mixed binary linear programs. This can be seen by modeling the condition $x_i \in \{0, 1\}$ by using the quadratic expression $x_i(x_i - 1) = 0$. Since mixed binary linear optimization is $\mathcal{NP}$-hard in general, so is quadratically constrained quadratic programming.

If the matrices $C, A_1, \ldots, A_m$ are positive semidefinite, the problem can be solved in polynomial time with the ellipsoid method [KTK80]. In this case, the problem is convex and the feasible region is an intersection of m ellipsoids. Sahni proved that the problem is $\mathcal{NP}$-hard if one of the matrices is negative definite [Sah74]. Pardalos and Vavasis sharpened the result and showed that even one negative eigenvalue makes the problem $\mathcal{NP}$-hard [PV90].

So in general, quadratically constrained quadratic optimization problems are non-convex. The difficulty of non-convex optimization problems consists in the possibility of having several local minima that a solver may interpret as a global minimum. One main strategy to solve non-convex problems is to find a tight convex relaxation that provides a lower bound for the optimal solution of the original problem. One approach to relax non-convex quadratically constrained quadratic problems uses semidefinite programming. First, note that every inhomogeneous quadratically constrained quadratic program can be homogenized to

$$\begin{aligned}
\min \quad & \begin{pmatrix} x \\ t \end{pmatrix}^T \begin{pmatrix} C & c \\ c^T & 0 \end{pmatrix} \begin{pmatrix} x \\ t \end{pmatrix} \\
& \begin{pmatrix} x \\ t \end{pmatrix}^T \begin{pmatrix} A_i & a_i \\ a_i^T & 0 \end{pmatrix} \begin{pmatrix} x \\ t \end{pmatrix} = b_i \quad \forall i \in [m] \\
& \begin{pmatrix} x \\ t \end{pmatrix}^T \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ t \end{pmatrix} = 1 \\
& \begin{pmatrix} x \\ t \end{pmatrix} \in \mathbb{R}^{n+1}
\end{aligned}$$

by introducing a new variable $t \in \mathbb{R}$.

Both the number of variables and the number of constraints, increase by one.

The last constraint ensures that $t^2 = 1$. If in the optimal solution t takes the value $t = 1$, then the optimal solution to the original problem is x. If in the optimal solution t takes the value $t = -1$, then the optimal solution to the original problem is $-x$. Hence, without loss of generality, we can assume the quadratically constrained quadratic program to be of the form

$$\begin{aligned} \min \quad & x^T C x \\ & x^T A_i x = b_i \quad \forall i \in [m] \\ & x \in \mathbb{R}^n. \end{aligned}$$

To derive the semidefinite relaxation of this program, we observe that

$$\begin{aligned} x^T C x &= \sum_{i=1}^n \sum_{j=1}^n x_i C_{ij} x_j \\ &= \sum_{i=1}^n \sum_{j=1}^n C_{ij} x_i x_j \\ &= \operatorname{tr}\left(C x x^T\right) \\ &= \langle C, x x^T \rangle. \end{aligned}$$

Here, $\langle A, B \rangle = \operatorname{tr}\left(A^T B\right) = \sum_{i=1}^n \sum_{j=1}^m A_{ij} B_{ij}$ denotes the Frobenius inner product of the two matrices $A, B \in \mathbb{R}^{m \times n}$. The same equation holds true when replacing the matrix C by any of the constraint matrices A_i for $i \in [m]$. So both the objective function and the constraints are linear in the matrix xx^T. This procedure is often referred to as "lifting" the variable $x \in \mathbb{R}^n$ into the space $\mathbb{S}^n$. We introduce a new variable $X \in \mathbb{S}^n$ and note that the identity $X = xx^T$ is equivalent to $\operatorname{rank}(X) = 1$ and $X \succeq 0$. By $X \succeq 0$ we denote the matrix X to be positive semidefinite, that is $v^T X v \geq 0$ for all $v \in \mathbb{R}^n$. Now we get the following equivalent formulation of the homogeneous quadratically constrained quadratic program:

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ & \langle A_i, X \rangle = b_i \quad \forall i \in [m] \\ & X \in \mathbb{S}^n \\ & X \succeq 0 \\ & \operatorname{rank}(X) = 1. \end{aligned}$$

By dropping the non-convex rank-one constraint, we obtain the desired semidefinite relaxation. Since the minimum is now taken over a possibly larger set, the optimal value of this semidefinite relaxation does not necessarily coincide with the original solution but yields a lower bound. In the next section we will give more insight into semidefinite optimization.

2.4 Semidefinite Optimization

Definition 2.5. A *semidefinite program* is of the form

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ & \langle A_i, X \rangle = b_i \quad \forall i \in [m] \\ & X \in \mathbb{S}^n \\ & X \succeq 0, \end{aligned}$$

where $C, A_1, \ldots, A_m \in \mathbb{S}^n$ and $b_1, \ldots, b_m \in \mathbb{R}$.

As seen in the previous section, every quadratically constrained quadratic program can be relaxed to a semidefinite program.

Furthermore, every linear program

$$\begin{aligned} \min \quad & c^T x \\ & Ax = b \\ & x \geq 0 \\ & x \in \mathbb{R}^n \end{aligned}$$

can be written as a semidefinite program by choosing $C = diag(c)$ and $A_i = diag(a_i)$, where a_i is the i^{th} row of the matrix A. Note that we consider the dual form of the linear program with renamed variables in order to be conform to the notation of the semidefinite program. The association $X = diag(x)$ can be modeled by the additional equations $\langle E_{ij}, X \rangle = 0$ for $1 \leq i < j \leq n$. Here, the matrices E_{ij} for $1 \leq i \leq j \leq n$ are the standard basis of $\mathbb{S}^n$. They are defined as

$$E_{ij} = \frac{1}{2} \left(e_i e_j^T + e_j e_i^T \right),$$

where $e_1 = (1\ 0 \cdots 0)^T, \ldots, e_n = (0\ \cdots 0\ 1)^T$ denote the standard basis of $\mathbb{R}^n$.

Analogously to linear optimization, one can define a dual semidefinite program as

$$\begin{aligned} \max \quad & y^T b \\ & \sum_{i=1}^{m} y_i A_i \preceq C \\ & y \in \mathbb{R}^m, \end{aligned}$$

where for any two matrices $A, B \in \mathbb{S}^n$, the expression $A \preceq B$ means $B - A \succeq 0$. The dual problem of the dual problem results in the primal problem again. Also, weak duality holds: If X is feasible for the primal problem and y is feasible for the dual problem, then

$$\begin{aligned}\langle C, X\rangle - y^T b &= \langle C, X\rangle - \sum_{i=1}^{m} y_i b_i \\ &= \langle C, X\rangle - \sum_{i=1}^{m} y_i \langle A_i, X\rangle \\ &= \langle C, X\rangle - \langle \sum_{i=1}^{m} y_i A_i, X\rangle \\ &= \langle C - \sum_{i=1}^{m} y_i A_i, X\rangle \\ &\geq 0.\end{aligned}$$

Here, the last inequality holds true because the inner product of two positive semidefinite matrices is always greater or equal to zero, which can be seen using spectral decomposition.

In contrast to linear programs, however, not every semidefinite problem satisfies strong duality. For this, one requires one problem to have a strictly feasible solution and to be bounded, which is referred to as Slater's condition. A feasible solution X of the primal problem is called strictly feasible if $X \succ 0$ and a feasible solution y of the dual problem is called strictly feasible if $\sum_{i=1}^{m} y_i A_i \prec C$. By $X \succ 0$ we denote the matrix X to be positive definite, that is $v^T X v > 0$ for all $v \in \mathbb{R}^n$ with $v \neq 0$.

Theorem 2.6 (Slater's Condition [Sla59])**.**
Let $C, A_1, \ldots, A_m \in \mathbb{S}^n$ and let $b_1, \ldots, b_m \in \mathbb{R}$. Consider the primal semidefinite program

$$\begin{aligned}p^\star = \min \quad & \langle C, X\rangle \\ & \langle A_i, X\rangle = b_i \quad \forall i \in [m] \\ & X \in \mathbb{S}^n \\ & X \succeq 0\end{aligned}$$

and the dual semidefinite program

$$\begin{aligned} d^\star = \max \quad & y^T b \\ & \sum_{i=1}^{m} y_i A_i \preceq C \\ & y \in \mathbb{R}^m. \end{aligned}$$

If the primal semidefinite program has a strictly feasible solution and if it is bounded from below, then strong duality holds: $p^\star = d^\star$. Likewise, if the dual semidefinite program has a strictly feasible solution and if it is bounded from above, then also strong duality holds.

Although semidefinite programs are much more general than linear programs, they are not much harder to solve. Most interior-point methods for linear programming have been generalized to semidefinite programs [Van+05].

2.5 Mixed Integer Bilinear Optimization

Bilinear optimization is a special case of quadratically constrained quadratic optimization. The distinctive feature is, that the variables can be partitioned into two sets such that in the objective function and in the constraints, there only appear products of variables coming from different partitions.

Definition 2.7. A *bilinear program* is of the form

$$\begin{aligned} \min \quad & x^T C y + c^T \left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right) \\ & x^T A_i y + a_i^T \left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right) \leq b_i \quad \forall i \in [m] \\ & x \in \mathbb{R}^{n_x}, \quad y \in \mathbb{R}^{n_y}, \end{aligned}$$

where $C, A_1, \ldots, A_m \in \mathbb{R}^{n_x \times n_y}$, $c, a_1, \ldots, a_m \in \mathbb{R}^{n_x + n_y}$ and $b_1, \ldots, b_m \in \mathbb{R}$.

While semidefinite relaxation is a powerful tool to provide lower bounds for the class of non-convex quadratically constrained quadratic programs, there is another type of convex relaxation for the class of non-convex bilinear programs. This relaxation consists in the application of McCormick envelopes [McC76].

Given a bilinear term $w = xy$ with lower bounds x^L, y^L and upper bounds x^U, y^U on the variables x and y, one has the following four inequalities:

$$\begin{aligned}
&x - x^L \geq 0, \\
&y - y^L \geq 0, \\
&x^U - x \geq 0, \\
&y^U - y \geq 0.
\end{aligned}$$

By multiplying the first two inequalities and the last two inequalities, respectively, we obtain

$$\begin{aligned}
0 &\leq \left(x - x^L\right)\left(y - y^L\right) = w - y^L x - x^L y + x^L y^L, \\
0 &\leq \left(x^U - x\right)\left(y^U - y\right) = w - y^U x - x^U y + x^U y^U,
\end{aligned}$$

which is equivalent to

$$\begin{aligned}
w &\geq y^L x + x^L y - x^L y^L, \\
w &\geq y^U x + x^U y - x^U y^U.
\end{aligned}$$

These are the two convex underestimators of the function w. Likewise, by taking the product of inequalities one and four and two and three, respectively, we obtain

$$\begin{aligned}
0 &\leq \left(x - x^L\right)\left(y^U - y\right) = -w + y^U x + x^L y - x^L y^U, \\
0 &\leq \left(x^U - x\right)\left(y - y^L\right) = -w + y^L x + x^U y - x^U y^L,
\end{aligned}$$

which is equivalent to

$$w \leq y^U x + x^L y - x^L y^U,$$
$$w \leq y^L x + x^U y - x^U y^L.$$

These are the two convex overestimators of the function w. A graphical representation of both is shown in Figure 2.5.

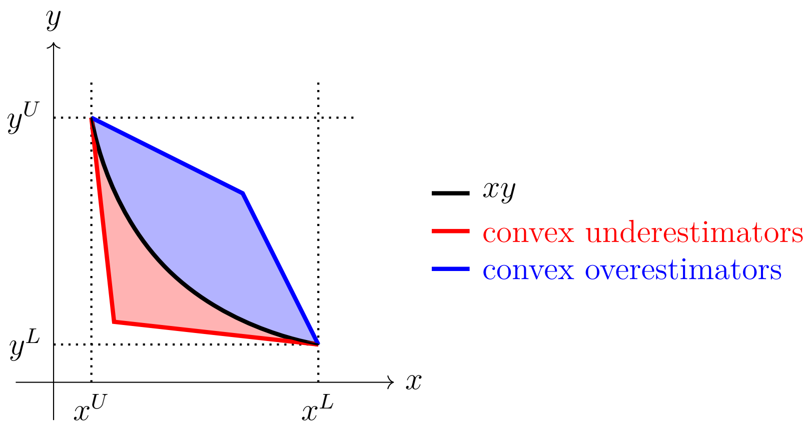


Figure 2.5: Geometric interpretation of McCormick envelopes

The advantage of McCormick envelopes is that they retain convexity while minimizing the size of the new feasible region. This allows the lower bound solutions obtained from the McCormick relaxation to be closer to the original solution than if other convex relaxations were used.

Unlike quadratically constrained quadratic programs, bilinear programs do not include mixed integer linear programs as a special case. If one set of the variables is required to by integer, we speak of a mixed integer bilinear program.

Definition 2.8. A *mixed integer bilinear program* is of the form

$$\begin{aligned} \min \quad & x^T C y + c^T \binom{x}{y} \\ & x^T A_i y + a_i^T \binom{x}{y} \leq b_i \quad \forall i \in [m] \\ & x \in \mathbb{R}^{n_x}, \quad y \in \mathbb{Z}^{n_y} \end{aligned}$$

where $C, A_1, \ldots, A_m \in \mathbb{R}^{n_x \times n_y}$, $c, a_1, \ldots, a_m \in \mathbb{R}^{n_x + n_y}$ and $b_1, \ldots, b_m \in \mathbb{R}$.

If we apply the McCormick relaxation to a mixed integer bilinear problem, we obtain a mixed integer linear problem that can than be approached by the solution techniques described in Section 2.2.

Chapter 3

Algorithmic Approach for $s_k\left(P^4(r), \omega_0\right)$

We now have both the appropriate background in symplectic geometry and in combinatorial optimization to set up an algorithm for calculating the simplex packing widths $s_k\left(P^4(r), \omega_0\right)$. We call this the outer optimization problem. The reason why we call this the *outer* optimization problem is because we will also define an *inner* optimization problem later on. Since we are dealing with simplices in dimension two, we will use the terms triangle and simplex interchangeably.

3.1 Outer Optimization Problem

Problem 3.1 (Outer Optimization Problem)**.**
Given $k \in \mathbb{N}$*, determine* $s_k\left(P^4(1), \omega_0\right)$.

As already mentioned in Chapter 1, the key to convert Problem 3.1 into a combinatorial optimization problem is Theorem 1.13. We recall the statement:

Let V be an open, connected subset of $\mathbb{R}^n$ with $H_1(V) = 0$. Then

$$s_k\left(V \times \mathbb{T}^n, \omega_0\right) = \sup \left\{ r \;\middle|\; \begin{array}{l} \exists A_1, \ldots, A_k \in \mathrm{GL_n}(\mathbb{Z}) \; \exists t_1, \ldots, t_k \in \mathbb{R}^n : \\ A_i\left(\text{◺}^n(r)\right) + t_i \subseteq V \quad \forall i \in [k] \\ \left(A_i\left(\text{◺}^n(r)\right) + t_i\right) \cap \left(A_j\left(\text{◺}^n(r)\right) + t_j\right) = \emptyset \\ \forall 1 \leq i < j \leq k \end{array} \right\}.$$

By applying this theorem to $P^4(1) = \mathbb{T}^2 \times \text{◺}^2(1)$, we get the following equivalent formulation of Problem 3.1.

Problem 3.2 (Outer Optimization Problem - Combinatorial Formulation)**.**
Given $k \in \mathbb{N}$, determine the minimum side length s such that there exist matrices $A_1, \ldots, A_k \in \mathrm{GL}_2(\mathbb{Z})$ and vectors $t_1, \ldots, t_k \in \mathbb{R}^2$ satisfying

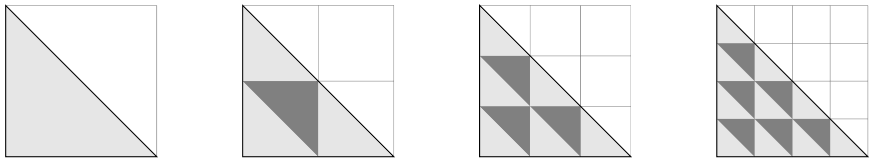
$$A_i\left(\text{◺}^2(1)\right) + t_i \subseteq \text{◺}^2(s) \qquad \forall i \in [k] \qquad \textit{(containment condition)},$$

$$\left(A_i\left(\text{◺}^2(1)\right) + t_i\right) \cap \left(A_j\left(\text{◺}^2(1)\right) + t_j\right) = \emptyset \quad \forall 1 \leq i < j \leq k \quad \textit{(disjointness condition)}.$$

Note that previously we were maximizing the size of the packing objects but now we are minimizing the size of the packing container. The reason for this is that the latter is easier to model. We denote the minimum side length s from Problem 3.2 by $s_k^{\text{◺}}$. Then we have the relation

$$s_k\left(P^4(1), \omega_0\right) = \frac{1}{s_k^{\text{◺}}}.$$

The complexity of computing $s_k^{\text{◺}}$ is unknown. The problem is conjectured to be $\mathcal{NP}$-hard in general. In contrast, the calculation of $s_k^{\text{◺}}$ is trivial if k is a square number. In this case we can create a packing with side length $\sqrt{k}$ that is dense and therefore optimal. The corresponding packings for the first four square numbers $k = 1, 4, 9, 16$ are visualized in Figure 3.1.



$s_1^{\triangle} = 1$ $s_4^{\triangle} = 2$ $s_9^{\triangle} = 3$ $s_{16}^{\triangle} = 4$

Figure 3.1: Optimal triangle packings for square numbers

If we can fit sixteen triangles into the standard simplex with side length four, then we can also fit less then sixteen triangles into it. Therefore, we obtain an upper bound on $s_k^{\triangle}$ by rounding up $\sqrt{k}$.

Theorem 3.3 (Trivial Upper Bound on $s_k^{\triangle}$)**.**
$s_k^{\triangle} \leq \left\lceil \sqrt{k} \right\rceil$

By taking the reciprocal value of $s_k\left(P^4(1), \omega_0\right)$ in Theorem 1.12, we obtain a stronger upper bound on $s_k^{\triangle}$ for $k = 1, \ldots, 20$. We denote the values by $\overline{s_k^{\triangle}}$. For $k \leq 12$ and $k = 16$ we even have equality $\overline{s_k^{\triangle}} = s_k^{\triangle}$.

Theorem 3.4 (Upper Bound on $s_k^{\triangle}$ [MMT00])**.**

k	1	2	3	4	5	6	7	8	9	10
$\overline{s_k^{\triangle}}$	1	2	2	2	$\frac{5}{2}$	$\frac{17}{6}$	3	3	3	$\frac{10}{3}$

k	11	12	13	14	15	16	17	18	19	20
$\overline{s_k^{\triangle}}$	$\frac{7}{2}$	$\frac{56}{15}$	$\frac{23}{6}$	$\frac{79}{20}$	4	4	$\frac{17}{4}$	$\frac{13}{3}$	$\frac{9}{2}$	$\frac{97}{21}$

We only need to consider triangles whose shapes are admissible for that upper bound. For a given k, we call the list of admissible triangles the shapelist $\mathcal{S}_k^{\triangle}$.

Definition 3.5. For $k = 1, \ldots, 20$ we define the *shapelist*

$$\mathcal{S}_k^{\triangle} = \left\{ T \;\middle|\; \exists A \in \mathrm{GL}_2(\mathbb{Z})\ \exists t \in \mathbb{R}^2 : T = A\left(\llcorner^2(1)\right) \ \text{and}\ T + t \subseteq \llcorner^2\left(\overline{s_k^{\triangle}}\right) \right\}.$$

The admissible shapes for $k = 1, \ldots, 20$ are shown in Table 3.1 and Table 3.2. The value $s_{\min}^{\triangle}$ in the last column denotes the minimum side length of the enclosing standard simplex.

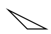
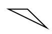
Shape	Image	$s_{\min}^{\triangle}$
$T_1 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right)\right\}\right)\right)$		1
$T_2 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\end{smallmatrix}\right)\right\}\right)\right)$		2
$T_3 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\end{smallmatrix}\right)\right\}\right)\right)$		2
$T_4 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right)\right\}\right)\right)$		2
$T_5 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\end{smallmatrix}\right)\right\}\right)\right)$		2
$T_6 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-2\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\end{smallmatrix}\right)\right\}\right)\right)$		2
$T_7 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\end{smallmatrix}\right)\right\}\right)\right)$		2
$T_8 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\end{smallmatrix}\right)\right\}\right)\right)$		2
$T_9 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\2\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{10} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\2\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{11} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\1\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{12} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\1\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{13} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-2\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{14} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-3\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-2\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{15} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-2\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-3\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{16} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-3\end{smallmatrix}\right)\right\}\right)\right)$		3

Table 3.1: Triangle shapes 1 - 16

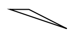
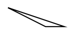
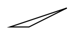
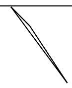
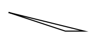
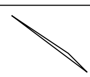
Shape	Image	$s^{\triangle}_{\min}$
$T_{17} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\-1\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{18} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\-2\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{19} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\-1\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{20} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\-2\end{smallmatrix}\right)\right\}\right)\right)$		3
$T_{21} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\3\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{22} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\2\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\3\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{23} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\1\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{24} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\1\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{25} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-3\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{26} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-4\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-3\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{27} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\-4\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{28} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}4\\-1\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{29} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}4\\-3\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{30} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-3\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\-4\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{31} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}4\\-1\end{smallmatrix}\right)\right\}\right)\right)$		4
$T_{32} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}3\\-2\end{smallmatrix}\right), \left(\begin{smallmatrix}4\\-3\end{smallmatrix}\right)\right\}\right)\right)$		4

Table 3.2: Triangle shapes 17 - 32

The image of the two-dimensional standard simplex under an integral transformation is the interior of a triangle with integer vertices and volume $\frac{1}{2}$. For a triangle $T = \operatorname{int}(\operatorname{conv}(\{a, b, c\}))$ with $a, b, c \in \mathbb{Z}^2$, the volume is given by

$$\operatorname{vol}(T) = \frac{1}{2}\left|\det\left(\begin{bmatrix} a-c & b-c \end{bmatrix}\right)\right|.$$

To establish the shapelists, we consider all triangles with integer vertices in the interval $[0,4] \times [0,4]$ that satisfy the volume condition. We then sort them by ascending x-coordinates. In case of equality between two x-coordinates, we sort them by ascending y-coordinates. Next, we shift the triangles, such that the first vertex becomes the origin and remove all copies.

For $k = 21, \ldots, 25$ the interval to be examined becomes $[0,5] \times [0,5]$ and the corresponding number of shapes increases to 44. Since computing $s_k^{\triangle}$ for $k > 14$ seems out of reach at the moment, we do not have to consider those shapes. By looking at $s_{\min}^{\triangle}$ we can immediately create the shapelist $\mathcal{S}_k^{\triangle}$. For $k = 1, \ldots, 20$ the shapelists are given by

$$\begin{aligned}
\mathcal{S}_k^{\triangle} &= \{T_1\} && \text{for } k = 1,\\
\mathcal{S}_k^{\triangle} &= \{T_1, \ldots, T_8\} && \text{for } k = 2, \ldots, 6,\\
\mathcal{S}_k^{\triangle} &= \{T_1, \ldots, T_{20}\} && \text{for } k = 7, \ldots, 14,\\
\mathcal{S}_k^{\triangle} &= \{T_1, \ldots, T_{32}\} && \text{for } k = 15, \ldots, 20.
\end{aligned}$$

Now we can determine $s_k^{\triangle}$ by computing an optimal packing for every k-cardinality multisubset of the shapelist. The number of multisubsets of length k on $|\mathcal{S}_k^{\triangle}|$ symbols is termed $|\mathcal{S}_k^{\triangle}|$ multichoose k and given by the formula

$$\left(\!\!\binom{|\mathcal{S}_k^{\triangle}|}{k}\!\!\right) = \binom{|\mathcal{S}_k^{\triangle}| + k - 1}{k} = \frac{\left(|\mathcal{S}_k^{\triangle}| + k - 1\right)!}{\left(|\mathcal{S}_k^{\triangle}| - 1\right)!k!}.$$

k	$\lvert\mathcal{S}_k^{\triangle}\rvert$	$\left(\!\binom{\lvert\mathcal{S}_k^{\triangle}\rvert}{k}\!\right)$
1	1	1
2	8	36
3	8	120
4	8	330
5	8	792
6	8	1 716
7	20	657 800
8	20	2 220 075
9	20	6 906 900
10	20	20 030 010
11	20	54 627 300
12	20	141 120 525
13	20	347 373 600
14	20	818 809 200
15	32	511 738 760 544
16	32	1 503 232 609 098
17	32	4 244 421 484 512
18	32	11 554 258 485 616
19	32	30 405 943 383 200
20	32	77 535 155 627 160

Table 3.3: Number of k-cardinality multisubsets of the shapelists

The value of $\left(\!\binom{\lvert\mathcal{S}_k^{\triangle}\rvert}{k}\!\right)$ according to k is shown in Table 3.3. For $k = 10$ one already needs to solve more than twenty million subproblems. This clearly indicates that complete enumeration is out of the question, especially in view of the fact that only exponential algorithms for the computation of the optimal packings are known. Rather, we implement a branch-and-bound approach. Each node of the branch-and-bound search tree represents a multisubset. The level of the node equates to the cardinality of the multisubset. For example, the root on level zero corresponds to the empty multisubset and the children on level one correspond to all subsets of cardinality one.

The (incomplete) search tree for $k = 3$ is shown in Figure 3.2. We start the search with the global upper bound from Theorem 3.4. Whenever the computation for a node produces an optimum packing that exceeds the global upper bound, we can fathom the subtree rooted at this node. Whenever the computation for a node at level k produces an optimum packing that improves the global upper bound, we update the global upper bound and memorize the packing as the incumbent solution. At termination the incumbent solution is an optimum packing with value of the upper bound. We will explain the procedure in more detail in Section 3.3.

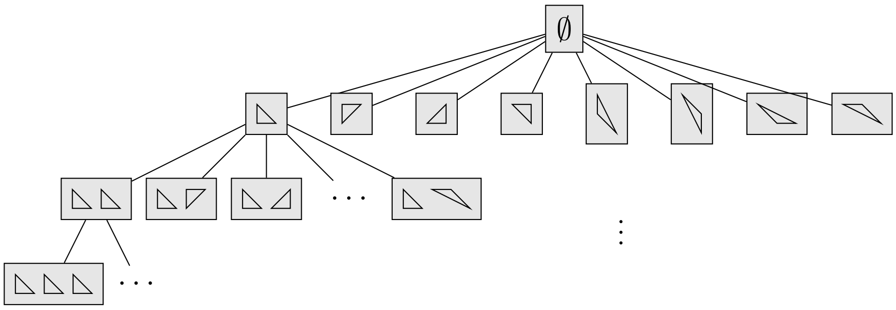


Figure 3.2: Branch-and-bound search tree for the 3-triangle packing

The next question we will address is how to actually compute an optimal packing at each node. We call this the inner optimization problem.

3.2 Inner Optimization Problem

Problem 3.6 (Inner Optimization Problem).
Given $T_1, \ldots, T_m \in \mathcal{S}_k^{\triangle}$, determine the minimum side length s such that there exist vectors $t_1, \ldots, t_m \in \mathbb{R}^2$ satisfying

$$T_i + t_i \subseteq \triangle^2(s) \qquad \forall i \in [m] \qquad \textit{(containment condition)},$$

$$(T_i + t_i) \cap (T_j + t_j) = \emptyset \qquad \forall 1 \leq i < j \leq m \qquad \textit{(disjointness condition)}.$$

We will formulate Problem 3.6 as a mixed integer linear program. For this we have to model the two conditions. First, we will model the containment condition.

The triangles T_i are given in the form

$$T_i = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}a_i\\b_i\end{smallmatrix}\right), \left(\begin{smallmatrix}c_i\\d_i\end{smallmatrix}\right)\right\}\right)\right).$$

The packing container $\llcorner^2(s)$ is given by

$$\llcorner^2(s) = \left\{ \left(\begin{smallmatrix}x\\y\end{smallmatrix}\right) \in \mathbb{R}^2 \;\middle|\; x > 0,\ y > 0,\ x + y < s \right\}.$$

Instead of working with the open sets T_i and $\llcorner^2(s)$, we will consider their closures. This does not make any difference for the containment condition but is easier to model. Let $t_i = \left(\begin{smallmatrix}x_i\\y_i\end{smallmatrix}\right)$ denote the translation vector, then the closure of the translated triangle $T_i + t_i$ is given by

$$\overline{T_i + t_i} = \operatorname{conv}\left(\left\{\left(\begin{smallmatrix}x_i\\y_i\end{smallmatrix}\right), \left(\begin{smallmatrix}x_i+a_i\\y_i+b_i\end{smallmatrix}\right), \left(\begin{smallmatrix}x_i+c_i\\y_i+d_i\end{smallmatrix}\right)\right\}\right)$$

and the closed standard simplex of side length s is given by

$$\overline{\llcorner^2(s)} = \left\{ \left(\begin{smallmatrix}x\\y\end{smallmatrix}\right) \in \mathbb{R}^2 \;\middle|\; x \geq 0,\ y \geq 0,\ x + y \leq s \right\}.$$

Because of convexity, it suffices to check the containment condition for the three vertices only. Thus, we obtain a total of nine inequalities for every $i \in [m]$:

$$\begin{aligned}
x_i &\geq 0, & y_i &\geq 0, & x_i + y_i &\leq s,\\
x_i + a_i &\geq 0, & y_i + b_i &\geq 0, & (x_i + a_i) + (y_i + b_i) &\leq s,\\
x_i + c_i &\geq 0, & y_i + d_i &\geq 0, & (x_i + c_i) + (y_i + d_i) &\leq s.
\end{aligned}$$

By putting the constants to the right hand sides and taking extrema in every column, we can reduce these nine inequalities to three inequalities for every $i \in [m]$:

$$x_i \geq \max\{0, -a_i, -c_i\} \qquad =: K_i,$$

$$y_i \geq \max\{0, -b_i, -d_i\} \qquad =: K_i',$$

$$x_i + y_i - s \leq \min\{0, -a_i - b_i, -c_i - d_i\} \qquad =: K_i''.$$

Thus, we get a total of $3m$ inequalities that we call the containment constraints.

Next, we model the disjointness condition. The equation $(T_i + t_i) \cap (T_j + t_j) = \emptyset$ is equivalent to $t_j - t_i \notin (T_i \ominus T_j)$. Here, $T_i \ominus T_j$ denotes the Minkowski difference of T_i and T_j that is defined as

$$T_i \ominus T_j = \{v_i - v_j \mid v_i \in T_i, v_j \in T_j\}.$$

The equivalence between the two formulations can be easily derived as follows:

$$\begin{aligned} & (T_i + t_i) \cap (T_j + t_j) \neq \emptyset \\ \Leftrightarrow \quad & \exists v_i \in T_i \, \exists v_j \in T_j : v_i + t_i = v_j + t_j \\ \Leftrightarrow \quad & \exists v_i \in T_i \, \exists v_j \in T_j : t_j - t_i = v_i - v_j \\ \Leftrightarrow \quad & t_j - t_i \in (T_i \ominus T_j). \end{aligned}$$

To understand what the Minkowski difference of two polytopes looks like, we need the following lemma.

Lemma 3.7. *Let $A, B \subseteq \mathbb{R}^n$ be two sets. Then* $\operatorname{conv}(A \ominus B) = \operatorname{conv}(A) \ominus \operatorname{conv}(B)$.

Proof. Let $A, B \subseteq \mathbb{R}^n$ be two sets. First, we will show that $\operatorname{conv}(A \ominus B)$ is contained in $\operatorname{conv}(A) \ominus \operatorname{conv}(B)$. For every point $p \in \operatorname{conv}(A \ominus B)$ there exist $N \in \mathbb{N}$, $p_1, \ldots, p_N \in A \ominus B$ and $\lambda_1, \ldots, \lambda_N \geq 0$ with $\sum_{i=1}^N \lambda_i = 1$ such that $p = \sum_{i=1}^N \lambda_i p_i$. By definition of the Minkowski difference, p_i is of the form $p_i = a_i - b_i$ with $a_i \in A$

and $b_i \in B$ for every $i \in [N]$. Hence,

$$\begin{aligned} p &= \sum_{i=1}^{N} \lambda_i p_i \\ &= \sum_{i=1}^{N} \lambda_i (a_i - b_i) \\ &= \underbrace{\sum_{i}^{N} \lambda_i a_i}_{\in \operatorname{conv}(A)} - \underbrace{\sum_{i}^{N} \lambda_i b_i}_{\in \operatorname{conv}(B)} \in \operatorname{conv}(A) \ominus \operatorname{conv}(B). \end{aligned}$$

Next, we show that $\operatorname{conv}(A) \ominus \operatorname{conv}(B)$ is contained in $\operatorname{conv}(A \ominus B)$. Every point $p \in \operatorname{conv}(A) \ominus \operatorname{conv}(B)$ is of the form $p = a - b$ with $a \in \operatorname{conv}(A)$ and $b \in \operatorname{conv}(B)$. Since $a \in \operatorname{conv}(A)$, there exist $a_1, \dots, a_N \in A$ and $\lambda_1, \dots, \lambda_N \geq 0$ with $\sum_{i=1}^{N} \lambda_i = 1$ such that $a = \sum_{i=1}^{N} \lambda_i a_i$. Thus,

$$\begin{aligned} p &= a - b \\ &= \sum_{i=1}^{N} \lambda_i a_i - b \\ &= \sum_{i=1}^{N} \lambda_i a_i - \sum_{i=1}^{N} \lambda_i b \\ &= \sum_{i=1}^{N} \lambda_i \underbrace{(a_i - b)}_{\in A \ominus B} \in \operatorname{conv}(A \ominus B). \end{aligned}$$

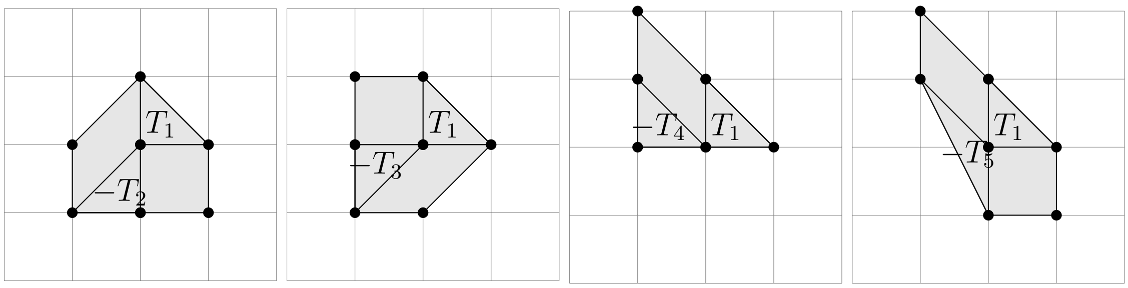
□

If we apply Lemma 3.7 to

$$\begin{aligned} T_i &= \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} a_i \\ b_i \end{pmatrix}, \begin{pmatrix} c_i \\ d_i \end{pmatrix}\right\}\right)\right), \\ T_j &= \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} a_j \\ b_j \end{pmatrix}, \begin{pmatrix} c_j \\ d_j \end{pmatrix}\right\}\right)\right), \end{aligned}$$

we immediately get a description of $T_i \ominus T_j$ as

$$T_i \ominus T_j = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}a_i\\b_i\end{pmatrix}, \begin{pmatrix}c_i\\d_i\end{pmatrix}, \begin{pmatrix}-a_j\\-b_j\end{pmatrix}, \begin{pmatrix}a_i-a_j\\b_i-b_j\end{pmatrix}, \begin{pmatrix}c_i-a_j\\d_i-b_j\end{pmatrix}, \begin{pmatrix}-c_j\\-d_j\end{pmatrix}, \begin{pmatrix}a_i-c_j\\b_i-d_j\end{pmatrix}, \begin{pmatrix}c_i-c_j\\d_i-d_j\end{pmatrix}\right\}\right)\right).$$



$T_1 \ominus T_2$ $\quad$ $T_1 \ominus T_3$ $\quad$ $T_1 \ominus T_4$ $\quad$ $T_1 \ominus T_5$

Figure 3.3: Visualization of the Minkowski difference $T_i \ominus T_j$

Figure 3.3 shows what the Minkowski difference for some triangles of our shapelist looks like. For our algorithmic approach, we need a description of $T_i \ominus T_j$ as an intersection of finitely many halfspaces. In this two dimensional setup, the halfspaces of $T_i \ominus T_j$ are among the set of halfspaces of T_i translated by a vertex of $-T_j$ and the set of halfspaces of $-T_j$ translated by a vertex of T_i. Let

$$H = \left\{ \begin{pmatrix}x\\y\end{pmatrix} \in \mathbb{R}^2 \;\middle|\; \alpha x + \beta y < \gamma \right\}$$

be a halfspace of T_i and let

$$\gamma' = \max\{0, \alpha(-a_j) + \beta(-b_j), \alpha(-c_j) + \beta(-d_j)\}$$

be the maximum value of the three vertices of $-T_j$ inserted into the halfspace description. Then

$$H' = \left\{ \begin{pmatrix}x\\y\end{pmatrix} \in \mathbb{R}^2 \;\middle|\; \alpha x + \beta y < \gamma' \right\}$$

is a halfspace of $T_i \ominus T_j$. By applying this procedure to all pairs of halfspaces and vertices, we end up with a description of $T_i \ominus T_j$ as

$$T_i \ominus T_j = \left\{ \binom{x}{y} \in \mathbb{R}^2 \;\middle|\; \alpha_f^{ij} x + \beta_f^{ij} y < \gamma_f^{ij} \forall f \in [6] \right\}$$

Here, $\alpha_f^{ij}, \beta_f^{ij} \gamma_f^{ij} \in \mathbb{Z}$ are integer constants. Working with this representation, the difference vector $t_j - t_i$ is not contained in the Minkowski difference $T_i \ominus T_j$ if and only if at least one of the six inequalities $\alpha_f^{ij} x + \beta_f^{ij} y < \gamma_f^{ij}$ is violated. To model this condition, we introduce a binary variable $z_f^{ij} \in \{0, 1\}$ for every $f \in [6]$ with the following meaning:

$$z_f^{ij} = 1 \Rightarrow \alpha_f^{ij} (x_j - x_i) + \beta_f^{ij} (y_j - y_i) \geq \gamma_f^{ij}.$$

Now one could by brute force enumerate all $\mathcal{O}\left(6^{\binom{m}{2}}\right)$ possible 0/1-assignments of the binary variables z_f^{ij} and solve a linear program for each. This strategy was pursued by Maley, Mastrangeli and Traynor [MMT00]. Instead, we model the implication using a Big-M-formulation, where the parameter M has to be chosen sufficiently large:

$$\alpha_f^{ij} (x_j - x_i) + \beta_f^{ij} (y_j - y_i) \geq \gamma_f^{ij} - M \left(1 - z_f^{ij}\right).$$

If we plug in $z_f^{ij} = 1$ into the Big-M-formulation, we obtain

$$\alpha_f^{ij} (x_j - x_i) + \beta_f^{ij} (y_j - y_i) \geq \gamma_f^{ij}$$

as desired. Let $\hat{s}$ be the current global upper bound on s in the branch-and-bound algorithm when encountering the inner optimization problem. To find an appropriate value for M, we can make the following estimation on the left hand side of the Big-M-formulation:

$$
\begin{aligned}
&\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) \\
\geq &- |\alpha_f^{ij}||x_j - x_i| - |\beta_f^{ij}||y_j - y_i| \\
\geq &- |\alpha_f^{ij}|\hat{s} - |\beta_f^{ij}|\hat{s} \\
= &- \left(|\alpha_f^{ij}| + |\beta_f^{ij}|\right)\hat{s}.
\end{aligned}
$$

Thus, we can choose M as

$$
M = \left(|\alpha_f^{ij}| + |\beta_f^{ij}|\right)\hat{s} + \gamma_f^{ij}.
$$

We want at least one of the six binary variables z_f^{ij} to take the value one. Equivalently we can say that their sum should be greater or equal to one. Altogether, the disjointness condition is equivalent to

$$
\begin{aligned}
&z_f^{ij} \in \{0, 1\}, \\
&z_1^{ij} + \cdots + z_6^{ij} \geq 1, \\
&\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) \geq \gamma_f^{ij} - M\left(1 - z_f^{ij}\right)
\end{aligned}
$$

for all $f \in [6]$. For every pair of triangles T_i, T_j with $1 \leq i < j \leq m$, we have these seven inequalities, so in total there are $7\binom{m}{2}$ inequalities that we call the disjointness constraints.

We consider the binary variables z_f^{ij} as components of the vector $z \in \{0,1\}^{6\binom{m}{2}}$ that is indexed by

$$
z = \begin{pmatrix} z_1^{12} & \cdots & z_6^{12} & z_1^{13} & \cdots & z_6^{13} & \cdots & z_1^{m-1\,m} & \cdots & z_6^{m-1\,m} \end{pmatrix}^T.
$$

By merging the containment constraints and the disjointness constraints, we obtain a mixed integer linear program with $1 + 2m$ continuous variables, $6\binom{m}{2}$ (binary) integer variables and $3m + 7\binom{m}{2}$ constraints.

Problem 3.8 (Inner Optimization Problem - Mixed Integer Linear Formulation)**.**

$$\begin{aligned}
\min \quad & s \\
& x_i \geq K_i && \forall i \in [m] \\
& y_i \geq K_i' && \forall i \in [m] \\
& x_i + y_i - s \leq K_i'' && \forall i \in [m] \\
& z_1^{ij} + \cdots + z_6^{ij} \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) \geq \gamma_f^{ij} - M\left(1 - z_f^{ij}\right) && \forall 1 \leq i < j \leq m \\
& && \forall f \in [6] \\
& s \in \mathbb{R}, \quad x, y \in \mathbb{R}^m, \quad z \in \{0,1\}^{6\binom{m}{2}}
\end{aligned}$$

In the next section we will give a brief summary of the implementation.

3.3 Implementation

The computer code is written in the programming language C. The backbone of the program is a queue of branch-and-bound nodes. Each of these nodes represents an inner optimization instance and has the following attributes:

`structure bbnode`

- `nsimplices`: the number of simplices,
- `nshapes`: the number of different shapes,
- `*shape`: the different shapes,
- `*multi`: the multiplicity of each shape,
- `lowerbound`: he local lower bound,
- `*next`: the pointer to the next `bbnode` element.

For example, the multisubset $\{T_1, T_1, T_1, T_3, T_3, T_4, T_5, T_8\}$ has the assignments

- `nsimplices` $= 8$;
- `nshapes` $= 5$;
- `*shape` $= [1, 3, 4, 5, 8]$;
- `*multi` $= [3, 2, 1, 1, 1]$.

The assignments of `lowerbound` and `*next` depend on the point of time the multisubset is generated. At the beginning, the queue is initialized by all feasible multisubsets of cardinality one that corresponds to all shapes from the corresponding shapelist. An example for $k = 3$ is shown in Figure 3.4. Each element has the value `lowerbound` $= 0$, since no packing has been computed yet.

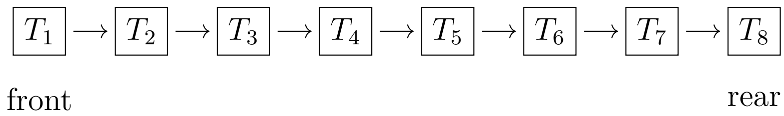


Figure 3.4: Initial queue for the 3-triangle packing

After the inner optimization problem at a node is solved, we extend the queue by adding all feasible extensions of the shapes from the inner optimization instance. Figure 3.5 shows what the extended queue looks like after treating the first node from the initial queue. We use a breadth-first search strategy in our branch-and-bound algorithm since it yields slightly better results than a depth-first search strategy.

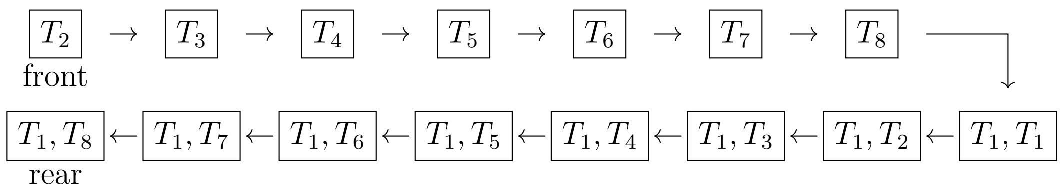


Figure 3.5: Extended queue for the 3-triangle packing

We use another structure with the same attributes as `bbnode` (except the attribute `lowerbound`) to record all packings that yield solutions that exceed the global upper bound. Because these packings serve as blocking configurations in our algorithm, the corresponding structure is called `config` and is given by:

structure config

- `nsimplices`: the number of simplices,
- `nshapes`: the number of different shapes,
- `*shape`: the different shapes,
- `*multi`: the multiplicity of each shape,
- `*next`: the pointer to the next `config` element.

To avoid unnecessary repeated computations, we store the values of all solved inner optimization problems in a file that we call the bounds file. To establish the bounds file, we use the following structure:

structure bound

- `fipri`: the unique fingerprint of the inner optimization instance,
- `nsimplices`: the number of simplices,
- `lowerbound`: the local lower bound,
- `optimal`: the indicator whether `lowerbound` is equal to the optimal value,
- `*next`: the pointer to the next `bound` structure.

We obtain the unique fingerprint for each inner optimization instance by assigning a prime number to each shape as shown in Table 3.4. We then take the product over all prime numbers of the corresponding shapes from the inner optimization instance. For example, the multisubset $\{T_1, T_1, T_1, T_3, T_3, T_4, T_5, T_8\}$ has fingerprint $2^3 \cdot 5^2 \cdot 7 \cdot 11 \cdot 19 = 292\,600$.

Shape	Prime Number
T_1	2
T_2	3
T_3	5
T_4	7
T_5	11
T_6	13
T_7	17
T_8	19
T_9	23
T_{10}	29
T_{11}	31
T_{12}	37
T_{13}	41
T_{14}	43
T_{15}	47
T_{16}	53
T_{17}	59
T_{18}	61
T_{19}	67
T_{20}	71

Table 3.4: Assignment of a prime number to each shape

Since we encounter each inner optimization problem with a global upper bound, some instances might be infeasible. In this case we set `optimal` $= 0$ and store the global upper bound in the attribute `lowerbound`. If the inner optimization problem is solved to optimality, we set `optimal` $= 1$ and store the optimal solution in the attribute `lowerbound`.

Each line of the bounds file contains the four attributes `fipri`, `nsimplices`, `lowerbound` and `optimal`. Table 3.5 gives an impression of the evolution of the bounds file. It shows the number of bounds contained in the bounds file after computing the k-triangle packing for $k = 1, \ldots, 13$.

k	#Bounds
1	1
2	44
3	55
4	66
5	467
6	585
7	255 526
8	255 789
9	255 836
10	273 136
11	311 057
12	429 705
13	603 174

Table 3.5: Number of bounds contained in the bounds file after computing the k-triangle packing for $k = 1, \ldots, 13$

Now as long as the queue is not empty, we extract its front element. If the value of `lowerbound` exceeds the global upper bound, the node is fathomed. Otherwise we check the inner optimization instance for blocking subsets by searching the list of `config` structures. If we detect a blocking subset, the node is fathomed. Otherwise, we compute the fingerprint of the inner optimization instance and check the bounds file for previously known bounds. If a lower bound is found that exceeds the global upper bound, the inner optimization instance is recorded as a new blocking structure and the node is fathomed. In any other case, we run the inner optimization solver.

If the solution computed by the inner optimization solver exceeds the global upper bound, the inner optimization instance is recorded as a new blocking structure and the node is fathomed. If the optimal value of the solution is smaller or equal to the global upper bound, we distinguish between two cases.

In the first case, the number of simplices in the inner optimization instance is equal to the number k of simplices in the outer optimization problem, which means the node is a leaf of the branch-and-bound tree. If the optimal value of the solution is equal to the current global upper bound, we draw a picture of the solution and append it to a TEX-file. This TEX-file already contains pictures of all previous solutions with the same optimal value. If the optimal value of the solution is strictly smaller than the global upper bound, we update the latter and replace the old TEX-file by a new TEX-file containing a picture of the solution.

In the second case, the number of simplices in the inner optimization instance is smaller than the number k of simplices in the outer optimization problem, which means the node is not a leaf. If so, we add all feasible extensions of the inner optimization instance to the queue.

All described steps are summarized in Figure 3.6. The whole program code encompasses 1841 lines. The branch-and-bound framework comprises 406 lines thereof ($\approx$ 22%). The code for the inner optimization procedure uses 471 lines ($\approx$ 26%). For reading and rewriting the bounds file and the TEXfile it takes 204 lines ($\approx$ 11%) and 162 lines ($\approx$ 9%), respectively.

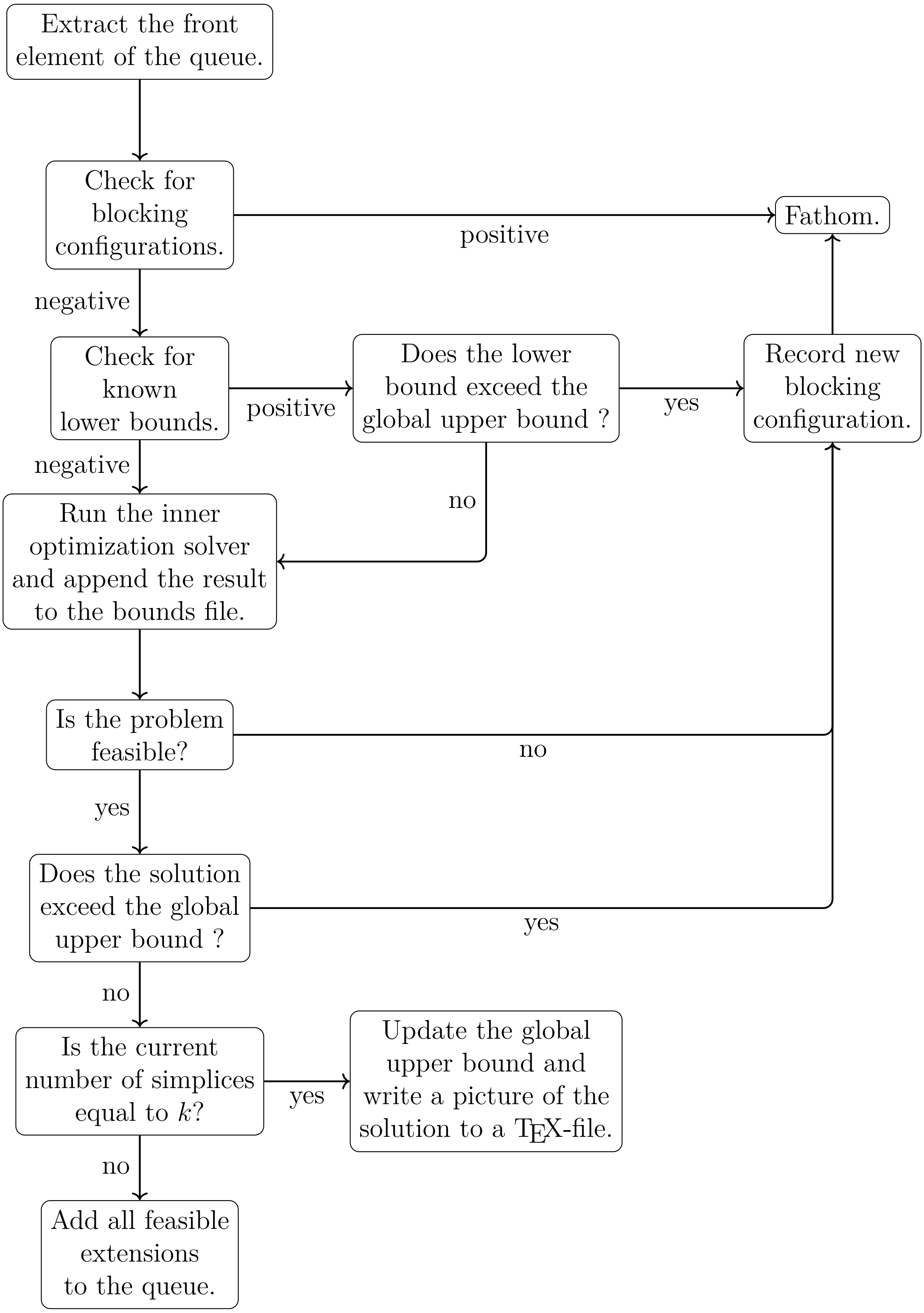


Figure 3.6: Workflow of our algorithm

3.4 Experimental Results

The computational experiments were carried out under the Debian 10 operating system on two Intel Xeon E5-2690v2 CPUs with 3.00GHz and 10 cores each. For solving the inner optimization problem, we call the GUROBI Optimization Software Version 9.0.3. We also implemented a version using the CPLEX Optimizer for solving the inner optimization problem but the performance was almost the same.

Before we discuss the timing statistics of the algorithm, we take a look at some of the computed packings. Figure 3.7 and Figure 3.8 show one exemplary optimal packing for $k = 1, \ldots, 13$.

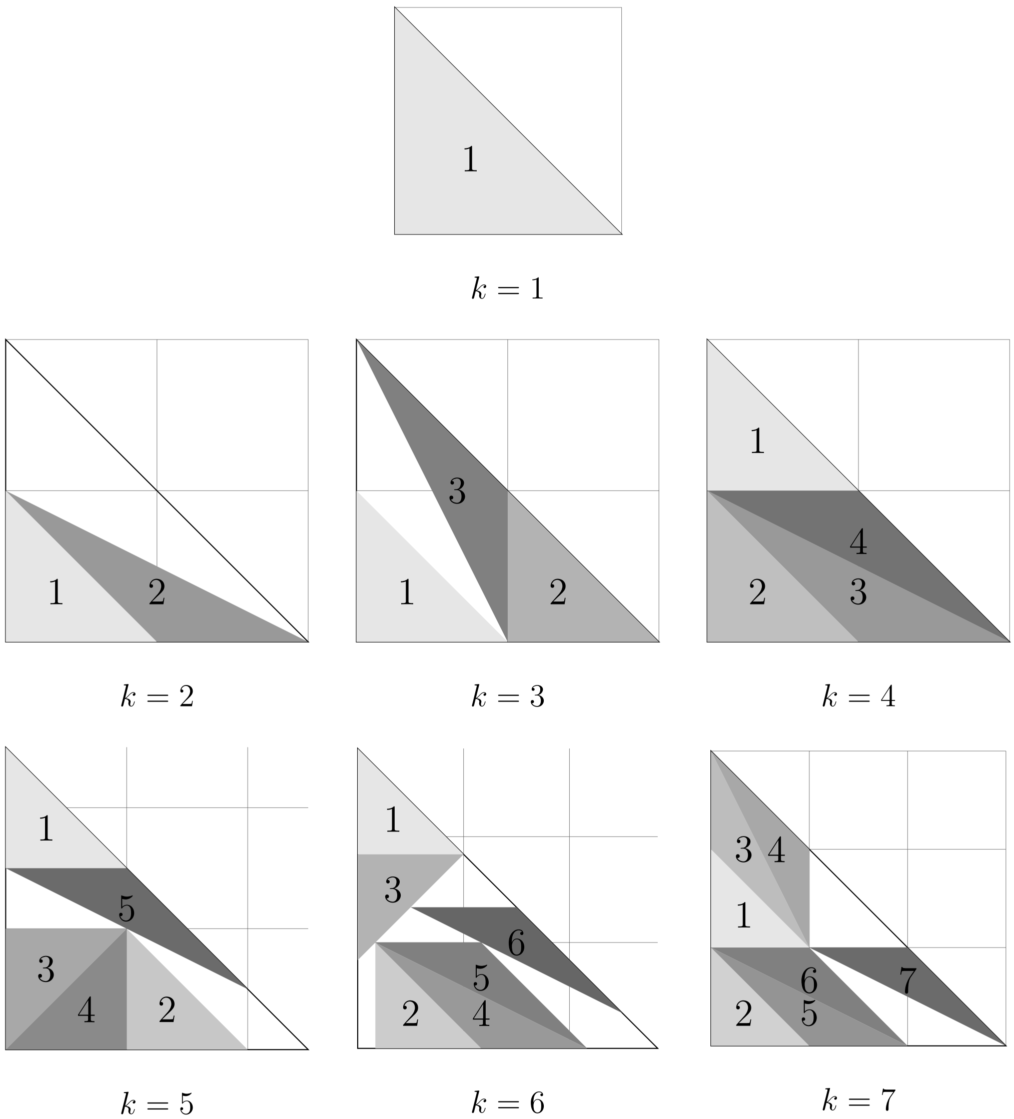


Figure 3.7: Optimal k-triangle packings for $k = 1, \ldots, 7$

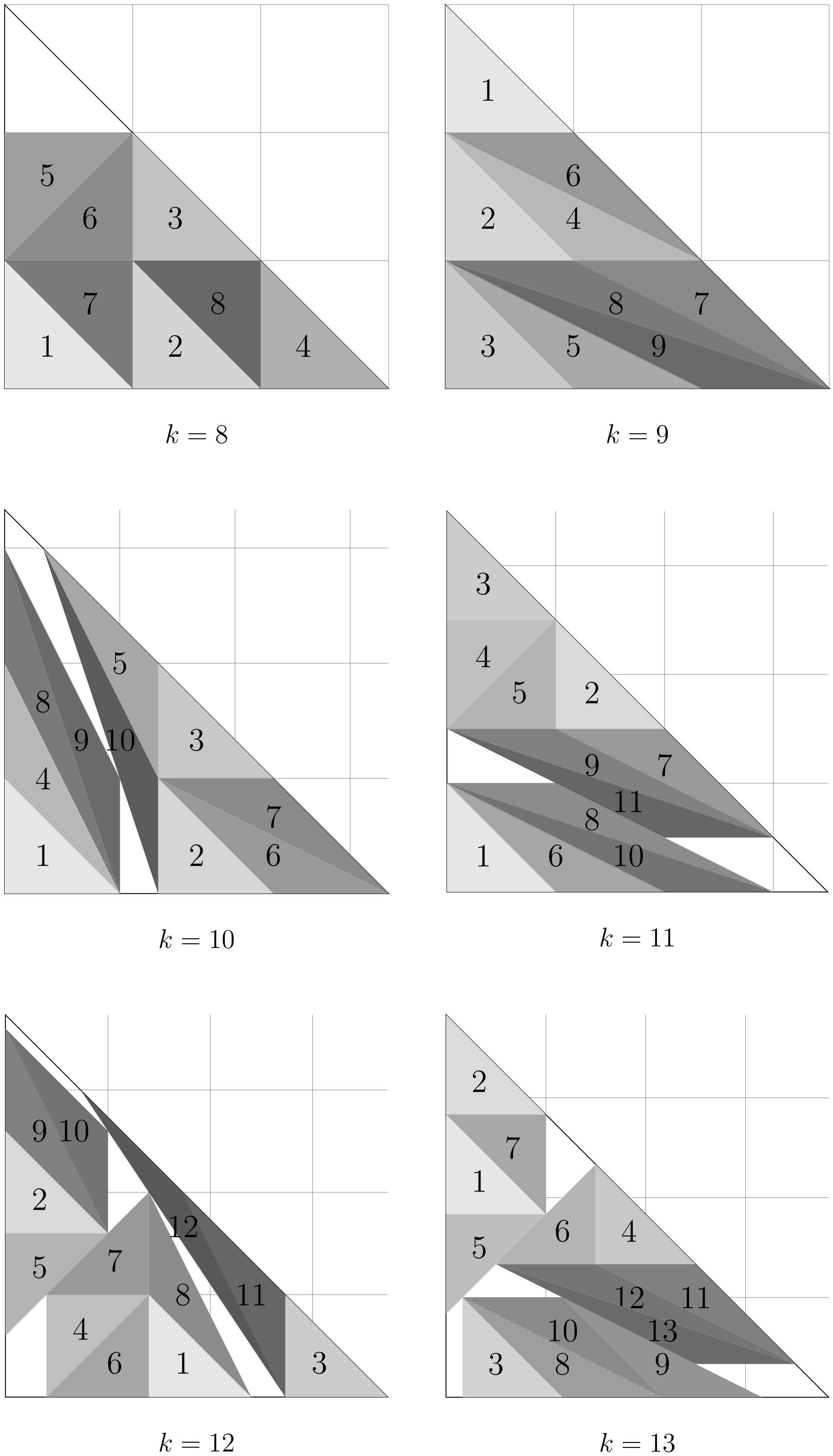


Figure 3.8: Optimal k-triangle packings for $k = 8, \ldots, 13$

For $k = 1, \ldots, 11$ we confirm the values of $s_k^{\triangle}$ found by Traynor et. al that were presented in Theorem 1.12. For $k = 12$ and $k = 13$ we can verify their conjecture that the upper bounds they found are indeed optimal.

Our program does not only compute one optimal packing for a given k but detects all multisubsets that allow for an optimal packing. We denote such multisubsets as optimal. Table 3.6 shows the number of optimal multisubsets for $k = 1, \ldots, 13$.

k	#Optimal Multisubsets
1	1
2	11
3	11
4	4
5	18
6	21
7	668
8	261
9	47
10	198
11	142
12	78
13	166

Table 3.6: Number of multisubsets that allow for an optimal k-triangle packing for $k = 1, \ldots, 13$

For some multisubsets there exist even more than one optimal packing but finding them all would blow up the computing time of our algorithm. We consider packings of the same multisubset whose optimal packing widths coincide as equivalent. This choice of equivalence relation is of pure combinatorial nature. From a symplectic point of view all packings for a given number k are equivalent. This is due to the following result of McDuff.

Theorem 3.9 (Symplectic Equivalence [McD91]).
All isometric embeddings

$$\varphi : \bigsqcup_{i=1}^{k} \big(B^4(r), \omega_0\big) \hookrightarrow \big(B^4(1), \omega_0\big)$$

are isotopic through symplectic embeddings.

Here, the word isometric refers to the standard Euclidean metric on $\mathbb{R}^4$. One could also consider other equivalence relations such as calling two packings equivalent if we can map one to the other one (up to renumbering) by an isometry of the outer triangle. The isometries of the outer triangle are the identity and the reflection at the axis $x = y$. Figure 3.9 shows two optimal 8-packings that are equivalent under this relation.

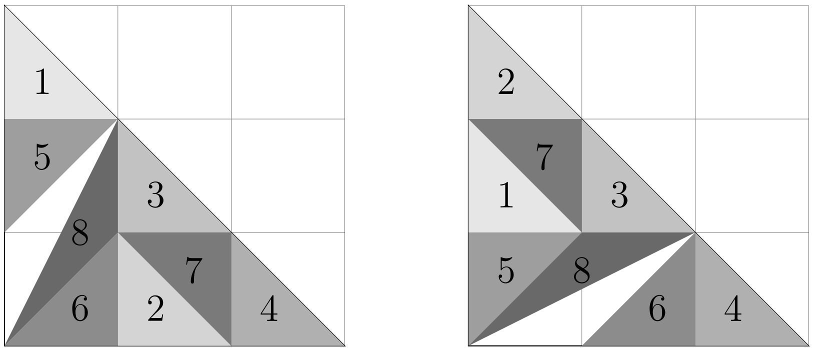


Figure 3.9: Equivalent 8-triangle packings under isometries of the outer triangle

One could also call two packings equivalent if we can map one to the other one by an affine transformation that preserves the outer triangle. Figure 3.10 shows two optimal 8-packings that are equivalent under this relation.

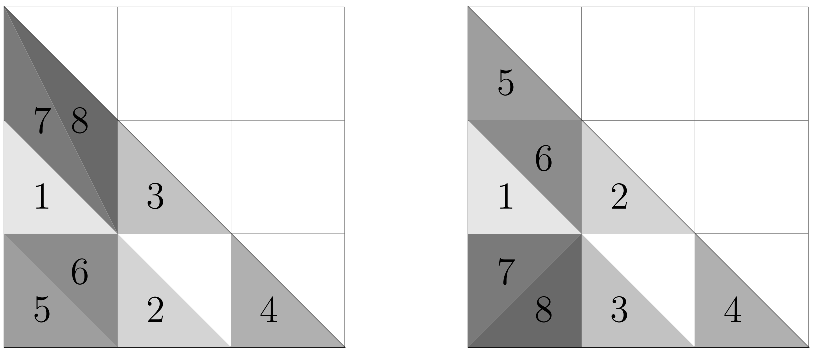


Figure 3.10: Equivalent 8-triangle packings under affine transformations

The triangle on the right hand side is obtained from the triangle on the left hand side by a function composition of shearing along the y-axis and reflecting along the x-axis. Shearing along the y-axis corresponds to the affine map

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ x+y \end{pmatrix},$$

which leads to the configuration in Figure 3.11 and changes the form of the outer triangle. Reflecting along the x-axis corresponds to the affine map

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ -y \end{pmatrix},$$

which maps the outer triangle back to its original shape.

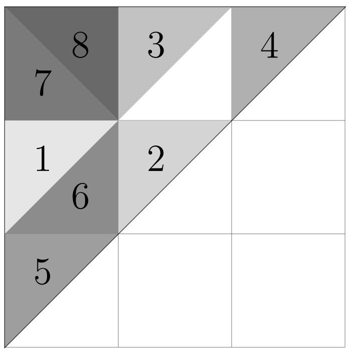


Figure 3.11: Shearing an 8-triangle packing along the y-axis

After this short digression about symplectic equivalence relations, we come back to the computational results of our algorithm. Table 3.7 shows the timing statistics of our algorithm for $k = 1, \ldots, 13$. The column labels are:

- k: the number of triangles to be packed,
- $\left(\!\binom{|S_k^{\triangle}|}{k}\!\right)$: the number of k-cardinality multisubsets of the shapelist,
- #I-Calls: the number of calls to the inner optimization procedure,
- Avg I-Time: the average cpu time spent in an inner optimization procedure,
- Max I-Time: the maximum cpu time spent in an inner optimization procedure,
- Total Time: the total cpu time.

The number of calls to the inner optimization procedure is significantly smaller than the number of k-cardinality multisubsets of the shapelist that would have been considered using complete enumeration. Due to the data base we built up from previous runs for smaller k, many calls of the inner optimization procedure have been replaced by simple data base queries.

k	$\left(\!\binom{\lvert \mathcal{S}_k^{\triangle} \rvert}{k}\!\right)$	#I-Calls	Avg I-Time	Max I-Time	Total Time
1	1	1	0 : 00 : 00.00	0 : 00 : 00.00	0 : 00 : 00.00
2	36	43	0 : 00 : 00.00	0 : 00 : 00.01	0 : 00 : 00.02
3	120	11	0 : 00 : 00.00	0 : 00 : 00.02	0 : 00 : 00.03
4	330	11	0 : 00 : 00.01	0 : 00 : 00.09	0 : 00 : 00.14
5	792	433	0 : 00 : 00.01	0 : 00 : 00.27	0 : 00 : 03.11
6	1 716	185	0 : 00 : 00.04	0 : 00 : 00.33	0 : 00 : 06.72
7	657 800	255 158	0 : 00 : 00.00	0 : 00 : 13.30	0 : 18 : 56.90
8	2 220 075	263	0 : 00 : 00.14	0 : 00 : 04.46	0 : 12 : 29.10
9	6 906 900	47	0 : 00 : 00.09	0 : 00 : 01.64	0 : 11 : 59.33
10	20 030 010	34 029	0 : 00 : 00.52	0 : 02 : 27.23	4 : 56 : 28.02
11	54 627 300	43 187	0 : 00 : 07.67	0 : 46 : 18.30	92 : 05 : 28.83
12	141 120 525	129 630	0 : 00 : 09.39	3 : 26 : 34.38	338 : 19 : 31.65
13	347 373 600	196 735	0 : 00 : 37.88	46 : 59 : 29.37	2070 : 08 : 23.43

Table 3.7: Timing statistics for the k-triangle packing given in the format "hh:mm:ss" for $k = 1, \ldots, 13$

One can see that the inner optimization procedure is quite fast on average but a few instances with larger values of k can take very long, especially for $k = 13$. Most of these hard instances are problems that are infeasible. The hope is that semidefinite relaxation of the inner optimization problem will allow for a much faster computation of bounds in the branch-and-bound tree. Thereby, the mixed integer linear formulation must only be employed if the semidefinite bounds are not strong enough.

We will discuss this approach in Chapter 4. Moreover, we will present further improvement strategies. They involve strengthening the mixed integer linear formulation by adding symmetry breaking inequalities, applying a time limit to the inner optimization procedure and computing the McCormick relaxation of a mixed integer binary version of the problem.

Chapter 4

Improvements to the Algorithmic Approach

4.1 Symmetry Breaking

Considering the outer optimization problem, there are no symmetries involved, since we are disregarding the order of the shapes in the multisubsets, which makes every multisubset unique. Considering the inner optimization problem, we encounter symmetric solutions whenever a shape occurs more than once in the respective multisubset. For example, consider the inner optimization instance $\{T_1, T_1, T_1, T_4\}$. As can be seen in Figure 4.1, we obtain six symmetric optimal solutions for this multisubset.

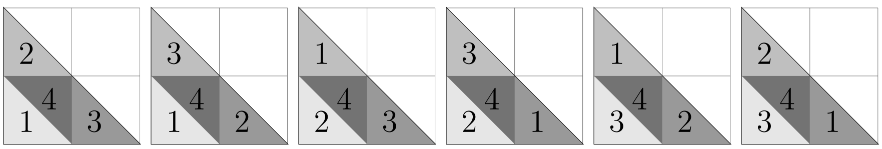


Figure 4.1: Symmetric solutions for the multisubset $\{T_1, T_1, T_1, T_4\}$

In general, for each shape with multiplicity m we obtain $m!$ symmetric solutions. Our approach to reduce the occurrence of symmetric solutions is to add artificial symmetry breaking inequalities to the mixed integer linear version of the inner optimization problem. We have examined three different types of symmetry breaking inequalities. All of them try to put an order on the displacement variables (x_i, y_i) belonging to triangles of the same shape. The first type puts a lexicographic order on the x-coordinates by adding the inequality

$$x_i \leq x_{i+1}$$

for each duplicate of a shape. In the example above, we would add the inequalities $x_1 \leq x_2$ and $x_2 \leq x_3$, which eliminates all solutions except the first and third one. The second type sorts the displacement variables (x_i, y_i) by their sum $x_i + y_i$ by adding the inequality

$$x_i + y_i \leq x_{i+1} + y_{i+1}$$

for each duplicate of a shape. In the example above, we would add the inequalities $x_1 + y_1 \leq x_2 + y_2$ and $x_2 + y_2 \leq x_3 + y_3$, which eliminates all solutions except the first and second one. The third type is an extension of the first type in terms of putting a total lexicographical order on the displacement variables. Like the first type, we add the inequality $x_i \leq x_{i+1}$ but additionally we also add the inequality $y_i \leq y_{i+1}$ in the case the two variables x_i and x_{i+1} coincide. To model the implication $x_i = x_{i+1} \Rightarrow y_i \leq y_{i+1}$, we use the Big-$M$-formulation

$$M(x_i - x_{i+1}) + y_i \leq y_{i+1}.$$

Applied to the example above, we would add the inequalities $x_1 \leq x_2$, $x_2 \leq x_3$, $M(x_1 - x_2) + y_1 \leq y_2$ and $M(x_2 - x_3) + y_2 \leq y_3$, which eliminates all solutions but the first one. In our computations we choose $M = 50$. This does not model the implication perfectly, since the difference between the variables x_i and x_{i+1} can be arbitrarily small. However, it suffices to disregard most symmetries.

To compare the three symmetry breaking inequality types, we calculated all optimal k-triangle packings for $k = 1, \ldots, 12$. Table 4.1 shows the computation time of our algorithm combined with the corresponding symmetry breaking constraint. The inequality type numbers are given in the following order:

- Type 0: No symmetry breaking inequality applied.
- Type 1: $x_i \leq x_{i+1}$.
- Type 2: $x_i + y_i \leq x_{i+1} + y_{i+1}$.
- Type 3: $(x_i \leq x_{i+1}) \wedge (x_i = x_{i+1} \Rightarrow y_i \leq y_{i+1})$.

k	Type 0	Type 1	Type 2	Type 3
1	0 : 00 : 00.00	0 : 00 : 00.00	0 : 00 : 00.00	0 : 00 : 00.00
2	0 : 00 : 00.04	0 : 00 : 00.04	0 : 00 : 00.02	0 : 00 : 00.03
3	0 : 00 : 00.05	0 : 00 : 00.03	0 : 00 : 00.03	0 : 00 : 00.01
4	0 : 00 : 00.36	0 : 00 : 00.14	0 : 00 : 00.14	0 : 00 : 00.02
5	0 : 00 : 05.67	0 : 00 : 03.16	0 : 00 : 03.11	0 : 00 : 02.88
6	0 : 00 : 12.71	0 : 00 : 09.37	0 : 00 : 06.72	0 : 00 : 08.62
7	0 : 28 : 47.35	0 : 17 : 42.80	0 : 18 : 56.90	0 : 14 : 33.28
8	0 : 12 : 47.89	0 : 12 : 25.75	0 : 12 : 29.10	0 : 12 : 30.34
9	0 : 11 : 57.62	0 : 11 : 57.32	0 : 11 : 59.33	0 : 11 : 59.01
10	16 : 04 : 10.11	5 : 33 : 28.68	4 : 56 : 28.02	5 : 29 : 36.59
11	1103 : 05 : 34.83	116 : 21 : 41.18	92 : 05 : 28.83	93 : 30 : 54.37
12	4006 : 41 : 10.15	506 : 53 : 59.92	338 : 19 : 31.56	530 : 56 : 44.44

Table 4.1: Computation time of the k-triangle packing involving symmetry breaking constraints given in the format "hh:mm:ss" for $k = 1, \ldots, 12$

For $k \geq 10$, one can see that applying symmetry breaking constraints is an enormous improvement to the run-time of the algorithm. For small values of k, there is no significant difference between the three symmetry breaking constraints, but for $k = 12$ the clear winner is the second symmetry breaking type of the form $x_i + y_i \leq x_{i+1} + y_{i+1}$. For the run producing the computational results presented in Table 3.7, we also chose this symmetry breaking type.

We also tested a generalization of the second symmetry breaking type of the form

$$ax_i + by_i \leq ax_{i+1} + by_{i+1},$$

where the parameters $a, b \in \mathbb{Z}$ can be chosen arbitrarily. However, none of the tested combinations of the two parameters could achieve better results than choosing $a = b = 1$.

4.2 Time Limit

As discussed in Section 3.4, the inner optimization procedure is quite fast on average, but a few instances for larger values of k can take very long. Therefore, we tried out to use a time limit for the inner optimization procedure. Interestingly, applying a time limit of $30s$ did not change the number of found optimal multisubsets for $k = 1, \ldots, 12$. This was not the case for $k = 13$. Table 4.2 shows the impact of applying different time limits to the 13-triangle packing problem.

Time Limit	Avg I-Time	Total Time	#Optimal Multisubsets	#Open Problems
∞	0 : 00 : 37.88	2069 : 57 : 15.91	166	0
$30s$	0 : 00 : 18.49	1001 : 34 : 32.45	160	1 650
$20s$	0 : 00 : 16.61	896 : 31 : 09.62	157	2 256
$10s$	0 : 00 : 13.82	738 : 34 : 51.60	155	4 253

Table 4.2: Timing statistics for the 13-triangle packing under different time limits to the inner optimization procedure given in the format "hh:mm:ss"

The first column shows the different time limits that we applied to the inner optimization procedure. The second and third column show the average cpu time spent in an inner optimization procedure and the total cpu time, respectively. The last two columns show the number of found multisubsets that allow for an optimal packing and those that exceeded the respective time limit. We collected all instances that exceeded the time limit in a file. For a random selection of those hard instances, we then tried different GUROBI parameter settings. The GUROBI Optimizer provides a wide variety of parameters that allow one to control the operation of the optimization engines. To find parameter values that improve the performance on our model, we both manually tested different parameter settings and used the built-in automated parameter tuning tool. We found that there are four parameters that speed up the computation time for most instances. Table 4.3 gives an overview of the four parameters together with their default value and the modified value.

Parameter	Description	Default Value	Modified Value
Presolve	Controls the presolve level	-1	2
PreDual	Controls presolve model dualization	-1	1
MIPFocus	MIP solver focus	0	2
Heuristics	Time spent in feasibility heuristics	0.05	0

Table 4.3: Modified GUROBI parameter setting

The parameter *Presolve* controls the process whereby the mixed integer linear program is examined for logical reduction opportunities. The default value of -1 corresponds to an automatic setting whereas the modified value of 2 corresponds to an aggressive setting. Other options are off (0) and conservative (1). The parameter *PreDual* controls whether the dual of the linear relaxation is formed in the presolve process. The default setting of -1 uses a heuristic to decide while the modified setting of 1 forces it to take the dual. The parameter *MIPFocus* defines the solution strategy of the mixed integer linear programming solver. The default value of 0 asks the solver to strike a balance between finding new feasible solutions and proving that the current solution is optimal. The modified value of 2 focuses more attention on proving optimality. The parameter *Heuristics* determines the amount of time spent in heuristics. The value can be chosen as a decimal number between 0 and 1 and represents the desired fraction of total run-time devoted to heuristic. Therefore, the modified value of 0 prohibits the solver to use heuristics. In Table 4.4 we compare the run-times using the default values and modified values of the described parameters for fifteen different inner optimization instances.

Instance	Default Values	Modified Values	Relation
$6 \times T_1, T_2, T_4, T_7, T_{20}$	$0:05:35.1$	$0:07:05.9$	+27%
$6 \times T_1, 2 \times T_2, T_3, T_5, T_7$	$1:09:27.9$	$1:18:41.9$	+13%
$6 \times T_1, T_6, 2 \times T_7, T_{13}$	$0:07:04.0$	$0:06:40.9$	−5%
$6 \times T_1, T_2, T_4, T_7, T_8, T_{19}$	$0:08:53.7$	$0:07:05.0$	−20%
$4 \times T_1, 2 \times T_2, 2 \times T_3, T_5, 2 \times T_6, T_7$	$0:23:38.4$	$0:18:37.8$	−21%
$4 \times T_1, 3 \times T_2, 2 \times T_3, T_4, T_7, 2 \times T_8$	$0:19:10.8$	$0:12:48.9$	−33%
$5 \times T_1, T_2, T_3, T_4, 3 \times T_5$	$0:09:18.2$	$0:06:10.6$	−34%
$4 \times T_1, 2 \times T_2, 3 \times T_3, T_{11}, 2 \times T_{12}$	$1:18:47.5$	$0:50:11.0$	−36%
$5 \times T_1, 3 \times T_2, T_6, T_7, T_8$	$0:51:31.6$	$0:30:46.2$	−40%
$5 \times T_1, T_2, T_3, T_5, 3 \times T_6$	$0:47:18.8$	$0:28:37.3$	−40%
$6 \times T_1, 2 \times T_4, T_5, 2 \times T_8$	$0:15:42.6$	$0:09:32.4$	−42%
$7 \times T_1, 2 \times T_3, T_{10}$	$0:13:22.1$	$0:07:29.5$	−44%
$6 \times T_1, 2 \times T_2, 2 \times T_4, T_5$	$0:57:16.6$	$0:31:22.7$	−45%
$6 \times T_1, 3 \times T_4, T_5, 2 \times T_7$	$0:45:13.3$	$0:17:18.8$	−62%
$8 \times T_1, T_4, T_6, T_7$	$1:11:40.1$	$0:24:28.9$	−66%

Table 4.4: Timing statistics for different inner optimization instances under modified GUROBI parameter setting given in the format "hh:mm:ss"

The instances shown in the first column were randomly chosen from the collection of instances of the 13-triangle packing problem that exceeded the time limit of 30 seconds. Column two and column three show the run-time of the inner optimization solver using the default values of the parameters *Presolve*, *PreDual*, *MIPFocus* and *Heuristics* and the modified values from Table 4.3, respectively. The last column highlights the relative change in run-time where negative numbers mean time reduction and positive numbers mean time increase. The instances have been sorted accordingly.

As can be seen, the modified parameter setting improves the run-time on most instances. On average, the run-time decreases by 36%. Therefore, we apply the following strategy for solving the outer optimization problem: Whenever an instance of the inner optimization problem exceeds the given time limit, we change the parameter values according to Table 4.3 and solve the problem again. The results are summarized in Table 4.5.

Time Limit	Parameter Setting	Avg I-Time	Total Time	#Optimal Multisubsets	#Open Problems
∞	default	0 : 00 : 37.88	2069 : 57 : 15.91	166	0
∞	modified	0 : 00 : 47.74	2608 : 06 : 35.65	166	0
$30s$	default	0 : 00 : 18.49	1001 : 34 : 32.45	160	1 650
$30s$	modified	0 : 00 : 23.31	1274 : 41 : 01.53	165	1 486

Table 4.5: Timing statistics for the 13-triangle packing under time limit to the inner optimization procedure combined with modified GUROBI parameter setting given in the format "hh:mm:ss"

By reapplying the inner optimization procedure under the modified parameter setting on the instances that exceeded the time limit of 30 seconds, we were able to find all optimal multisubsets but one. Because we kept the given time limit for the second iteration, there are still 1486 problems that are left unsolved. Instead of increasing the time limit or solving those instances offline, we decided to just ignore those problems as the effort seems to exceed to benefit.

This decreased the computation time by approximately 38% compared to applying no time limit and using the default parameter setting. In contrast, when applying the modified parameter setting to all inner optimization instances, the computation time increases by approximately 26%. Unfortunately even after applying this strategy, we were still not able to compute an optimal 14-triangle packing.

4.3 Semidefinite Relaxation

We are going to reformulate the inner optimization problem as a quadratically constrained quadratic program. Once this formulation is achieved, we can consider the semidefinite relaxation of this program. The hope is that the semidefinite relaxation gives us strong lower bounds on s such that the mixed integer linear exact formulation only needs to be applied if the bounds are not strong enough. This could in turn speed up the branch-and-bound process of the outer optimization problem. We recall the definition of the inner optimization problem.

Problem 4.1 (Inner Optimization Problem)**.**
Given $T_1, \dots, T_m \in \mathcal{S}_k^{\triangle}$*, determine the minimum side length* s *such that there exist vectors* $t_1, \dots, t_m \in \mathbb{R}^2$ *satisfying*

$$\begin{aligned} T_i + t_i &\subseteq \triangle^2(s) \quad \forall i \in [m] && \textit{(containment condition)}, \\ (T_i + t_i) \cap (T_j + t_j) &= \emptyset \quad \forall 1 \le i < j \le m && \textit{(disjointness condition)}. \end{aligned}$$

We have three different approaches to model the inner optimization problem as a quadratically constrained quadratic program. We will successively describe each approach and conclude this section with a comparison of the results.

4.3.1 First Approach

In this approach we will model the containment condition just like we did in the mixed integer linear formulation. As derived in Section 3.2, the containment condition is equivalent to

$$\begin{aligned} x_i &\ge \max\{0, -a_i, -c_i\} && =: K_i, \\ y_i &\ge \max\{0, -b_i, -d_i\} && =: K_i', \\ x_i + y_i - s &\le \min\{0, -a_i - b_i, -c_i - d_i\} && =: K_i'' \end{aligned}$$

for all $i \in [m]$. Now we will model the disjointness constraint. In Section 3.2 we showed that $(T_i + t_i) \cap (T_j + t_j) = \emptyset$ is equivalent to $t_j - t_i \notin (T_i \ominus T_j)$ where the Minkowski difference is given by

$$T_i \ominus T_j = \left\{ \left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right) \in \mathbb{R}^2 \;\middle|\; \alpha_f^{ij} x + \beta_f^{ij} y < \gamma_l^{ij} \forall f \in [6] \right\}.$$

Hence, the difference vector $t_j - t_i$ is not contained in the Minkowski difference $T_i \ominus T_j$ if and only if at least one of the six inequalities $\alpha_f^{ij} x + \beta_f^{ij} y < \gamma_f^{ij}$ is violated. As before, we introduce the binary variable $z_f^{ij} \in \{0, 1\}$ for every $f \in [6]$ with the following meaning:

$$z_f^{ij} = 1 \Rightarrow \alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) \geq \gamma_f^{ij}.$$

Now instead of expressing this implication by a Big-M-inequality, we will use the quadratic inequality

$$z_f^{ij} \left(\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) - \gamma_f^{ij} \right) \geq 0.$$

We formulate the binary constraint $z_f^{ij} \in \{0, 1\}$ by the equation $z_f^{ij}(1 - z_f^{ij}) = 0$. Since we want at least one of the variables z_f^{ij} to take the value one, we require $z_1^{ij} + \ldots + z_6^{ij} \geq 1$.

We desire the quadratically constrained quadratic program to be in homogeneous form. To eliminate the linear terms, we introduce a variable $t \in \mathbb{R}$ with $t^2 = 1$ and multiply all linear terms by it. In the optimal solution of this modified problem, the variable t can either take the value $t = 1$ or $t = -1$. In the first case the solution to the original problem is (s, x, y, z) and in the second case the solution to the original problem is $(-s, -x, -y, -z)$. Thus, we obtain the following quadratically constrained quadratic formulation of the inner optimization problem.

Problem 4.2 (Inner Optimization Problem - Quadratically Constrained Quadratic Formulation 1)**.**

$$\begin{aligned}
\min \quad & ts \\
& tx_i \geq K_i && \forall i \in [m] \\
& ty_i \geq K_i' && \forall i \in [m] \\
& tx_i + ty_i - ts \leq K_i'' && \forall i \in [m] \\
& tz_1^{ij} + \cdots + tz_6^{ij} \geq 1 && \forall 1 \leq i < j \leq m \\
& z_f^{ij}\left(\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) - t\gamma_f^{ij}\right) \geq 0 && \forall 1 \leq i < j \leq m \quad \forall f \in [6] \\
& z_f^{ij}\left(t - z_f^{ij}\right) = 0 && \forall 1 \leq i < j \leq m \quad \forall f \in [6] \\
& t^2 = 1 \\
& s \in \mathbb{R}, \quad x, y \in \mathbb{R}^m, \quad z \in \mathbb{R}^{6\binom{m}{2}}, \quad t \in \mathbb{R}.
\end{aligned}$$

Note, that in the standard form of a quadratically constrained quadratic program as defined in Section 2.3, we did not allow inequalities within the constraints. Even though the inequalities can be easily resolved by introducing slack variables, we will refrain from doing so, since the solver we are using to compute the semidefinite relaxation of this program is capable of handling inequality constraints.

All constraints apart from

$$z_f^{ij}\left(\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) - t\gamma_f^{ij}\right) \geq 0$$

are convex. To see that this constraint is not convex, we compute the Hessian matrix of the function

$$f(x_i, x_j, y_i, y_j, z, t) = z\left(\alpha(x_j - x_i) + \beta(y_j - y_i) - t\gamma\right)$$

and check whether it is positive semidefinite. The partial derivatives of first order are given by

$$\begin{aligned}
\frac{\partial f}{\partial x_i} &= -\alpha z, \\
\frac{\partial f}{\partial x_j} &= \alpha z, \\
\frac{\partial f}{\partial y_i} &= -\beta z, \\
\frac{\partial f}{\partial y_j} &= \beta z, \\
\frac{\partial f}{\partial z} &= \alpha(x_j - x_i) + \beta(y_j - y_i) - t\gamma, \\
\frac{\partial f}{\partial t} &= -\gamma z.
\end{aligned}$$

Therefore, the Hessian matrix H_f is of the form

$$H_f = \begin{bmatrix} 0 & 0 & 0 & 0 & -\alpha & 0 \\ 0 & 0 & 0 & 0 & \alpha & 0 \\ 0 & 0 & 0 & 0 & -\beta & 0 \\ 0 & 0 & 0 & 0 & \beta & 0 \\ -\alpha & \alpha & -\beta & \beta & 0 & -\gamma \\ 0 & 0 & 0 & 0 & -\gamma & 0 \end{bmatrix}.$$

The characteristic polynomial of H_f is given by

$$\begin{aligned}
\det\left(H_f - \lambda I\right) &= \det\left(\begin{bmatrix} -\lambda & 0 & 0 & 0 & -\alpha & 0 \\ 0 & -\lambda & 0 & 0 & \alpha & 0 \\ 0 & 0 & -\lambda & 0 & -\beta & 0 \\ 0 & 0 & 0 & -\lambda & \beta & 0 \\ -\alpha & \alpha & -\beta & \beta & -\lambda & -\gamma \\ 0 & 0 & 0 & 0 & -\gamma & -\lambda \end{bmatrix}\right) \\
&= -\lambda^4\left(\lambda^2 - 2\alpha^2 - 2\beta^2 - \gamma^2\right).
\end{aligned}$$

Hence, the eigenvalues of H_f are 0 with multiplicity four, $\sqrt{2\alpha^2 + 2\beta^2 + \gamma^2}$ and $-\sqrt{2\alpha^2 + 2\beta^2 + \gamma^2}$. This implies that H_f is not positive semidefinite - unless $\alpha = \beta = \gamma = 0$, which is not the case in our setting. Because of the non-convexity of

Problem 4.2, it cannot be solved directly. Instead, we will apply the semidefinite relaxation technique described in Section 2.3. Let us define the vector of variables

$$v = \begin{pmatrix} s & x_1 \cdots x_m & y_1 \cdots y_m & z_1^{12} \cdots z_6^{m-1m} & t \end{pmatrix}^T \in \mathbb{R}^N,$$

where $N = 2 + 2m + 6\binom{m}{2}$. We lift this vector v into the space $\mathbb{S}^N$ by introducing the matrix $V = vv^T$ of $\operatorname{rank}(V) = 1$ given by

$$V = \begin{bmatrix} s^2 & sx_1 & \cdots & sx_m & sy_1 & \cdots & sy_m & sz_6^{12} & \cdots & sz_6^{m-1m} & st \\ x_1 s & x_1^2 & \cdots & x_1 x_m & x_1 y_1 & \cdots & x_1 y_m & x_1 z_6^{12} & \cdots & x_1 z_6^{m-1m} & x_1 t \\ \vdots & \vdots & & & & & & & & & \\ x_m s & x_m x_1 & & & & & & & & & \\ y_1 s & y_1 x_1 & & & & & & & & & \\ \vdots & \vdots & & & & \ddots & & & & & \\ y_m s & y_m x_1 & & & & & & & & & \\ z_1^{12} s & z_1^{12} x_1 & & & & & & & & & \\ \vdots & & & & & & & & & & \\ z_6^{m-1m} s & z_6^{m-1m} x_1 & & & & & & & & & \\ ts & tx_1 & & & & & & & & & \end{bmatrix}.$$

Dropping the rank-one constraint yields a semidefinite program whose optimal value is a lower bound on the optimal value of the inner optimization problem. Let the matrices

$$E_{ij} = \frac{1}{2}\left(e_i e_j^T + e_j e_i^T\right)$$

for $1 \leq i \leq j \leq N$ denote standard basis of $\mathbb{S}^N$. We define the index h_{ijf} by the formula

$$h_{ijf} = 2m + 1 + 6\left(\left(\sum_{l=1}^{i-1} m - l\right) + j - i - 1\right) + f.$$

The semidefinite program is then given by

$$
\begin{aligned}
\min \quad & \langle E_{1N}, V\rangle \\
& \langle E_{1+iN}, V\rangle \geq K_i && \forall i \in [m] \\
& \langle E_{m+1+i\,N}, V\rangle \geq K_i' && \forall i \in [m] \\
& \langle E_{1+i\,N} + E_{m+1+i\,N} - E_{1N}, V\rangle \leq K_i'' && \forall i \in [m] \\
& \langle E_{iN} + \cdots + E_{i+6\,N}, V\rangle \geq 1 && \forall i = 2m+2, \ldots, N-6 \\
& \langle -\alpha_f^{ij} E_{1+i\,h_{ijf}} + \alpha_f^{ij} E_{1+j\,h_{ijf}} - \beta_f^{ij} E_{m+1+i\,h_{ijf}} \\
& + \beta_f^{ij} E_{m+1+j\,h_{ijf}} - \gamma_f^{ij} E_{h_{ijf}\,N}, V\rangle \geq 0 && \forall 1 \leq i < j \leq m \,\forall f \in [6] \\
& \langle E_{h_{ijf}\,N} - E_{h_{ijf}\,h_{ijf}}, V\rangle = 0 && \forall 1 \leq i < j \leq m \,\forall f \in [6] \\
& \langle E_{NN}, V\rangle = 1 \\
& V \in \mathbb{S}^N \\
& V \succeq 0.
\end{aligned}
$$

To solve this semidefinite program, we call the MOSEK Optimization Software Version 8.1. Unfortunately, the lower bounds obtained from the semidefinite relaxation are too weak to be applied successfully in our branch-and-bound framework. In order to strengthen the formulation, we propose four different strategies. We describe them using the original variables s, x, y, z, t instead of the matrix variable V for the sake of clarity.

1. Strategy: Add all possible products of the four linear inequalities $x_i \geq K_i$, $y_i \geq K_i'$, $x_i + y_i - s \leq K_i''$ and $z_1^{ij} + \cdots + z_6^{ij} \geq 1$.
2. Strategy: Add all possible products of the four linear inequalities with each binary variable z_f^{ij} and $\left(t - z_f^{ij}\right)$.
3. Strategy: Add all possible products of the four linear inequalities with $x_i \geq x_{i+1}$ for each duplicate of a shape $T_i = T_{i+1}$.
4. Strategy: Add violated triangle inequalities.

The first strategy consists of adding all possible products of the four linear inequalities

$$
\begin{aligned}
& x_i \geq K_i && \forall i \in [m], \\
& y_i \geq K'_i && \forall i \in [m], \\
& x_i + y_i - s \leq K''_i && \forall i \in [m], \\
& z_1^{ij} + \cdots + z_6^{ij} \geq 1 && \forall 1 \leq i < j \leq m
\end{aligned}
$$

from the mixed integer binary formulation. Taking products within these inequalities gives new valid constraints, since the constants $K_i, K'_i, -K''_i$ and 1 are all non-negative. The resulting $3m^2 + 3m\binom{m}{2} + \binom{m}{2}^2$ new constraints are given by

$$
\begin{aligned}
& x_i x_j \geq K_i K_j && \forall 1 \leq i \leq j \leq m, \\
& x_i y_j \geq K_i K'_j && \forall 1 \leq i \leq j \leq m, \\
& x_i(x_j + y_j - s) \leq K_i K''_j && \forall 1 \leq i \leq j \leq m, \\
& x_l(z_1^{ij} + \cdots + z_6^{ij}) \geq K_l && \forall 1 \leq i < j \leq m \quad \forall l \in [m], \\
& y_i y_j \geq K'_i K'_j && \forall 1 \leq i \leq j \leq m, \\
& y_i(x_j + y_j - s) \leq K'_i K''_j && \forall 1 \leq i \leq j \leq m, \\
& y_l(z_1^{ij} + \cdots + z_6^{ij}) \geq K'_l && \forall 1 \leq i < j \leq m \quad \forall l \in [m], \\
& (x_i + y_i - s)(x_j + y_j - s) \geq K''_i K''_j && \forall 1 \leq i \leq j \leq m, \\
& (x_l + y_l - s)(z_1^{ij} + \cdots + z_6^{ij}) \leq K''_l && \forall 1 \leq i < j \leq m \quad \forall l \in [m], \\
& (z_1^{ij} + \cdots + z_6^{ij})(z_1^{ln} + \cdots + z_6^{ln}) \geq 1 && \forall 1 \leq i < j \leq m \quad \forall 1 \leq l < n \leq m.
\end{aligned}
$$

The second strategy is to add all possible products of the four linear inequalities with the binary variables z_f^{ij} and $\left(t - z_f^{ij}\right)$ for all $1 \leq i < j \leq m$ and all $f \in [6]$. This results in the following $48\binom{m}{2}$ new constraints:

$$
\begin{array}{llll}
x_l z_f^{ij} \geq K_i & \forall l \in [m] & \forall 1 \leq i < j \leq m & \forall f \in [6]\,, \\
y_l z_f^{ij} \geq K_i' & \forall l \in [m] & \forall 1 \leq i < j \leq m & \forall f \in [6]\,, \\
(x_i + y_i - s)\, z_f^{ij} \leq K_i'' & \forall l \in [m] & \forall 1 \leq i < j \leq m & \forall f \in [6]\,, \\
\left(z_1^{ln} + \cdots + z_6^{ln}\right) z_f^{ij} \geq 1 & \forall 1 \leq l < n \leq m & \forall 1 \leq i < j \leq m & \forall f \in [6]\,, \\
x_l \left(t - z_f^{ij}\right) \geq K_i & \forall l \in [m] & \forall 1 \leq i < j \leq m & \forall f \in [6]\,, \\
y_l \left(t - z_f^{ij}\right) \geq K_i' & \forall l \in [m] & \forall 1 \leq i < j \leq m & \forall f \in [6]\,, \\
(x_i + y_i - s)\left(t - z_f^{ij}\right) \leq K_i'' & \forall l \in [m] & \forall 1 \leq i < j \leq m & \forall f \in [6]\,.
\end{array}
$$

Because the lower bounds obtained from the semidefinite relaxation are particularly bad when there are several copies of one shape, the third strategy is to add all possible products of the four linear inequalities with the symmetry breaking constraint $x_i \geq x_{i+1}$ whenever T_i and T_{i+1} are of the same shape. The upcoming linear terms are multiplied by the parameter t just like we did to obtain the homogeneous version of the quadratically constrained quadratic program. Whenever $T_l = T_{l+1}$, we add the following new constraints:

$$
\begin{array}{ll}
(x_i - tK_i)\,(x_l - x_{l+1}) \geq 0 & \forall i \in [m]\,, \\
(y_i - tK_i')\,(x_l - x_{l+1}) \geq 0 & \forall i \in [m]\,, \\
(tK_i'' x_i + y_i - s)\,(x_l - x_{l+1}) \geq 0 & \forall i \in [m]\,, \\
\left(z_1^{ij} + \cdots + z_6^{ij} - t\right)(x_l - x_{l+1}) \geq 0 \geq 1 & \forall 1 \leq i < j \leq m.
\end{array}
$$

The fourth strategy is to separate all violated triangle inequalities concerning the binary variables and add them to the program. This is a promising strategy applied for example in [HRW95] and [RRW10]. For the sake of convenience, let Z_l for $l = 1, \ldots, 6\binom{m}{2}$ be the collection of all binary variables z_f^{ij} for $1 \leq i < j \leq m$ and $f \in [6]$. By Z_{ij} we denote the product $Z_i Z_j$ of two binary variables. The triangle inequalities are given by

1. $Z_{ij} \geq 0$,
2. $Z_{ij} \leq Z_{ii}$,
3. $Z_{ii} + Z_{jj} \leq 1 + Z_{ij}$,
4. $Z_{ij} + Z_{il} + Z_{jl} + 1 \geq Z_{ii} + Z_{jj} + Z_{ll}$,
5. $Z_{il} + Z_{jl} \leq Z_{ll} + Z_{ij}$.

They hold for all distinct triplets (i, j, l) with $i, j, l \in \left[6\binom{m}{2}\right]$. Instead of adding all triangle inequalities to the system, we compute the optimal solution of the semidefinite program and check which triangle inequalities are violated by this solution. Subsequently, all violated triangle inequalities are added to the semidefinite program and the procedure is repeated until all triangle inequalities are satisfied. Contrary to our expectations, we only found few violated triangle inequalities and adding them to the program did not change the optimal value. Therefore, we will concentrate on the previous three strategies. The effects of the three strategies applied either solely or in combination are shown in Table 4.6.

The first column shows the different instances for the inner optimization problem. The second column gives the solution of the mixed integer linear formulation (Problem 3.8). This is the exact value of the inner optimization problem, that is the minimum side length s such that the respective multisubset is contained in $\triangle^2(s)$. The third column gives the value of the semidefinite relaxation without any additional constraints (Problem 4.2). This is a lower bound on the minimum side length s. The fourth column gives the value of the semidefinite program combined with all possible products of the four linear inequalities $x_i \geq K_i$, $y_i \geq K_i'$, $x_i + y_i - s \leq K_i''$ and $z_1^{ij} + \cdots + z_6^{ij} \geq 1$ as additional constraints (strategy 1). The fifth column gives the value of the semidefinite program combined with all possible products of the four linear inequalities with each binary variable z_f^{ij} and $\left(t - z_f^{ij}\right)$ as additional constraints (strategy 2). The sixth column gives the value of the semidefinite program combined with all possible products of the four linear inequalities with $x_i \geq x_{i+1}$ for each duplicate of a shape $T_i = T_{i+1}$ as additional constraints (strategy 3). The last four columns give the value of the semidefinite program together with combinations of the three strategies.

Instance	MILP	SDP	1	2	3	1&2	1&3	2&3	1 - 3
$2 \times T_1$	2.0	1.0	1.0	1.6	1.0	1.6	1.0	1.6	1.6
$3 \times T_1$	2.0	1.0	1.0	1.6	1.0	1.6	1.0	1.6	1.6
$4 \times T_1$	2.5	1.0	1.0	1.6	1.0	1.6	1.0	1.6	1.6
$2 \times T_2$	2.5	2.0	2.0	2.5	2.0	2.5	1.0	2.5	2.5
$3 \times T_2$	3.0	2.0	2.0	2.5	2.0	2.5	2.0	2.5	2.5
$4 \times T_2$	3.5	2.0	2.0	2.5	2.0	2.5	2.0	2.5	2.5
$2 \times \{T_1, T_2\}$	2.5	2.0	2.0	2.5	2.0	2.5	2.0	2.5	2.5
T_1, T_2, T_3, T_4	3.0	2.0	2.0	2.54	2.0	2.54	2.0	2.54	2.54
$2 \times \{T_1, T_2, T_3, T_4\}$	3.5	2.0	2.0	2.6	2.0	2.6	2.0	2.59	2.6

Table 4.6: Optimal values of the mixed integer linear program, the semidefinite program and the semidefinite program combined with the first three improvement strategies for different inner optimization instances

We can see that the second strategy generates the best improvement in all cases. However, the gap between the exact value coming from the mixed integer linear program and the lower bound coming from the semidefinite relaxation is quite large and even increases for instances of larger cardinality. Apart from the bounds being too weak, the computation time of the semidefinite program is also higher than the computation time of the mixed integer linear program. We will circumstantiate the last two statements when comparing the three approaches in the last subsection.

We continue with the description of the second approach to model the inner optimization problem as a quadratically constrained quadratic program.

4.3.2 Second Approach

This approach uses the separating hyperplane theorem.

Theorem 4.3 (Separating Hyperplane Theorem [VB96])**.**
Let $C, D \subseteq \mathbb{R}^n$ be nonempty disjoint convex sets. Then there exist a nonzero vector $a \in \mathbb{R}^n$ and a real number b such that $a^T x \leq b$ for all $x \in C$ and $a^T x \geq b$ for all $x \in D$.

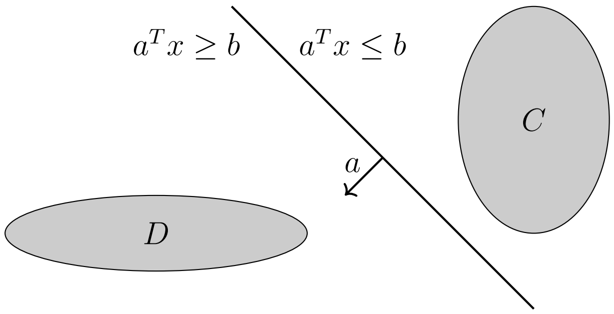


Figure 4.2: The hyperplane $\left\{x \in \mathbb{R}^n \mid a^T x = b\right\}$ separates the convex sets C and D.

The hyperplane $\left\{x \in \mathbb{R}^n \mid a^T x = b\right\}$ visualized in Figure 4.2 is called *separating hyperplane* for the sets C and D. We can directly apply Theorem 4.3 to the disjointness condition

$$\overline{T_i + t_i} \cap \overline{T_j + t_j} = \emptyset.$$

Recall, that the triangles T_i are defined as open sets and we are working with their closures instead. Each triangle $\overline{T_i + t_i}$ is given as the convex hull of its vertices

$$\overline{T_i + t_i} = \operatorname{conv}\left(\left\{\left(\begin{smallmatrix} x_i \\ y_i \end{smallmatrix}\right), \left(\begin{smallmatrix} x_i+a_i \\ y_i+b_i \end{smallmatrix}\right), \left(\begin{smallmatrix} x_i+c_i \\ y_i+d_i \end{smallmatrix}\right)\right\}\right).$$

By Theorem 4.3, the two sets are disjoint if there exist a nonzero vector $\left(\begin{smallmatrix} \alpha^{ij} \\ \beta^{ij} \end{smallmatrix}\right)$ and a real number γ^{ij} such that

$$\begin{aligned}
\alpha^{ij} x_i + \beta^{ij} y_i &\leq \gamma^{ij}, & \alpha^{ij} x_j + \beta^{ij} y_j &\geq \gamma^{ij}, \\
\alpha^{ij}(x_i + a_i) + \beta^{ij}(y_i + b_i) &\leq \gamma^{ij}, & \alpha^{ij}(x_j + a_j) + \beta^{ij}(y_j + b_j) &\geq \gamma^{ij}, \\
\alpha^{ij}(x_i + c_i) + \beta^{ij}(y_i + d_i) &\leq \gamma^{ij}, & \alpha^{ij}(x_j + c_j) + \beta^{ij}(y_j + d_j) &\geq \gamma^{ij}.
\end{aligned}$$

Because of convexity, the inequalities hold for the entire set $\overline{t_i + T_i}$ and $\overline{t_j + T_j}$ if they are satisfied by their vertices. To ensure that $\binom{\alpha^{ij}}{\beta^{ij}}$ is nonzero, we impose the additional constraint $(\alpha^{ij})^2 + (\beta^{ij})^2 = 1$. Together with the three containment constraints we obtain the following quadratically constrained quadratic program.

Problem 4.4 (Inner Optimization Problem - Quadratically Constrained Quadratic Formulation 2)**.**

$$
\begin{aligned}
\min \quad & s \\
& tx_i \geq K_i && \forall i \in [m] \\
& ty_i \geq K'_i && \forall i \in [m] \\
& tx_i + ty_i - ts \leq K''_i && \forall i \in [m] \\
& \left(\alpha^{ij}\right)^2 + \left(\beta^{ij}\right)^2 = 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij} x_i + \beta^{ij} y_i \leq t\gamma^{ij} && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(x_i + ta_i) + \beta^{ij}(y_i + tb_i) \leq t\gamma^{ij} && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(x_i + tc_i) + \beta^{ij}(y_i + td_i) \leq t\gamma^{ij} && \forall 1 \leq i < j \leq m \\
& \alpha^{ij} x_j + \beta^{ij} y_j \geq t\gamma^{ij} && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(x_j + ta_j) + \beta^{ij}(y_j + tb_j) \geq t\gamma^{ij} && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(x_j + tc_j) + \beta^{ij}(y_j + td_j) \geq t\gamma^{ij} && \forall 1 \leq i < j \leq m \\
& t^2 = 1 \\
& s \in \mathbb{R}, \quad x, y \in \mathbb{R}^m, \quad \alpha, \beta, \gamma \in \mathbb{R}^{\binom{m}{2}}, \quad t \in \mathbb{R}
\end{aligned}
$$

As before, we introduced an additional variable $t \in \mathbb{R}$ with $t^2 = 1$ to homogenize the program. Compared to the first quadratically constrained quadratic problem (Problem 4.2), we have $2 + 2m + 3\binom{m}{2}$ variables instead of $2 + 2m + 6\binom{m}{2}$ and $1 + 3m + 7\binom{m}{2}$ constraints instead of $1 + 3m + 13\binom{m}{2}$.

We replace the vector of variables

$$v = \begin{pmatrix} s & x & y & \alpha & \beta & \gamma & t \end{pmatrix}^T \in \mathbb{R}^N,$$

where $N = 2 + 2m + 3\binom{m}{2}$, by the matrix $V = vv^T \in \mathbb{S}^N$ with $\operatorname{rank}(V) = 1$, and rewrite the objective function and constraints by using the identity $v^T A v = \langle A, vv^T \rangle = \langle A, V \rangle$. Dropping the rank-one constraint yields a semidefinite program whose optimal value is a lower bound on the value of the inner optimization problem. For the inner optimization instances that were investigated in Table 4.6 the optimal values of the semidefinite relaxation of Problem 4.4 coincided with the optimal values of the semidefinite relaxation of Problem 4.2. We will extend the results to further inner optimization instances in the last subsection.

We continue with the description of the third approach to model the inner optimization problem as a quadratically constrained quadratic program.

4.3.3 Third Approach

This approach utilizes Farkas Lemma.

Theorem 4.5 (Farkas Lemma [Far02])**.**
Let $A \in \mathbb{R}^{m \times n}$ be a matrix and $b \in \mathbb{R}^m$ be a vector. There exists a vector $x \in \mathbb{R}^n$ such that $x \geq 0$ and $Ax = b$ if and only if there does not exist a vector $y \in \mathbb{R}^m$ such that $y^T A \geq 0$ and $y^T b = -1$.

denote the $2n$-dimensional open ball of radius $\sqrt{\frac{r}{\pi}}$ and the $2n$-dimensional open cylinder of radius $\sqrt{\frac{s}{\pi}}$, respectively. Gromov's Non-squeezing theorem states that one cannot symplectically embed $B^{2n}(r)$ into $Z^{2n}(s)$ unless the radius r of the ball is less than or equal to the radius s of the cylinder.

$$T_i \ominus T_j = \operatorname{int}\left(\operatorname{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}a_i\\b_i\end{smallmatrix}\right), \left(\begin{smallmatrix}c_i\\d_i\end{smallmatrix}\right), \left(\begin{smallmatrix}-a_j\\-b_j\end{smallmatrix}\right), \left(\begin{smallmatrix}a_i-a_j\\b_i-b_j\end{smallmatrix}\right), \right.\right.\right.$$
$$\left.\left.\left. \left(\begin{smallmatrix}c_i-a_j\\d_i-b_j\end{smallmatrix}\right), \left(\begin{smallmatrix}-c_j\\-d_j\end{smallmatrix}\right), \left(\begin{smallmatrix}a_i-c_j\\b_i-d_j\end{smallmatrix}\right), \left(\begin{smallmatrix}c_i-c_j\\d_i-d_j\end{smallmatrix}\right)\right\}\right)\right).$$

The difference $t_j - t_i$ is not an element of the Minkowski difference $\overline{T_i \ominus T_j}$ if and only if it cannot be written as a convex combination of its vertices. More formally, there do not exist $\lambda_1^{ij}, \ldots, \lambda_9^{ij} \geq 0$ with $\lambda_1^{ij} + \cdots + \lambda_9^{ij} = 1$ such that

$$\begin{aligned}\left(\begin{smallmatrix}x_j-x_i\\y_j-y_i\end{smallmatrix}\right) =& \lambda_1^{ij}\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right) + \lambda_2^{ij}\left(\begin{smallmatrix}a_i\\b_i\end{smallmatrix}\right) + \lambda_3^{ij}\left(\begin{smallmatrix}c_i\\d_i\end{smallmatrix}\right) + \lambda_4^{ij}\left(\begin{smallmatrix}-a_j\\-b_j\end{smallmatrix}\right) + \lambda_5^{ij}\left(\begin{smallmatrix}a_i-a_j\\b_i-b_j\end{smallmatrix}\right)\\ &+ \lambda_6^{ij}\left(\begin{smallmatrix}c_i-a_j\\d_i-b_j\end{smallmatrix}\right) + \lambda_7^{ij}\left(\begin{smallmatrix}-c_j\\-d_j\end{smallmatrix}\right) + \lambda_8^{ij}\left(\begin{smallmatrix}a_i-c_j\\b_i-d_j\end{smallmatrix}\right) + \lambda_9^{ij}\left(\begin{smallmatrix}c_i-c_j\\d_i-d_j\end{smallmatrix}\right).\end{aligned}$$

Written in matrix form, this means that there does not exist a vector $\lambda^{ij} \in \mathbb{R}^9$ such that $\lambda^{ij} \geq 0$ and

$$\begin{bmatrix} 0 & a_i & c_i & -a_j & a_i - a_j & c_i - a_j & -c_j & a_i - c_j & c_i - c_j \\ 0 & b_i & d_i & -b_j & b_i - b_j & d_i - b_j & -d_j & b_i - d_j & d_i - d_j \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \lambda^{ij} = \begin{bmatrix} x_j - x_i \\ y_j - y_i \\ 1 \end{bmatrix}.$$

Then, by Farkas Lemma, there must exist a vector $\mu^{ij} \in \mathbb{R}^3$ such that

$$\left(\mu^{ij}\right)^T \begin{bmatrix} 0 & a_i & c_i & -a_j & a_i - a_j & c_i - a_j & -c_j & a_i - c_j & c_i - c_j \\ 0 & b_i & d_i & -b_j & b_i - b_j & d_i - b_j & -d_j & b_i - d_j & d_i - d_j \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \geq 0$$

and

$$\left(\mu^{ij}\right)^T \begin{bmatrix} x_j - x_i \\ y_j - y_i \\ 1 \end{bmatrix} = -1.$$

From the second constraint we can derive $\mu_3^{ij} = -1 - \mu_1^{ij}(x_j - x_i) - \mu_2^{ij}(y_j - y_i)$. By plugging this expression into the first constraint and renaming μ_1^{ij} to α^{ij} and μ_2^{ij} to β^{ij}, we obtain the following quadratically constrained quadratic program.

Problem 4.6 (Inner Optimization Problem - Quadratically Constrained Quadratic Formulation 3)**.**

$$
\begin{aligned}
\min \quad & s \\
& tx_i \geq K_i && \forall i \in [m] \\
& ty_i \geq K'_i && \forall i \in [m] \\
& tx_i + ty_i - ts \leq K''_i && \forall i \in [m] \\
& -\alpha^{ij}(x_j - x_i) - \beta^{ij}(y_j - y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(ta_i - x_j + x_i) + \beta^{ij}(tb_i - y_j + y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(tc_i - x_j + x_i) + \beta^{ij}(td_i - y_j + y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(-ta_j - x_j + x_i) + \beta^{ij}(-tb_j - y_j + y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(t(a_i - a_j) - x_j + x_i) + \beta^{ij}(t(b_i - b_j) - y_j + y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(t(c_i - a_j) - x_j + x_i) + \beta^{ij}(t(d_i - b_j) - y_j + y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(-tc_j - x_j + x_i) + \beta^{ij}(-td_j - y_j + y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(t(a_i - c_j) - x_j + x_i) + \beta^{ij}(t(b_i - d_j) - y_j + y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha^{ij}(t(c_i - c_j) - x_j + x_i) + \beta^{ij}(t(d_i - d_j) - y_j + y_i) \geq 1 && \forall 1 \leq i < j \leq m \\
& t^2 = 1 \\
& s \in \mathbb{R}, \quad x, y \in \mathbb{R}^m, \quad \alpha, \beta \in \mathbb{R}^{\binom{m}{2}}, \quad t \in \mathbb{R}
\end{aligned}
$$

Again, we introduced an additional variable $t \in \mathbb{R}$ with $t^2 = 1$ to homogenize the program. Compared to Problem 4.2 and Problem 4.4 this quadratically constrained quadratic problem has $2 + 2m + 2\binom{m}{2}$ variables instead of $2 + 2m + 6\binom{m}{2}$ and $2 + 2m + 3\binom{m}{2}$, respectively. Furthermore, it has $1 + 3m + 9\binom{m}{2}$ constraints instead of $1 + 3m + 13\binom{m}{2}$ and $1 + 3m + 7\binom{m}{2}$, respectively. As before, we relax Problem 4.6 to a semidefinite program. For the inner optimization instances that were investigated in Table 4.6 the optimal values of the semidefinite relaxation of Problem 4.6 also coincided with the optimal values of the semidefinite relaxations of Problem 4.2 and Problem 4.4. We will now have a look at some other inner optimization instances and also compare the computation times of the three approaches.

4.3.4 Comparison of the Approaches

Although the origin of the three quadratic formulations is quite different, the values of the corresponding semidefinite relaxations are exactly the same for the inner optimization instances that we have investigated so far. We will now compare the optimal values for some of the hard inner optimization instances that we have already encountered in Section 4.2.

Table 4.7 shows the optimal value of the mixed integer linear program (MILP) and the three semidefinite programs (SDP 1-3). Supplementary, Table 4.8 shows the corresponding computation times of the four optimization problems.

Instance	Optimal Value			
	MILP	SDP 1	SDP 2	SDP 3
$7 \times T_1, 2 \times T_3, T_{10}$	3.89	3.00	1.00	1.00
$6 \times T_1, T_6, 2 \times T_7, T_{13}$	3.92	3.00	2.00	2.00
$8 \times T_1, T_4, T_6, T_7$	3.94	2.00	2.00	2.00
$6 \times T_1, 2 \times T_2, T_3, T_5, T_7$	3.83	2.00	2.00	2.00
$6 \times T_1, 2 \times T_2, 2 \times T_4, T_5$	3.81	2.00	2.00	2.00
$5 \times T_1, 2 \times T_2, T_3, T_7, 2 \times T_8$	3.80	2.00	2.00	2.00
$5 \times T_1, 2 \times T_2, T_3, T_7, 2 \times T_8$	3.80	2.00	2.00	2.00
$5 \times T_1, T_2, T_3, T_4, 3 \times T_5$	3.83	2.00	2.00	2.00
$4 \times T_1, 2 \times T_2, 3 \times T_3, T_{11}, T_{12}$	4.00	3.00	3.00	3.00
$4 \times T_1, 3 \times T_2, 2 \times T_3, T_4, T_7, 2 \times T_8$	3.92	3.00	2.00	2.00

Table 4.7: Optimal values of the mixed integer linear program and the three semidefinite programs for different inner optimization instances

Computation Time			
MILP	SDP 1	SDP 2	SDP 3
0 : 13 : 26.6	0 : 00 : 03.7	0 : 00 : 00.4	0 : 00 : 00.3
0 : 07 : 06.2	0 : 00 : 04.2	0 : 00 : 00.3	0 : 00 : 00.3
1 : 13 : 16.2	0 : 00 : 08.0	0 : 00 : 00.7	0 : 00 : 00.4
1 : 08 : 18.9	0 : 00 : 07.1	0 : 00 : 00.7	0 : 00 : 00.5
0 : 56 : 57.6	0 : 00 : 06.1	0 : 00 : 00.6	0 : 00 : 00.6
0 : 46 : 36.9	0 : 00 : 06.5	0 : 00 : 00.6	0 : 00 : 00.4
0 : 45 : 26.9	0 : 00 : 06.8	0 : 00 : 00.7	0 : 00 : 00.5
0 : 09 : 21.5	0 : 00 : 07.5	0 : 00 : 00.9	0 : 00 : 00.4
1 : 17 : 29.6	0 : 00 : 12.4	0 : 00 : 00.9	0 : 00 : 00.6
0 : 19 : 24.2	0 : 00 : 17.3	0 : 00 : 01.2	0 : 00 : 00.8

Table 4.8: Computation time of the mixed integer linear program and the three semidefinite programs for different inner optimization instances given in the format "hh:mm:ss". The instances are in the same order as in Table 4.7.

For the first two and the last inner optimization instances, the optimal values of the three semidefinite relaxations differ. For the remaining inner optimization instances the optimal values coincide. The computation time of the first semidefinite relaxation is slightly larger than the computation time of the other two relaxations but much faster than the computation time of the mixed integer linear program. However, the gap between the exact value of the inner optimization problem obtained by the mixed integer linear program and the best lower bound obtained by the semidefinite relaxations is too large for a successful application in our branch and bound framework.

We also tried to compute the strengthened formulation of the first semidefinite relaxation as described in Subsection 4.3.1. For the first and the second inner optimization instance the optimal value of the strengthened semidefinite program did not change compared to the original semidefinite program but the computation time increased to more than six hours and more than seven hours, respectively. Consequently, it also exceeds the computation time of the mixed integer linear program by far. For the other inner optimization instances we were not able to compute the strengthened semidefinite relaxation due to the increased complexity of the problem.

Albeit semidefinite relaxation is a powerful tool to obtain strong bounds for many optimization problems, it has turned out to be little promising for computing satisfying bounds for our setup.

4.4 McCormick Relaxation

We want to apply the McCormick relaxation technique described in Section 2.5 to the inner optimization problem. For this purpose, we need to model the inner optimization problem as a mixed integer bilinear program. We have two approaches for obtaining such a formulation.

4.4.1 First Approach

Consider the following mixed integer bilinear formulation of the inner optimization problem.

Problem 4.7 (Inner Optimization Problem - Mixed Integer Bilinear Formulation 1)**.**

$$
\begin{aligned}
\min \quad & s \\
& x_i \geq K_i && \forall i \in [m] \\
& y_i \geq K'_i && \forall i \in [m] \\
& x_i + y_i - s \leq K''_i && \forall i \in [m] \\
& z_1^{ij} + \cdots + z_6^{ij} \geq 1 && \forall 1 \leq i < j \leq m \\
& z_f^{ij} \left(\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) - \gamma_f^{ij} \right) \geq 0 && \forall 1 \leq i < j \leq m \quad \forall f \in [6] \\
& s \in \mathbb{R}, \quad x, y \in \mathbb{R}^m, \quad z \in \{0,1\}^{6\binom{m}{2}}
\end{aligned}
$$

This is the mixed integer linear formulation of the inner optimization problem

(Problem 3.8) with the difference that we replaced the linear Big-M-inequality

$$\left(\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i)\right) \geq \gamma_f^{ij} - M\left(1 - z_f^{ij}\right)$$

by the quadratic inequality

$$z_f^{ij}\left(\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) - \gamma_f^{ij}\right) \geq 0.$$

By expanding the quadratic inequality we obtain

$$\alpha_f^{ij} x_j z_f^{ij} - \alpha_f^{ij} x_i z_f^{ij} + \beta_f^{ij} y_j z_f^{ij} - \beta_f^{ij} y_i z_f^{ij} - \gamma_f^{ij} z_f^{ij} \geq 0.$$

This inequality consists of the four bilinear terms

$$XI_f^{ij} := x_i z_f^{ij}, \qquad XJ_f^{ij} := x_j z_f^{ij}, \qquad YI_f^{ij} := y_i z_f^{ij}, \qquad YJ_f^{ij} := y_j z_f^{ij}.$$

We will construct McCormick envelopes for each of these four products. From the containment constraints we know that a lower bound on x_i and y_i is given by K_i and K_i', respectively, for all $i \in [m]$. Each inner optimization problem corresponds to a node in the branch-and-bound tree that is equipped with a global upper bound $\hat{s}$ on the variable s. Since $x_i \leq s$ and $y_i \leq s$, this is also an upper bound on x_i and y_i for all $i \in [m]$. The variables z_f^{ij} are of binary nature and consequently have a lower bound equal to zero and an upper bound equal to one for all $1 \leq i < j \leq m$ and $f \in [6]$. Summarized, the variables are bounded by

$$\begin{aligned} K_i \leq x_i \leq \hat{s} & \qquad \forall i \in [m], \\ K_i' \leq y_i \leq \hat{s} & \qquad \forall i \in [m], \\ 0 \leq z_f^{ij} \leq 1 & \qquad \forall 1 \leq i < j \leq m \quad \forall f \in [6]. \end{aligned}$$

Hence, the McCormick relaxation of Problem 4.7 is given by

$$
\begin{aligned}
\min \quad & s \\
& x_i \geq K_i && \forall i \in [m] \\
& y_i \geq K_i' && \forall i \in [m] \\
& x_i + y_i - s \leq K_i'' && \forall i \in [m] \\
& z_1^{ij} + \cdots + z_6^{ij} \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha_f^{ij} \left(XJ_f^{ij} - XI_f^{ij} \right) \\
& + \beta_f^{ij} \left(YJ_f^{ij} - YI_f^{ij} \right) - \gamma_f^{ij} z_f^{ij} \geq 0 && \forall 1 \leq i < j \leq m \quad \forall f \in [6] \\
& \text{(MC 1)} && \forall 1 \leq i < j \leq m \quad \forall f \in [6] \\
& s \in \mathbb{R}, \quad x, y \in \mathbb{R}^m, \quad z \in \{0,1\}^{6\binom{m}{2}}, \\
& XI, XJ, YI, YJ \in \mathbb{R}^{6\binom{m}{2}},
\end{aligned}
$$

where the McCormick constraints are of the form

$$
\begin{aligned}
XI_f^{ij} &\geq K_i z_f^{ij}, & XI_f^{ij} &\leq x_i + K_i z_f^{ij} - K_i, \\
XI_f^{ij} &\geq x_i + \hat{s} z_f^{ij} - \hat{s}, & XI_f^{ij} &\leq \hat{s} z_f^{ij}, \\
XJ_f^{ij} &\geq K_j z_f^{ij}, & XJ_f^{ij} &\leq x_j + K_j z_f^{ij} - K_j, \\
XJ_f^{ij} &\geq x_j + \hat{s} z_f^{ij} - \hat{s}, & XJ_f^{ij} &\leq \hat{s} z_f^{ij}, \\
YI_f^{ij} &\geq K_i' z_f^{ij}, & YI_f^{ij} &\leq y_i + K_i' z_f^{ij} - K_i', \\
YI_f^{ij} &\geq y_i + \hat{s} z_f^{ij} - \hat{s}, & YI_f^{ij} &\leq \hat{s} z_f^{ij}, \\
YJ_f^{ij} &\geq K_j' z_f^{ij}, & YJ_f^{ij} &\leq y_j + K_j' z_f^{ij} - K_j', \\
YJ_f^{ij} &\geq y_j + \hat{s} z_f^{ij} - \hat{s}, & YJ_f^{ij} &\leq \hat{s} z_f^{ij}.
\end{aligned}
\tag{MC 1}
$$

In general, the McCormick relaxation just gives a lower bound on the optimal solution of the original problem. In our case, the McCormick relaxation gives the exact value of the inner optimization problem due to the binary nature of the variables

z_f^{ij}. To see this, consider the binary term $XI_f^{ij} = z_f^{ij} x_i$ with the four McCormick envelopes

$$
\begin{aligned}
XI_f^{ij} &\geq K_i z_f^{ij}, \\
XI_f^{ij} &\geq x_i + \hat{s} z_f^{ij} - \hat{s}, \\
XI_f^{ij} &\leq x_i + K_i z_f^{ij} - K_i, \\
XI_f^{ij} &\leq \hat{s} z_f^{ij}.
\end{aligned}
$$

On the one hand, the first inequality and the third inequality ensure that $XI_f^{ij} = 0$ if $z_f^{ij} = 0$. On the other hand, the second and last inequality ensure that $XI_f^{ij} = x_i$ if $z_f^{ij} = 1$. So in both cases, the identity $XI_f^{ij} = z_f^{ij} x_i$ is correct. The same holds true for the other binary terms $XJ_f^{ij} = z_f^{ij} x_j$, $YI_f^{ij} = z_f^{ij} y_i$ and $YJ_f^{ij} = z_f^{ij} y_j$.

Compared to the mixed integer linear formulation of Problem 3.8, this McCormick relaxation uses $24\binom{m}{2}$ additional variables and $96\binom{m}{2}$ additional constraints. Hence, it is unsurprising that the computation time of the McCormick relaxation is greater than the computation time of Problem 3.8. But we have another approach that uses less variables and less constraints than the first McCormick relaxation.

4.4.2 Second Approach

Let us consider the quadratic constraint

$$z_f^{ij} \left(\alpha_f^{ij} (x_j - x_i) + \beta_f^{ij} (y_j - y_i) - \gamma_f^{ij} \right) \geq 0 \quad \forall f \in [6]$$

for any pair i and j with $1 \leq i < j \leq m$. We can replace this constraint by summing up over all indices $f \in [6]$. It then becomes

$$\left(\sum_{f=1}^{6} \alpha_f^{ij} z_f^{ij} \right) (x_j - x_i) + \left(\sum_{f=1}^{6} \beta_f^{ij} z_f^{ij} \right) (y_j - y_i) - \left(\sum_{f=1}^{6} \gamma_f^{ij} z_f^{ij} \right) \geq 0.$$

These two constraints are equivalent for our problem. It is easy to see that the first constraint implies the second one. To see the other direction, suppose that the second constraint is satisfied. The inequality

$$z_1^{ij} + \cdots + z_6^{ij} \geq 1$$

ensures that there exists an index $f^\star \in [6]$ with $z_{f^\star}^{ij} = 1$. Therefore, the corresponding summand satisfies

$$z_{f^\star}^{ij} \left(\alpha_{f^\star}^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) - \gamma_{f^\star}^{ij} \right) \geq 0.$$

By setting the binary variables z_f^{ij} to 0 for all $f \neq f^\star$ the first constraint is also satisfied. For every $1 \leq i < j \leq m$ we define two new variables a^{ij} and b^{ij} by

$$a^{ij} := \sum_{f=1}^{6} \alpha_f^{ij} z_f^{ij},$$
$$b^{ij} := \sum_{f=1}^{6} \beta_f^{ij} z_f^{ij}.$$

The new constraint then simplifies to

$$a^{ij}(x_j - x_i) + b^{ij}(y_j - y_i) - \left(\sum_{f=1}^{6} \gamma_f^{ij} z_f^{ij} \right) \geq 0$$

for all $1 \leq i < j \leq m$. Hence, we obtain the second mixed integer bilinear formulation of the inner optimization problem.

Problem 4.8 (Inner Optimization Problem - Mixed Integer Bilinear Formulation 2)**.**

$$
\begin{aligned}
\min \quad & s \\
& x_i \geq K_i && \forall i \in [m] \\
& y_i \geq K_i' && \forall i \in [m] \\
& x_i + y_i - s \leq K_i'' && \forall i \in [m] \\
& z_1^{ij} + \cdots + z_6^{ij} \geq 1 && \forall 1 \leq i < j \leq m \\
& a^{ij}(x_j - x_i) + b^{ij}(y_j - y_i) - \left(\sum_{f=1}^{6} \gamma_f^{ij} z_f^{ij} \right) \geq 0 && \forall 1 \leq i < j \leq m \\
& a^{ij} - \sum_{f=1}^{6} \alpha_f^{ij} z_f^{ij} = 0 && \forall 1 \leq i < j \leq m \\
& b^{ij} - \sum_{f=1}^{6} \beta_f^{ij} z_f^{ij} = 0 && \forall 1 \leq i < j \leq m \\
& s \in \mathbb{R}, \quad x, y \in \mathbb{R}^m, \quad z \in \{0,1\}^{6\binom{m}{2}}, \quad a, b \in \mathbb{R}^{\binom{m}{2}}
\end{aligned}
$$

Now we can derive the McCormick envelopes for the four bilinear terms

$$
a_1^{ij} := a^{ij} x_i, \qquad a_2^{ij} := a^{ij} x_j, \qquad b_1^{ij} := b^{ij} y_i, \qquad b_2^{ij} := b^{ij} y_j.
$$

Let

$$
\alpha_L^{ij} = \sum_{f:\alpha_f^{ij}<0} \alpha_f^{ij} \qquad \text{and} \qquad \alpha_U^{ij} = \sum_{f:\alpha_f^{ij}>0} \alpha_f^{ij}
$$

denote the sum over all indices $f \in [6]$ for which the value of α_f^{ij} is negative or positive, respectively. Likewise, let

$$
\beta_L^{ij} = \sum_{f:\beta_f^{ij}<0} \beta_f^{ij} \qquad \text{and} \qquad \beta_U^{ij} = \sum_{f:\beta_f^{ij}>0} \beta_f^{ij}
$$

denote the sum over all indices $f \in [6]$ for which the value of β_f^{ij} is negative or positive, respectively. The bounds on the variables from the bilinear terms are given by

$$
\begin{aligned}
&K_i \leq x_i \leq \hat{s} && \forall i \in [m]\,, \\
&K_i' \leq y_i \leq \hat{s} && \forall i \in [m]\,, \\
&\alpha_L^{ij} \leq \alpha^{ij} \leq \alpha_U^{ij} && \forall 1 \leq i < j \leq m, \\
&\beta_L^{ij} \leq \beta^{ij} \leq \beta_U^{ij} && \forall 1 \leq i < j \leq m,
\end{aligned}
$$

where $\hat{s}$ is the global upper bound on s as described in the previous McCormick relaxation. Hence, the second McCormick relaxation of Problem 4.7 is given by

$$
\begin{aligned}
\min \quad & s \\
& x_i \geq K_i && \forall i \in [m] \\
& y_i \geq K_i' && \forall i \in [m] \\
& x_i + y_i - s \leq K_i'' && \forall i \in [m] \\
& z_1^{ij} + \cdots + z_6^{ij} \geq 1 && \forall 1 \leq i < j \leq m \\
& a^{ij}\left(x_j - x_i\right) + b^{ij}\left(y_j - y_i\right) - \sum_{f=1}^{6} \gamma_f^{ij} z_f^{ij} \geq 0 && \forall 1 \leq i < j \leq m \\
& a^{ij} = \sum_{f=1}^{6} \alpha_f^{ij} z_f^{ij} && \forall 1 \leq i < j \leq m \\
& b^{ij} = \sum_{f=1}^{6} \beta_f^{ij} z_f^{ij} && \forall 1 \leq i < j \leq m \\
& \text{(MC 2)} && \forall 1 \leq i < j \leq m \quad \forall f \in [6] \\
& s \in \mathbb{R}, \quad x, y \in \mathbb{R}^m, \quad z \in \{0,1\}^{6\binom{m}{2}} \\
& a, a_1, a_2, b, b_1, b_2 \in \mathbb{R}^{\binom{m}{2}},
\end{aligned}
$$

where the McCormick constraints are of the form

$$
\begin{aligned}
&a_1^{ij} \geq \alpha_L^{ij} x_i + K_i a^{ij} - K_i \alpha_L^{ij} \quad , && a_1^{ij} \leq \alpha_U^{ij} x_i + K_i a^{ij} - K_i \alpha_U^{ij}, \\
&a_1^{ij} \geq \alpha_U^{ij} x_i + \hat{s} a^{ij} - \hat{s} \alpha_U^{ij}, && a_1^{ij} \leq \alpha_L^{ij} x_i + \hat{s} a^{ij} - \hat{s} \alpha_L^{ij}, \\
&a_2^{ij} \geq \alpha_L^{ij} x_j + K_j a^{ij} - K_j \alpha_L^{ij}, && a_2^{ij} \leq \alpha_U^{ij} x_j + K_j a^{ij} - K_j \alpha_U^{ij}, \\
&a_2^{ij} \geq \alpha_U^{ij} x_j + \hat{s} a^{ij} - \hat{s} \alpha_U^{ij}, && a_2^{ij} \leq \alpha_L^{ij} x_j + \hat{s} a^{ij} - \hat{s} \alpha_L^{ij}, \\
&b_1^{ij} \geq \beta_L^{ij} y_i + K_i' b^{ij} - K_i' \beta_L^{ij}, && b_1^{ij} \leq \beta_U^{ij} y_i + K_i' b^{ij} - K_i' \beta_U^{ij}, \\
&b_1^{ij} \geq \beta_U^{ij} y_i + \hat{s} b^{ij} - \hat{s} \beta_U^{ij}, && b_1^{ij} \leq \beta_L^{ij} y_i + \hat{s} b^{ij} - \hat{s} \beta_L^{ij}, \\
&b_2^{ij} \geq \beta_L^{ij} y_j + K_j' b^{ij} - K_j' \beta_L^{ij}, && b_2^{ij} \leq \beta_U^{ij} y_j + K_j' b^{ij} - K_j' \beta_U^{ij}, \\
&b_2^{ij} \geq \beta_U^{ij} y_j + \hat{s} b^{ij} - \hat{s} \beta_U^{ij}, && b_2^{ij} \leq \beta_L^{ij} y_j + \hat{s} b^{ij} - \hat{s} \beta_L^{ij}.
\end{aligned}
\tag{MC 2}
$$

The optimal value of this mixed integer linear program gives a lower bound on the optimal value of the inner optimization problem. For all investigated instances the optimal values are exactly the same as the optimal values of the three semidefinite programs described in the previous section. Consequently, this McCormick relaxation is also unsuitable for our branch-and-bound framework.

Chapter 5

Algorithmic Approach for $s_k\left(P^6(r),\omega_0\right)$

In this chapter we are going to extend the algorithmic approach for the computation of $s_k\left(P^4(r),\omega_0\right)$ to the next higher dimension and compute the k-simplex packing width of the six-dimensional open prism $P^6(r) = \llcorner^3(r) \times \mathbb{T}^3$. Since we are dealing with simplices of dimension three, we will use the terms tetrahedron and simplex interchangeably.

5.1 Outer Optimization Problem

In dimension six the outer optimization problem becomes

Problem 5.1 (Outer Optimization Problem).
Given $k \in \mathbb{N}$, determine $s_k\left(P^6(1),\omega_0\right)$.

Analogously to Chapter 3, we apply Theorem 1.13 to obtain the following equivalent formulation of Problem 5.1.

Problem 5.2 (Outer Optimization Problem - Combinatoral Formulation)**.**
Given $k \in \mathbb{N}$*, determine the minimum side length* s *such that there exist matrices* $A_1, \dots, A_k \in \mathrm{GL}_3(\mathbb{Z})$ *and vectors* $t_1, \dots, t_k \in \mathbb{R}^3$ *satisfying*

$$A_i\left(\triangle^3(1)\right) + t_i \subseteq \triangle^3(s) \qquad \forall i \in [k] \qquad \textit{(containment condition)},$$

$$\left(A_i\left(\triangle^3(1)\right) + t_i\right) \cap \left(A_j\left(\triangle^3(1)\right) + t_j\right) = \emptyset \quad \forall 1 \leq i < j \leq k \quad \textit{(disjointness condition)}.$$

We denote the minimum side length s from Problem 5.2 by $s_k^{\triangle}$. Then we have the relation

$$s_k\left(P^6(1), \omega_0\right) = \frac{1}{s_k^{\triangle}}.$$

As visualized in Figure 5.1 and Figure 5.2, we can create an 8-tetrahedron packing with side length two that is dense and therefore optimal. This packing consists of four different shapes each represented by a different colour.

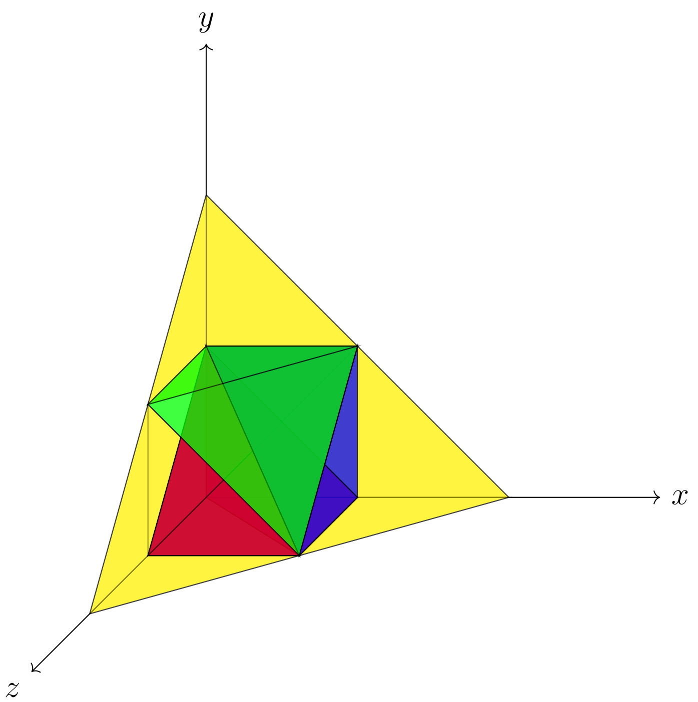


Figure 5.1: Optimal 8-tetrahedron packing in front view

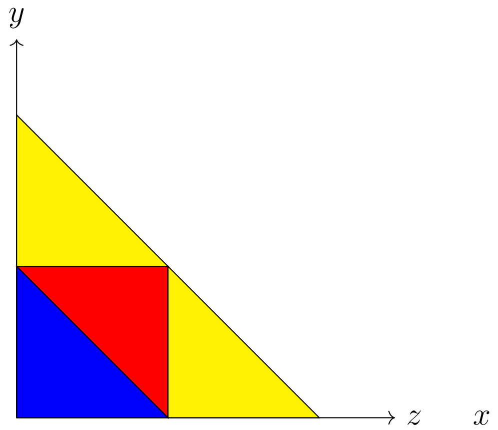


Figure 5.2: Optimal 8-tetrahedron packing in side view

Hence, an upper bound on $s_k^{\triangle}$ is given by two for $k = 1, \ldots, 8$. This reduces the number of possible integral affine images of $\triangle^3(1)$ to those that fit into the three-dimensional standard simplex with side length two.

The image of the three-dimensional standard simplex under an integral transformation is the interior of a tetrahedron with integer vertices and volume $\frac{1}{6}$. For a tetrahedron $T = \operatorname{int}(\operatorname{conv}(\{a, b, c, d\}))$ with $a, b, c, d \in \mathbb{Z}^3$ the volume is given by

$$\operatorname{vol}(T) = \frac{1}{6} |\det\left(\begin{bmatrix} a-d & b-d & c-d \end{bmatrix}\right)|.$$

To establish the shapelists, we consider all tetrahedra with integer vertices in the interval $[0, 2] \times [0, 2] \times [0, 2]$ that satisfy the volume condition. We then sort the coordinates, shift them to the origin and remove duplicates as in the two-dimensional case described in Chapter 3. This results in a list of 73 different shapes that are shown in Tables 5.1, 5.2, 5.3 and 5.4.

Shape	Image
$T_1 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_2 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\0\\1\end{pmatrix}\right\}\right)\right)$	
$T_3 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\1\end{pmatrix}, \begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_4 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\1\end{pmatrix}, \begin{pmatrix}1\\0\\1\end{pmatrix}\right\}\right)\right)$	
$T_5 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}1\\0\\-1\end{pmatrix}\right\}\right)\right)$	
$T_6 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_7 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\0\\-1\end{pmatrix}\right\}\right)\right)$	
$T_8 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)\right)$	
$T_9 = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\1\end{pmatrix}, \begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)\right)$	
$T_{10} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}1\\1\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{11} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\1\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{12} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\1\\1\end{pmatrix}, \begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_{13} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\1\\1\end{pmatrix}, \begin{pmatrix}1\\0\\1\end{pmatrix}\right\}\right)\right)$	
$T_{14} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\0\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{15} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_{16} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\-2\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}1\\0\\-2\end{pmatrix}\right\}\right)\right)$	
$T_{17} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\-2\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}1\\0\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{18} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\1\\1\end{pmatrix}, \begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)\right)$	
$T_{19} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\1\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{20} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\-2\end{pmatrix}, \begin{pmatrix}0\\1\\-1\end{pmatrix}, \begin{pmatrix}1\\1\\-2\end{pmatrix}\right\}\right)\right)$	

Table 5.1: Tetrahedron shapes 1 - 20

Shape	Image
$T_{21} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\-1\end{pmatrix},\begin{pmatrix}0\\2\\-1\end{pmatrix},\begin{pmatrix}1\\0\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{22} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\-1\end{pmatrix},\begin{pmatrix}0\\2\\-1\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_{23} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}0\\2\\-1\end{pmatrix},\begin{pmatrix}1\\0\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{24} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}0\\2\\-1\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_{25} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\-1\end{pmatrix},\begin{pmatrix}0\\2\\-1\end{pmatrix},\begin{pmatrix}1\\1\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{26} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}0\\2\\-1\end{pmatrix},\begin{pmatrix}1\\1\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{27} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\0\\1\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}1\\-1\\0\end{pmatrix}\right\}\right)\right)$	
$T_{28} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\0\\1\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}1\\-1\\1\end{pmatrix}\right\}\right)\right)$	
$T_{29} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\0\\1\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix},\begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)\right)$	
$T_{30} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\0\\1\end{pmatrix},\begin{pmatrix}1\\0\\1\end{pmatrix},\begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)\right)$	
$T_{31} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\0\\1\end{pmatrix},\begin{pmatrix}1\\0\\-1\end{pmatrix},\begin{pmatrix}1\\1\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{32} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\0\\1\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix},\begin{pmatrix}1\\1\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{33} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix},\begin{pmatrix}1\\0\\1\end{pmatrix}\right\}\right)\right)$	
$T_{34} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\1\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix},\begin{pmatrix}1\\0\\1\end{pmatrix}\right\}\right)\right)$	
$T_{35} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\-1\end{pmatrix},\begin{pmatrix}1\\0\\-1\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_{36} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}1\\0\\-1\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix}\right\}\right)\right)$	
$T_{37} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\-2\end{pmatrix},\begin{pmatrix}1\\0\\-2\end{pmatrix},\begin{pmatrix}1\\0\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{38} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\-1\end{pmatrix},\begin{pmatrix}1\\0\\-2\end{pmatrix},\begin{pmatrix}1\\0\\-1\end{pmatrix}\right\}\right)\right)$	
$T_{39} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}1\\0\\1\end{pmatrix},\begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)\right)$	
$T_{40} = \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\1\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix},\begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)\right)$	

Table 5.2: Tetrahedron shapes 21 - 40

Shape	Image
$T_{41} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{42} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{43} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\-2\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-2\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{44} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-2\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-2\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{45} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\2\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{46} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\2\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{47} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{48} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{49} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{50} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{51} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{52} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{53} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{54} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{55} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{56} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{57} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{58} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{59} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{60} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\\0\end{smallmatrix}\right)\right\}\right)\right)$	

Table 5.3: Tetrahedron shapes 41 - 60

Shape	Image
$T_{61} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}0\\0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{62} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{63} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{64} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-2\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-2\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{65} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{66} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{67} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-2\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-2\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{68} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{69} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\0\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{70} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{71} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\\-1\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{72} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\\0\end{smallmatrix}\right)\right\}\right)\right)$	
$T_{73} = \mathrm{int}\left(\mathrm{conv}\left(\left\{\left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\-1\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}1\\0\\-1\end{smallmatrix}\right), \left(\begin{smallmatrix}2\\-1\\-1\end{smallmatrix}\right)\right\}\right)\right)$	

Table 5.4: Tetrahedron shapes 61 - 73

Now we can determine $s_k^{\triangle}$ by computing an optimal packing for every k-cardinality multisubset of the shapelist. The number of k-cardinality multisubsets for $k = 1, \ldots, 8$ is shown in Table 5.5.

k	$\lvert\mathcal{S}_k^{\triangle}\rvert$	$\left(\!\binom{\lvert\mathcal{S}_k^{\triangle}\rvert}{k}\!\right)$
1	1	1
2	73	2 701
3	73	67 525
4	73	1 282 975
5	73	19 757 815
6	73	256 851 595
7	73	2 898 753 715
8	73	28 987 537 150

Table 5.5: Number of k-cardinality multisubsets of the shapelists for $k = 1, \ldots, 8$

The first column shows the value of k. The second column shows the cardinality of the corresponding shapelist. The third column shows the number of k-cardinality multisubsets of this shapelist. Due to the cardinality of the shapelist, the number of multisubsets is quite large even for small numbers of k. However, we were able to compute k-tetrahedron packings for $k = 1, \ldots, 8$ by employing the same branch-and-bound strategy as in the two-dimensional setting. The (incomplete) search tree for $k = 3$ is shown in Figure 5.3.

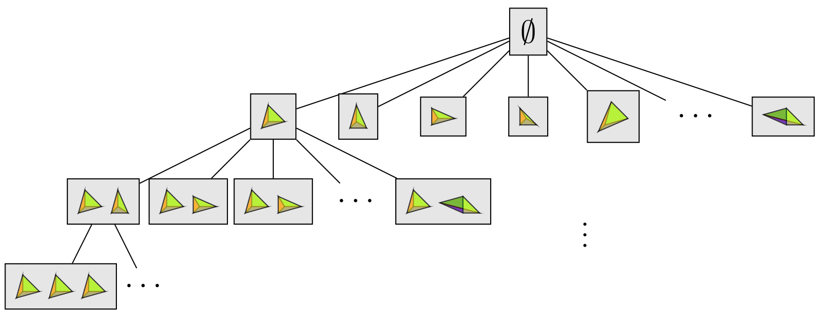


Figure 5.3: Branch-and-bound search tree for the 3-tetrahedron packing

We start the search with the global upper bound of two. Whenever the computation for a node produces an optimum packing that exceeds the global upper bound, we can fathom the subtree rooted at this node. Whenever the computation for a node at level k produces an optimum packing that improves the global upper bound, we update it and memorize the packing as the incumbent solution. At termination, the incumbent solution is an optimum packing with value of the upper bound. In the next section we will explain how to compute an optimal packing at each node. Again, we refer to this as the *inner optimization problem*.

5.2 Inner Optimization Problem

Problem 5.3 (Inner Optimization Problem)**.**
Given $T_1, \dots, T_m \in \mathcal{S}_k^{\triangle}$, determine the minimum side length s such that there exist vectors $t_1, \dots, t_m \in \mathbb{R}^3$ satisfying

$$
\begin{array}{lll}
T_i + t_i \subseteq \text{◺}^3(s) & \forall i \in [m] & \textit{(containment condition)}, \\
(T_i + t_i) \cap (T_j + t_j) = \emptyset & \forall 1 \le i < j \le m & \textit{(disjointness condition)}.
\end{array}
$$

We will formulate Problem 5.3 as a mixed integer linear program. For this, we have to model the two conditions. To model the containment condition $T_i + t_i \subseteq \text{◺}^3(s)$, we first replace the open sets by their closures as this does not alter the containment condition. Each tetrahedron from the shapelist is given in the form

$$
\overline{T_i} = \operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}a_{i1}\\b_{i1}\\c_{i1}\end{pmatrix}, \begin{pmatrix}a_{i2}\\b_{i2}\\c_{i2}\end{pmatrix}, \begin{pmatrix}a_{i3}\\b_{i3}\\c_{i3}\end{pmatrix}\right\}\right).
$$

For the packing target $\text{◺}^3(s)$ we use the polyhedral description

$$
\overline{\text{◺}^3(s)} = \left\{ \begin{pmatrix}x\\y\\z\end{pmatrix} \in \mathbb{R}^3 \;\middle|\; x \ge 0,\ y \ge 0,\ z \ge 0,\ x + y + z \le s \right\}.
$$

Let the translation vector be given by $t_i = \begin{pmatrix}x_i\\y_i\\z_i\end{pmatrix}$. Because of convexity, we only need to check that the four translated vertices are contained in $\text{◺}^3(s)$, which leads to the following sixteen inequalities for every $i \in [m]$:

$$
\begin{aligned}
&x_i \geq 0, && y_i \geq 0, && z_i \geq 0, && x_i + y_i + z_i \leq s,\\
&x_i + a_{i1} \geq 0, && y_i + b_{i1} \geq 0, && z_i + c_{i1} \geq 0, && (x_i + a_{i1}) + (y_i + b_{i1}) + (z_i + c_{i1}) \leq s,\\
&x_i + a_{i2} \geq 0, && y_i + b_{i2} \geq 0, && z_i + c_{i2} \geq 0, && (x_i + a_{i2}) + (y_i + b_{i2}) + (z_i + c_{i2}) \leq s,\\
&x_i + a_{i3} \geq 0, && y_i + b_{i3} \geq 0, && z_i + c_{i3} \geq 0, && (x_i + a_{i3}) + (y_i + b_{i3}) + (z_i + c_{i3}) \leq s.
\end{aligned}
$$

By putting the constants to the right hand sides and taking minima/maxima in every column, we can reduce these sixteen inequalities to four inequalities for every $i \in [m]$:

$$
\begin{aligned}
x_i &\geq \max\{0, -a_{i1}, -a_{i2}, -a_{i3}\} && =: K_i^1,\\
y_i &\geq \max\{0, -b_{i1}, -b_{i2}, -b_{i3}\} && =: K_i^2,\\
z_i &\geq \max\{0, -c_{i1}, -c_{i2}, -c_{i3}\} && =: K_i^3,\\
x_i + y_i + z_i - s &\leq \min\{0, -a_{i1} - b_{i1} - c_{i1}, -a_{i2} - b_{i2} - c_{i2}, -a_{i3} - b_{i3} - c_{i3}\} && =: K_i^4.
\end{aligned}
$$

Thus, we get a total of $4m$ inequalities that we call the *containment constraints*.

Next, we model the disjointness constraint. As derived in Chapter 3, we know that $(T_i + t_i) \cap (T_j + t_j) = \emptyset$ if and only if $t_j - t_i \notin T_i \ominus T_j$.

By applying Lemma 3.7 to

$$
\begin{aligned}
T_i &= \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}a_{i1}\\b_{i1}\\c_{i1}\end{pmatrix}, \begin{pmatrix}a_{i2}\\b_{i2}\\c_{i2}\end{pmatrix}, \begin{pmatrix}a_{i3}\\b_{i3}\\c_{i3}\end{pmatrix}\right\}\right)\right),\\
T_j &= \operatorname{int}\left(\operatorname{conv}\left(\left\{\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}a_{j1}\\b_{j1}\\c_{j1}\end{pmatrix}, \begin{pmatrix}a_{j2}\\b_{j2}\\c_{j2}\end{pmatrix}, \begin{pmatrix}a_{j3}\\b_{j3}\\c_{j3}\end{pmatrix}\right\}\right)\right),
\end{aligned}
$$

we get a description of $T_i \ominus T_j$ as

$$T_i \ominus T_j = \operatorname{int}\Bigg(\operatorname{conv}\Bigg(\Bigg\{ \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} a_{i1} \\ b_{i1} \\ c_{i1} \end{pmatrix}, \begin{pmatrix} a_{i2} \\ b_{i2} \\ c_{i2} \end{pmatrix}, \begin{pmatrix} a_{i3} \\ b_{i3} \\ c_{i3} \end{pmatrix},$$
$$\begin{pmatrix} -a_{j1} \\ -b_{j1} \\ -c_{j1} \end{pmatrix}, \begin{pmatrix} a_{i1}-a_{j1} \\ b_{i1}-b_{j1} \\ c_{i1}-c_{j1} \end{pmatrix}, \begin{pmatrix} a_{i2}-a_{j1} \\ b_{i2}-b_{j1} \\ c_{i2}-c_{j1} \end{pmatrix}, \begin{pmatrix} a_{i3}-a_{j1} \\ b_{i3}-b_{j1} \\ c_{i3}-c_{j1} \end{pmatrix},$$
$$\begin{pmatrix} -a_{j2} \\ -b_{j2} \\ -c_{j2} \end{pmatrix}, \begin{pmatrix} a_{i1}-a_{j2} \\ b_{i1}-b_{j2} \\ c_{i1}-c_{j2} \end{pmatrix}, \begin{pmatrix} a_{i2}-a_{j2} \\ b_{i2}-b_{j2} \\ c_{i2}-c_{j2} \end{pmatrix}, \begin{pmatrix} a_{i3}-a_{j2} \\ b_{i3}-b_{j2} \\ c_{i3}-c_{j2} \end{pmatrix},$$
$$\begin{pmatrix} -a_{j3} \\ -b_{j3} \\ -c_{j3} \end{pmatrix}, \begin{pmatrix} a_{i1}-a_{j3} \\ b_{i1}-b_{j3} \\ c_{i1}-c_{j3} \end{pmatrix}, \begin{pmatrix} a_{i2}-a_{j3} \\ b_{i2}-b_{j3} \\ c_{i2}-c_{j3} \end{pmatrix}, \begin{pmatrix} a_{i3}-a_{j3} \\ b_{i3}-b_{j3} \\ c_{i3}-c_{j3} \end{pmatrix} \Bigg\}\Bigg)\Bigg).$$

For each triplet of these sixteen possible vertices, we compute the hyperplane spanned by those and check whether it is a facet of the Minkowski difference $T_i \ominus T_j$. In this manner, for each pair of tetrahedra we obtain the following description

$$T_i \ominus T_j = \left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3 \;\middle|\; \alpha_f^{ij} x + \beta_f^{ij} y + \gamma_f^{ij} z < \delta_f^{ij} \quad \forall f \in [16] \right\}.$$

In our computations the number of halfspaces was at most sixteen. If the number was strictly smaller than sixteen, we inserted copies of the last halfspace to fill the gap to simplify the implementation.

We now proceed analogously to the two-dimensional setting. The difference vector $t_j - t_i$ is not contained in the Minkowski difference if and only if at least one of the sixteen inequalities is violated. To model this condition, we introduce a binary variable w_f^{ij} that is equal to one if inequality f is violated. To express this implication, we use a Big-M-formulation, where the parameter M has to be chosen sufficiently large:

$$\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) + \gamma_f^{ij}(z_j - z_i) \geq \delta_f^{ij} - M\left(1 - w_f^{ij}\right).$$

Let $\hat{s}$ be the current global upper bound on s. To find an appropriate value for M, we can make the following estimation on the left hand side of the Big-M-formulation:

$$
\begin{aligned}
&\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) + \gamma_f^{ij}(z_j - z_i) \\
\geq & -|\alpha_f^{ij}||x_j - x_i| - |\beta_f^{ij}||y_j - y_i| - |\gamma_f^{ij}||z_j - z_i| \\
\geq & -|\alpha_f^{ij}|\hat{s} - |\beta_f^{ij}|\hat{s} - |\gamma_f^{ij}|\hat{s} \\
= & -\left(|\alpha_f^{ij}| + |\beta_f^{ij}| + |\gamma_f^{ij}|\right)\hat{s}
\end{aligned}
$$

Thus, we can choose M as

$$
M = \left(|\alpha_f^{ij}| + |\beta_f^{ij}| + |\gamma_f^{ij}|\right)\hat{s} + \delta_f^{ij}.
$$

We want at least one of the six binary variables w_f^{ij} to take the value one Equivalently, we can say that their sum should be greater or equal to one. Altogether, the disjointness condition is equivalent to

$$
\begin{aligned}
&w_f^{ij} \in \{0, 1\}, \\
&w_1^{ij} + \cdots + w_{16}^{ij} \geq 1, \\
&\alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) + \gamma_f^{ij}(z_j - z_i) \geq \delta_f^{ij} - M\left(1 - w_f^{ij}\right)
\end{aligned}
$$

for all $f \in [16]$. For every pair of triangles T_i, T_j with $1 \leq i < j \leq m$, we have these seventeen inequalities, so in total there are $17\binom{m}{2}$ inequalities that we call the disjointness constraints.

Problem 5.4 (Inner Optimization Problem - Mixed Integer Linear Formulation)**.**

$$
\begin{aligned}
\min \quad & s \\
& x_i \geq K_i^1 && \forall i \in [m] \\
& y_i \geq K_i^2 && \forall i \in [m] \\
& z_i \geq K_i^3 && \forall i \in [m] \\
& x_i + y_i + z_i - s \leq K_i^4 && \forall i \in [m] \\
& w_1^{ij} + \cdots + w_{16}^{ij} \geq 1 && \forall 1 \leq i < j \leq m \\
& \alpha_f^{ij}(x_j - x_i) + \beta_f^{ij}(y_j - y_i) + \gamma_f^{ij}(z_j - z_i) \geq \\
& \delta_f^{ij} - M\left(1 - w_f^{ij}\right) && \forall 1 \leq i < j \leq m \\
& && \forall f \in [16] \\
& s \in \mathbb{R}, \quad x, y, z \in \mathbb{R}^m, \quad w \in \{0,1\}^{16\binom{m}{2}}
\end{aligned}
$$

Since the implementation of the outer optimization problem is equivalent to the implementation in the two-dimensional setting described in Section 3.3, we will skip the details and continue with the experimental results.

5.3 Experimental Results

As in the two-dimensional setting, the computational experiments were carried out under the Debian 10 operating system on two Intel Xeon E5-2690v2 CPUs with 3.00GHz and 10 cores each. For solving the inner optimization problem we call the GUROBI Optimization Software Version 9.0.3.

We found that $s_k^{\triangle} = 1$ for $k = 1$ and $s_k^{\triangle} = 2$ for $k = 2, \ldots, 8$. Figure 5.4 and Figure 5.5 show one exemplary optimal packing for $k = 1, \ldots, 8$.

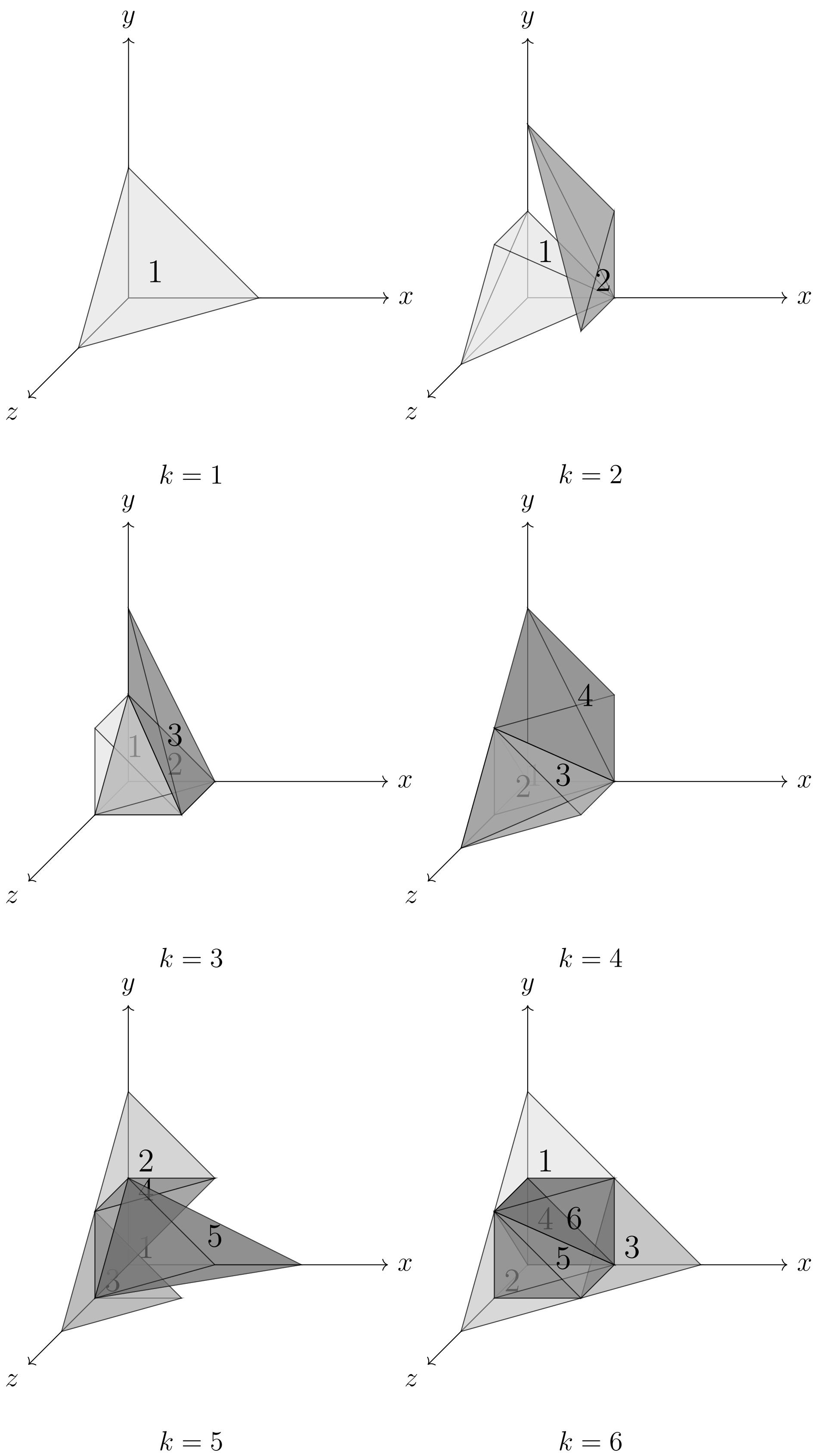
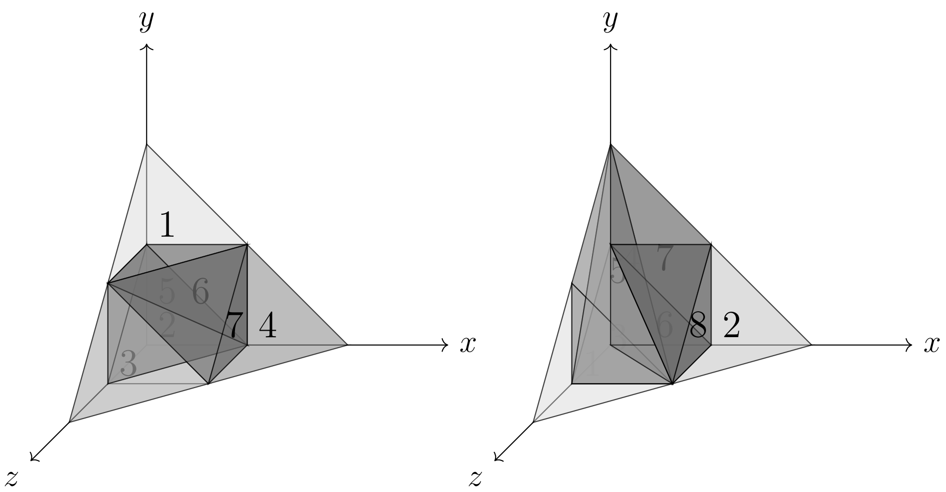


Figure 5.4: Optimal k-tetrahedron packings for $k = 1, \ldots, 6$



$k = 7$ $k = 8$

Figure 5.5: Optimal k-tetrahedron packings for $k = 7, 8$

As in the two-dimensional setting, our program detects all multisubsets that allow for an optimal packing for a given k. Table 5.6 shows the number of optimal multisubsets for $k = 1, \ldots, 8$.

k	#Optimal Multisubsets
1	1
2	1 123
3	4 871
4	9 914
5	11 709
6	8 075
7	2 899
8	408

Table 5.6: Number of multisubsets that allow for an optimal k-tetrahedron packing for $k = 1, \ldots, 8$

There are far more optimal multisubsets than in the two-dimensional setting due to the greater number of feasible multisubsets. Unlike the four-dimensional case, it is not yet known whether all these computed packings are equivalent.

Table 5.7 shows the timing statistics of our algorithm for $k = 1, \ldots, 8$. The column labels are

- k: the number of tetrahedra to be packed,
- $\left(\!\!\binom{|S_k^{\triangle}|}{k}\!\!\right)$: the number of k-cardinality multisubsets of the shapelist,
- #I-Calls: the number of calls to the inner optimization procedure,
- Avg I-Time: the average cpu time spent in an inner optimization procedure,
- Max I-Time: the maximum cpu time spent in an inner optimization procedure,
- Total Time: the total cpu time.

k	$\left(\!\!\binom{\lvert S_k^{\triangle}\rvert}{k}\!\!\right)$	#I-Calls	Avg I-Time	Max I-Time	Total Time
1	1	1	0 : 00 : 00.00	0 : 00 : 00.00	0 : 00 : 00.00
2	2 701	2 773	0 : 00 : 00.00	0 : 00 : 00.07	0 : 00 : 01.90
3	67 525	4 871	0 : 00 : 00.00	0 : 00 : 00.04	0 : 00 : 06.59
4	1 282 975	9 914	0 : 00 : 00.00	0 : 00 : 00.14	0 : 00 : 25.83
5	19 757 815	13 118	0 : 00 : 00.00	0 : 00 : 10.15	0 : 06 : 04.31
6	256 851 595	8 075	0 : 00 : 00.01	0 : 00 : 00.09	0 : 01 : 01.06
7	2 898 753 715	2 899	0 : 00 : 00.01	0 : 00 : 00.35	0 : 00 : 39.66
8	28 987 537 150	408	0 : 00 : 00.01	0 : 00 : 00.08	0 : 00 : 12.77

Table 5.7: Timing statistics for the k-tetrahedron packing given in the format "hh:mm:ss" for $k = 1, \dots, 8$

The inner optimization procedure is very fast on all instances. For computing k-tetrahedron packings for $k \geq 9$, the difficulty rather consists in the high number of feasible multisubsets. For $k = 9$ the cardinality of the shapelist increases to 854 and the number of multisubsets thereof increases to 694 392 240 786 929 755 070 $\approx 7 \times 10^{20}$. Therefore, computing $s_9^{\triangle}$ seems out of reach. Albeit, instead of working with the shapelist $S_9^{\triangle}$, we can consider the smaller shapelist $S_8^{\triangle}$ to compute upper bounds on $s_9^{\triangle}$. This work is in progress at the moment.

Appendix

In this appendix we provide the source code of the algorithms described in the main part of the thesis. The code is stored on a compact disc that contains seven different directories

- `solveinner/`,
- `solveinner3D/`,
- `packing/`,
- `packing3D/`,
- `sdp/`,
- `mccormick/`,
- `output/`.

The directory `solveinner/` contains the source code `solveinner_gurobi.c` for solving the inner optimization problem (Problem 3.8). There is also a version called `solveinner_cplex.c` that uses the CPLEX Optimizer instead of the GUROBI Optimizer for solving the mixed integer linear program. Besides, the directory contains the source codes `shapes.c` and `minkowski.c` that are used to establish the input files `shapelist.txt` and `minkowski.txt`, respectively. The structure of those two input files is explained in the corresponding establishing files. To solve the inner optimization problem for a specific multisubset, one gives the indices of the corresponding triangles to the executable file `solveinner_gurobi`. For example, to solve the inner optimization problem for the multisubset $\{T_1, T_1, T_3, T_7\}$ the command is "`solveinner_gurobi 1 1 3 7`".

The directory `packing/` contains the source code `packing_gurobi.c` for solving the outer optimization problem (Problem 3.2). The input files `shapelist.txt` and `minkowski.txt` are the same as in the directory `solveinner/`. The bounds file described in Chapter 3 is called `bounds-triangle.txt`. The first line of the bounds file has the two entries

`#bounds`, `k`,

which represent the number of bounds and the value of k that stem from the last run of the program. The subsequent lines have four entries

`fipri`, `nsimplices`, `lowerbound`, `optimal`

as described in Chapter 3. The executable file `initbounds` resets the bounds-file. To solve the outer optimization problem for $k \in \{1, \ldots, 13\}$, the command is "`packing_gurobi k`".

The directories `solveinner3D/` and `packing3D` are the three-dimensional counterparts of the previous two directories and are constructed analogously to the former.

The directory `sdp/` contains the semidefinite relaxations of the three different quadratically constrained quadratic programs described in Chapter 4. The program for the semidefinite relaxation of Problem 4.2 is called `sdp.c`. The programs for the semidefinite relaxation of Problem 4.4 and Problem 4.6 are called `sdp_hyperplane.c` and `sdp_convexcombination.c`, respectively. All three programs also require the two input files `shapelist.txt` and `minkowski.txt`.

The directory `mccormick/` contains the two McCormick relaxations of the mixed integer binary program (Problem 4.7). They are called `mccormick_1.c` and `mccormick_2.c`. Again, both programs require the two input files `shapelist.txt` and `minkowski.txt`.

The last directory `output/` contains the log files corresponding to the experimental results presented in this thesis.

Bibliography

[Dar82] Gaston Darboux. "Sur le problème de Pfaff". fr. In: *Bulletin des Sciences Mathématiques et Astronomiques* 2e série, 6.1 (1882), pp. 49–68. URL: http://www.numdam.org/item/BSMA_1882_2_6_1_49_1.

[Far02] Julius Farkas. "Theorie der einfachen Ungleichungen." In: *Journal für die reine und angewandte Mathematik* 124 (1902), pp. 1–27. URL: http://eudml.org/doc/149129.

[Neu47] John von Neumann. "Discussion of a maximum problem". In: *Manuscript* (1947).

[Dan51] George B. Dantzig. *Maximization of a Linear Function of Variables Subject to Linear Inequalities*. Ed. by Tjalling C. Koopmans. New York: Wiley, 1951, pp. 339–347.

[DFJ54] George B. Dantzig, Delbert R. Fulkerson, and Selmer M. Johnson. "Solution of a large-scale traveling-salesman problem". In: *Operations Research* 2 (1954), pp. 393–410.

[Gom58] Ralph E. Gomory. "Outline of an algorithm for integer solutions to linear programs". In: *Bull. Amer. Math. Soc.* 64.5 (Sept. 1958), pp. 275–278. URL: https://projecteuclid.org:443/euclid.bams/1183522679.

[Sla59] Morton Slater. "Lagrange Multipliers Revisited". In: 80 (1959). URL: https://EconPapers.repec.org/RePEc:cwl:cwldpp:80.

[LD60] Alisa H. Land and Alison G. Doig. "An Automatic Method of Solving Discrete Programming Problems". In: *Econometrica* 28.3 (1960), pp. 497–520. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1910129.

[Sah74] Sartaj Sahni. "Computationally Related Problems". In: *SIAM Journal on Computing* 3.4 (1974), pp. 262–279. DOI: 10.1137/0203021.

[McC76] Garth P. McCormick. "Computability of Global Solutions to Factorable Nonconvex Programs: Part I – Convex Underestimating Problems". In: *Math. Program.* 10.1 (Dec. 1976), pp. 147–175. ISSN: 0025-5610. DOI: 10.1007/BF01580665.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co., 1979. ISBN: 0716710447.

[KTK80] M.K. Kozlov, S.P. Tarasov, and L.G. Khachiyan. "The polynomial solvability of convex quadratic programming". In: *USSR Computational Mathematics and Mathematical Physics* 20.5 (1980), pp. 223–228. ISSN: 0041-5553. DOI: https://doi.org/10.1016/0041-5553(80)90098-1.

[Gro85] Michail Gromov. "Pseudo holomorphic curves in symplectic manifolds." In: *Inventiones mathematicae* 82 (1985), pp. 307–348. URL: http://eudml.org/doc/143289.

[Sik89] Jean-Claude Sikorav. "Rigidité symplectique dans le cotangent de T^n". In: *Duke Math. J.* 59.3 (Dec. 1989), pp. 759–763. DOI: 10.1215/S0012-7094-89-05935-8.

[PV90] Panos M. Pardalos and Stephen A. Vavasis. "Quadratic Programming with One Negative Eigenvalue is NP-hard". In: *Journal of Global Optimization* 1 (1990), pp. 15–22.

[LS91] Francois Lalonde and Jean-Claude Sikorav. "Sous-variétés lagrangiennes et lagrangiennes exactes des fibrés cotangents." fre. In: *Commentarii mathematici Helvetici* 66.1 (1991), pp. 18–33. URL: http://eudml.org/doc/140221.

[McD91] Dusa McDuff. "Blow ups and symplectic embeddings in dimension 4". In: *Topology* 30.3 (1991), pp. 409–421. ISSN: 0040-9383. DOI: https://doi.org/10.1016/0040-9383(91)90021-U.

[MP94] Dusa McDuff and Leonid Polterovich. "Symplectic packings and algebraic geometry". In: *Inventiones mathematicae* 115.3 (1994), pp. 405–430. URL: http://eudml.org/doc/144175.

[HRW95] Christoph Helmberg, Franz Rendl, and Robert Weismantel. *Quadratic Knapsack Relaxations Using Cutting Planes and Semidefinite Programming*. eng. Tech. rep. SC-95-37. Takustr. 7, 14195 Berlin: ZIB, 1995.

[Bir96] Paul Biran. "Symplectic Packing in Dimension 4". In: *Geometric & Functional Analysis GAFA* 7 (1996), pp. 420–437.

[VB96] Lieven Vandenberghe and Stephen Boyd. *Semidefinite Programming*. Vol. 38. 1. 1996, pp. 49–95. DOI: 10.1137/1038003.

[Sch98] Alexander Schrijver. *Theory of Linear and Integer Programming*. New York: Wiley, 1998. ISBN: 978-0-471-98232-6.

[MMT00] F. Miller Maley, Jean Beth Mastrangeli, and Lisa Traynor. "Symplectic packings in cotangent bundles of tori". In: *Experimental Mathematics* 9.3 (2000), pp. 435–455. URL: https://projecteuclid.org:443/euclid.em/1045604678.

[Sil04] Ana Cannas da Silva. *Lectures on Symplectic Geometry*. Berlin, Heidelberg: Springer, 2004. ISBN: 978-3-540-45330-7.

[Van+05] Lieven Vandenberghe et al. *Interior-Point Algorithms for Semidefinite Programming Problems Derived from the KYP Lemma*. Vol. 312. May 2005, pp. 579–579. ISBN: 978-3-540-23948-2. DOI: 10.1007/10997703_12.

[RRW10] Franz Rendl, Giovanni Rinaldi, and Angelika Wiegele. "Solving Max-Cut to Optimality by Intersecting Semidefinite and Polyhedral Relaxations". In: *Math. Programming* 121.2 (2010), pp. 307–335.

[HZ12] Helmut Hofer and Eduard Zehnder. *Symplectic Invariants and Hamiltonian Dynamics* -. Basel: Birkhäuser, 2012. ISBN: 978-3-034-88540-9.

[MS17] Dusa McDuff and Dietmar Salamon. *Introduction to Symplectic Topology*. New York: Oxford University Press, 2017. ISBN: 978-0-198-79489-9.

Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Dissertation selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel und Literatur angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Werken dem Wortlaut oder dem Sinn nach entnommen wurden, sind als solche kenntlich gemacht. Ich versichere an Eides statt, dass diese Dissertation noch keiner anderen Fakultät oder Universität zur Prüfung vorgelegen hat; dass sie - abgesehen von unten angegebenen Teilpublikationen und eingebundenen Artikeln und Manuskripten - noch nicht veröffentlicht worden ist sowie, dass ich eine Veröffentlichung der Dissertation vor Abschluss der Promotion nicht ohne Genehmigung des Promotionsausschusses vornehmen werde. Die Bestimmungen dieser Ordnung sind mir bekannt. Darüber hinaus erkläre ich hiermit, dass ich die Ordnung zur Sicherung guter wissenschaftlicher Praxis und zum Umgang mit wissenschaftlichem Fehlverhalten der Universität zu Köln gelesen und sie bei der Durchführung der Dissertation zugrundeliegenden Arbeiten und der schriftlich verfassten Dissertation beachtet habe und verpflichte mich hiermit, die dort genannten Vorgaben bei allen wissenschaftlichen Tätigkeiten zu beachten und umzusetzen. Ich versichere, dass die eingereichte elektronische Fassung der eingereichten Druckfassung vollständig entspricht.

Kreuzau, den 15.02.2021