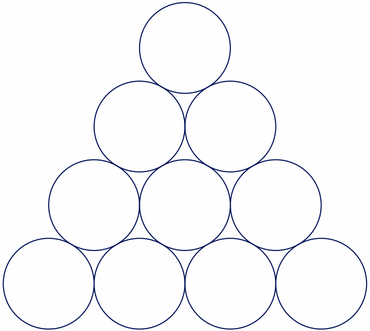
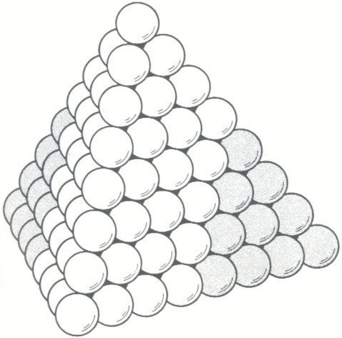
The lattice packing problem in dimension 9 by Voronoi's algorithm

Mathieu Dutour Sikirić & **Wessel van Woerden** (PQShield).

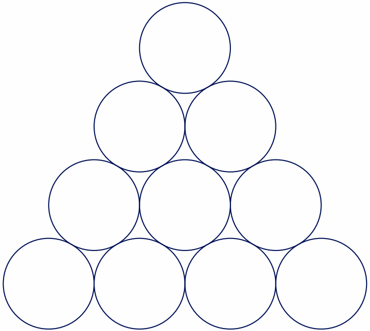
Sphere Packing Problem

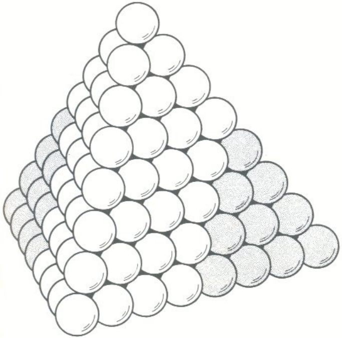
Sphere Packing Problem





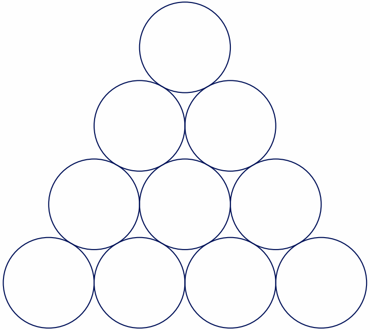
Sphere Packing Problem

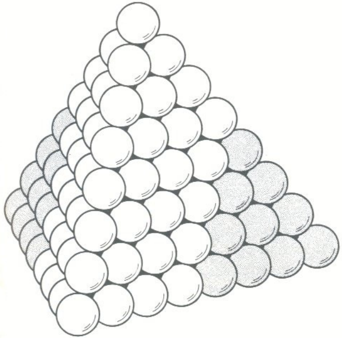




- Only solved in dimensions **2, 3, 8** and **24**...

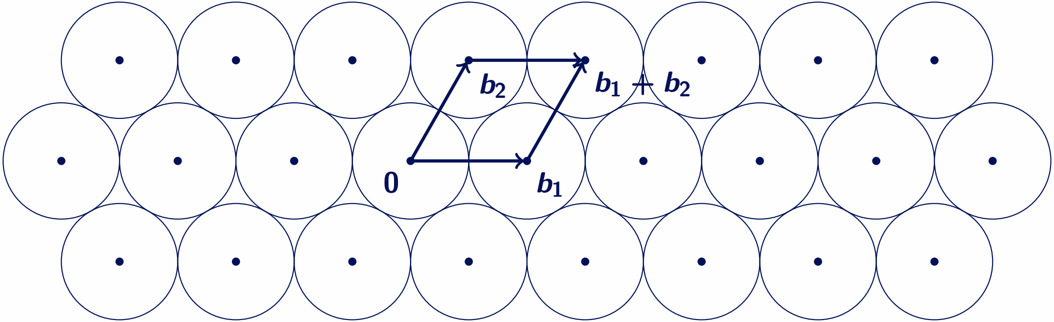
Sphere Packing Problem



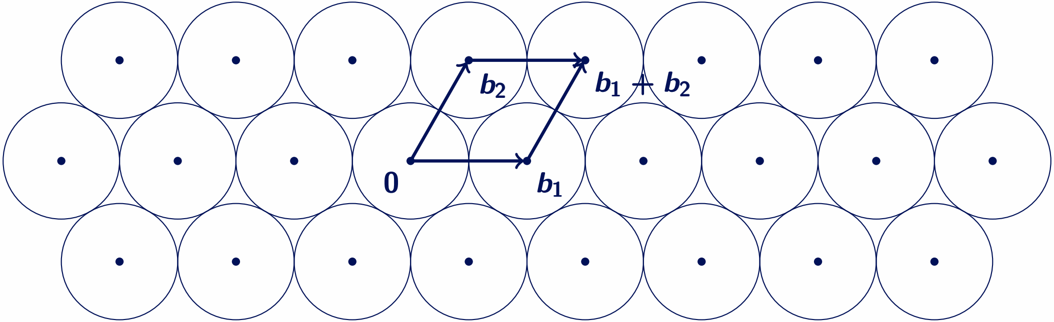


- Only solved in dimensions **2, 3, 8** and **24**...
- Dimension **3** only in **1998** by a computational proof (Thomas Hales)

Lattice Packing Problem

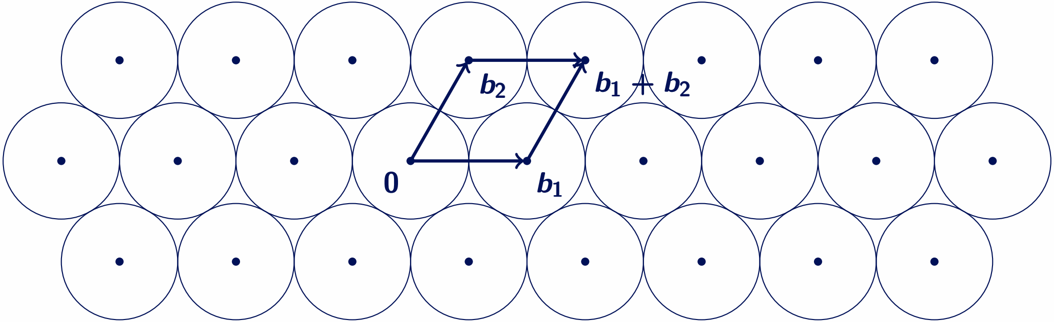


Lattice Packing Problem



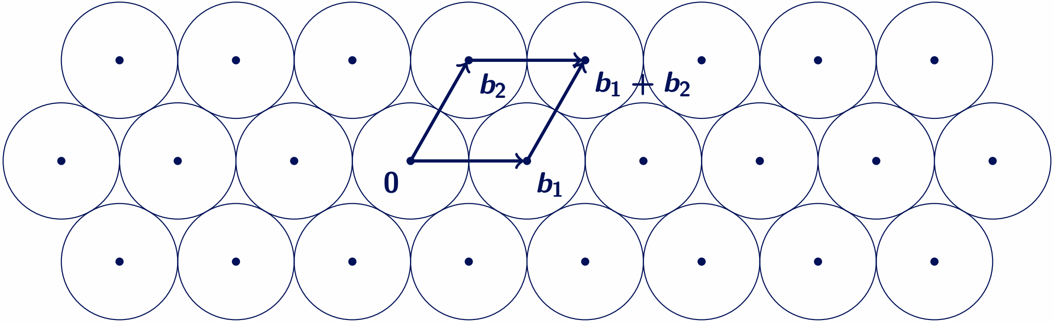
- Solved in dimensions $1, 2, \ldots, 8$ and 24.

Lattice Packing Problem



- Solved in $\underbrace{\text{dimensions } \mathbf{1, 2, \ldots, 8}}_{\geq \mathbf{90} \text{ years ago}}$ and $\mathbf{24}$.

Lattice Packing Problem



- Solved in $\underbrace{\text{dimensions } \mathbf{1, 2, \ldots, 8}}_{\geq \mathbf{90} \text{ years ago}}$ and $\mathbf{24}$.
- What about dimension $\mathbf{9}$?

- **dim $\leq$ 8**: theoretical proofs based on (H)KZ reduction.

The Lattice Packing Problem in dimension 9

- **dim $\leq$ 8**: theoretical proofs based on (H)KZ reduction.
- **Idea:** reduction theory gives an upper bound that is attained

The Lattice Packing Problem in dimension 9

- **dim $\leq$ 8**: theoretical proofs based on (H)KZ reduction.
- **Idea:** reduction theory gives an upper bound that is attained
- **Problem dim. 9:** conjectured best packing $\mathbf{\Lambda_9}$ is **not that good** (relatively)

The Lattice Packing Problem in dimension 9

- **dim $\leq$ 8**: theoretical proofs based on (H)KZ reduction.
- **Idea:** reduction theory gives an upper bound that is attained
- **Problem dim. 9:** conjectured best packing $\mathbf{\Lambda_9}$ is **not that good** (relatively)
- Best theoretical bounds are far off: current techniques do not seem sufficient.

The Lattice Packing Problem in dimension 9

- **dim $\leq$ 8**: theoretical proofs based on (H)KZ reduction.
- **Idea:** reduction theory gives an upper bound that is attained
- **Problem dim. 9:** conjectured best packing $\boldsymbol{\Lambda_9}$ is **not that good** (relatively)
- Best theoretical bounds are far off: current techniques do not seem sufficient.

What about a computational approach?

The Lattice Packing Problem in dimension 9

- **dim $\leq$ 8**: theoretical proofs based on (H)KZ reduction.
- **Idea:** reduction theory gives an upper bound that is attained
- **Problem dim. 9:** conjectured best packing $\mathbf{\Lambda_9}$ is **not that good** (relatively)
- Best theoretical bounds are far off: current techniques do not seem sufficient.

What about a computational approach?

Theorem: Voronoi's algorithm (**1908**)

For any fixed dimension $d \geq 1$, there exists an algorithm that runs in finite time and determines the best lattice packing.

The Lattice Packing Problem in dimension 9

- **dim $\leq$ 8**: theoretical proofs based on (H)KZ reduction.
- **Idea:** reduction theory gives an upper bound that is attained
- **Problem dim. 9:** conjectured best packing $\mathbf{\Lambda_9}$ is **not that good** (relatively)
- Best theoretical bounds are far off: current techniques do not seem sufficient.

What about a computational approach?

Theorem: Voronoi's algorithm (**1908**)

For any fixed dimension $d \geq 1$, there exists an algorithm that runs in finite time and determines the best lattice packing.

- **This work**: successfully completing Voronoi's algorithm in dimension **9**.

The Lattice Packing Problem in dimension 9

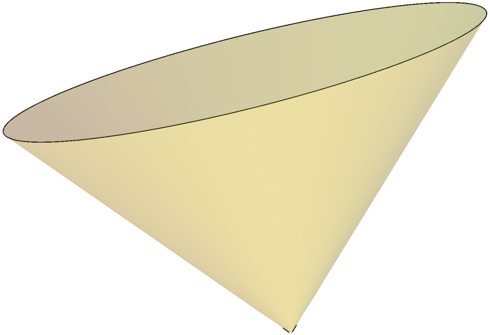
- **dim $\leq$ 8**: theoretical proofs based on (H)KZ reduction.
- **Idea:** reduction theory gives an upper bound that is attained
- **Problem dim. 9:** conjectured best packing $\mathbf{\Lambda_9}$ is **not that good** (relatively)
- Best theoretical bounds are far off: current techniques do not seem sufficient.

What about a computational approach?

Theorem: Voronoi's algorithm (**1908**)

For any fixed dimension $\boldsymbol{d} \geq \mathbf{1}$, there exists an algorithm that runs in finite time and determines the best lattice packing.

- **This work**: successfully completing Voronoi's algorithm in dimension **9**.
- **Corollary:** the laminated lattice $\mathbf{\Lambda_9}$ is the unique densest lattice packing.



- Cone of positive definite matrices

$$\mathcal{S}^d_{<0} \subset \mathcal{S}^d \subset \mathbb{R}^{d\times d}.$$

$$\dim(\mathcal{S}^d) = \tfrac{1}{2}d(d+1) =: n$$

- inner product: (to show these pictures)

$$\langle A, B\rangle := \mathrm{Tr}(A^t B) = \sum_{i,j} A_{ij}B_{ij}$$

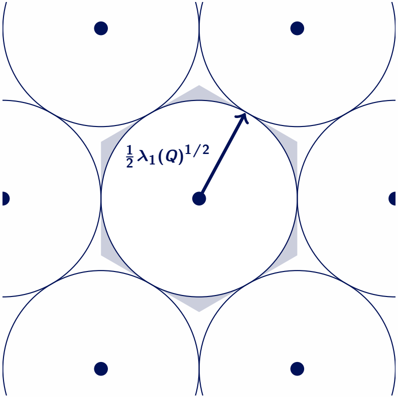
- $Q \in \mathcal{S}^d$ defines a quadratic form by

$$Q[x] := x^t Q x = \langle Q, xx^t\rangle \quad \forall x \in \mathbb{R}^d$$

Hermite Constant

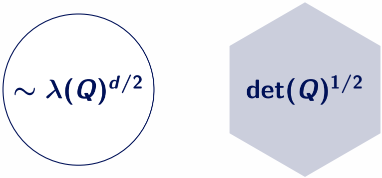
- Lattice $\boldsymbol{L} = \boldsymbol{B}\mathbb{Z}^{\boldsymbol{d}} \implies$ PQF $\boldsymbol{Q} = \boldsymbol{B}^{\boldsymbol{t}}\boldsymbol{B} \in \boldsymbol{S}^{\boldsymbol{d}}_{>\mathbf{0}}$.

- Lattice $L = B\mathbb{Z}^d \implies$ PQF $Q = B^t B \in S^d_{>0}$.



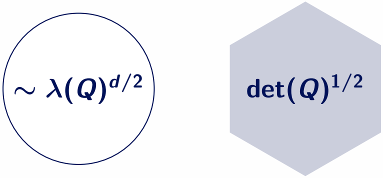
$$\lambda(Q) := \min_{x \in \mathbb{Z}^d \setminus \{0\}} Q[x] = \min_{y \in L \setminus \{0\}} \|y\|^2$$

$$\operatorname{Min} Q := \{x \in \mathbb{Z}^d : Q[x] = \lambda(Q)\}$$



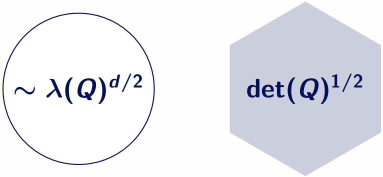
Hermite Constant

- Lattice $\boldsymbol{L} = \boldsymbol{B}\mathbb{Z}^d \implies$ PQF $\boldsymbol{Q} = \boldsymbol{B}^t\boldsymbol{B} \in \boldsymbol{S}^d_{>0}$.



Hermite Constant

- Lattice $\boldsymbol{L} = \boldsymbol{B}\mathbb{Z}^{\boldsymbol{d}} \implies$ PQF $\boldsymbol{Q} = \boldsymbol{B}^t\boldsymbol{B} \in \boldsymbol{S}^{\boldsymbol{d}}_{>0}$.

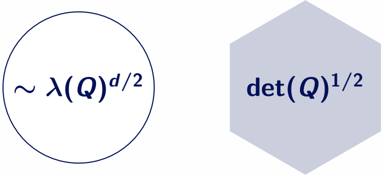


- Hermite invariant:

$$\gamma(\boldsymbol{Q}) = \frac{\lambda(\boldsymbol{Q})}{(\det \boldsymbol{Q})^{1/d}} \sim \text{density}(\boldsymbol{L})^{2/d}$$

Hermite Constant

- Lattice $L = B\mathbb{Z}^d \implies$ PQF $Q = B^t B \in S^d_{>0}$.



- Hermite invariant:

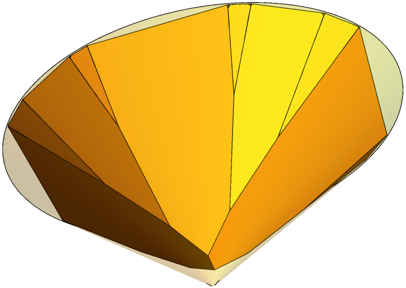
$$\gamma(Q) = \frac{\lambda(Q)}{(\det Q)^{1/d}} \sim \text{density}(L)^{2/d}$$

- Lattice packing problem $\Leftrightarrow$ determine Hermite's constant:

$$\gamma_d := \sup_{Q \in \mathcal{S}^d_{>0}} \gamma(Q)$$

- For $\lambda > 0$ we define the Ryshkov Polyhedra

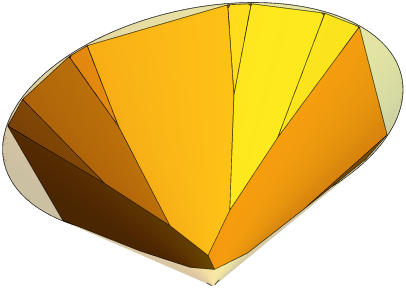
$$\mathcal{P}_\lambda = \{Q \in \mathcal{S}^d_{>0} : \lambda(Q) \geq \lambda\}$$



Ryshkov Polyhedra

- For $\lambda > 0$ we define the Ryshkov Polyhedra

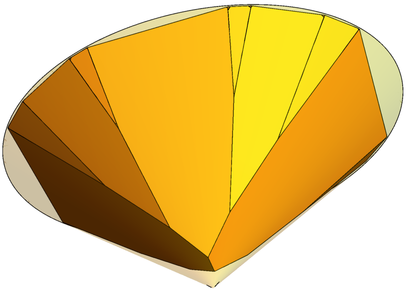
$$\mathcal{P}_\lambda = \bigcap_{x \in \mathbb{Z}^d \setminus \{0\}} \{Q \in \mathcal{S}^d : \langle Q, xx^t \rangle \geq \lambda\} \subset \mathcal{S}^d_{>0}$$



Ryshkov Polyhedra

- For $\boldsymbol{\lambda} > \mathbf{0}$ we define the Ryshkov Polyhedra

$$\mathcal{P}_\lambda = \bigcap_{x \in \mathbb{Z}^d \setminus \{0\}} \{Q \in \mathcal{S}^d : \langle Q, xx^t \rangle \geq \lambda\} \subset \mathcal{S}^d_{>0}$$



- Each facet corresponds to some primitive $\pm x \in \mathbb{Z}^d$.
- Locally finite

Ryshkov Polyhedra

- For $\lambda > 0$ we define the Ryshkov Polyhedra

$$\mathcal{P}_\lambda = \{Q \in \mathcal{S}^d_{>0} : \lambda(Q) \geq \lambda\}$$

- We have

$$\gamma_d = \frac{\lambda}{\inf_{Q \in \mathcal{P}_\lambda} \det(Q)^{1/d}}$$

Ryshkov Polyhedra

- For $\lambda > 0$ we define the Ryshkov Polyhedra

$$\mathcal{P}_\lambda = \{Q \in \mathcal{S}^d_{>0} : \lambda(Q) \geq \lambda\}$$

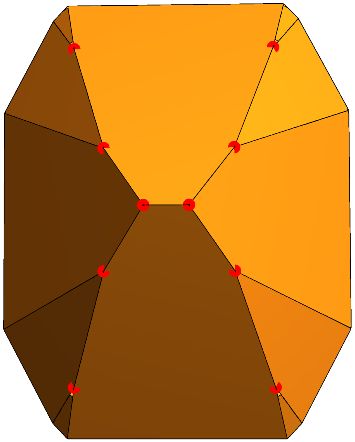
- We have

$$\gamma_d = \frac{\lambda}{\inf_{Q \in \mathcal{P}_\lambda} \det(Q)^{1/d}}$$

- Minkowski: $\det(Q)^{1/d}$ is (strictly) concave on $\mathcal{S}^d_{>0}$

  $\Longrightarrow$ Local optima at vertices of $\mathcal{P}_\lambda$. (uses that $\mathcal{P}_\lambda$ is locally finite)

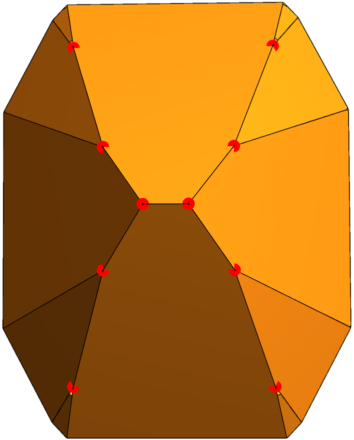
Perfect forms



- $\boldsymbol{Q}$ is perfect $\Leftrightarrow$ $\boldsymbol{Q}$ is a vertex of $\mathcal{P}_{\lambda(\boldsymbol{Q})}$.

  $\Leftrightarrow$ $\boldsymbol{Q}$ is fully determined by Min $\boldsymbol{Q}$ and $\boldsymbol{\lambda_1(Q)}$.

Perfect forms

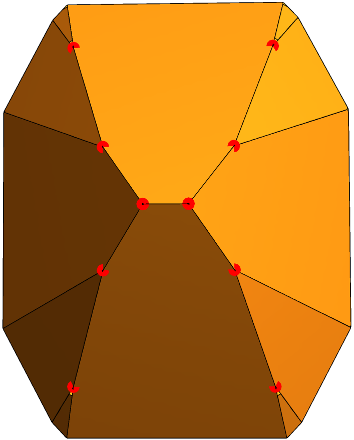


- Q is perfect $\Leftrightarrow$ Q is a vertex of $\mathcal{P}_{\lambda(Q)}$.

  $\Leftrightarrow$ Q is fully determined by Min Q and $\lambda_1(Q)$.
- Facets adjacent to $Q \leftrightarrow \pm x \in \text{Min } Q$

  $\Longrightarrow \; |\text{Min } Q| \geq 2n = d(d+1)$

Perfect forms



- $\boldsymbol{Q}$ is perfect $\Leftrightarrow$ $\boldsymbol{Q}$ is a vertex of $\mathcal{P}_{\lambda(\boldsymbol{Q})}$.

  $\Leftrightarrow$ $\boldsymbol{Q}$ is fully determined by Min $\boldsymbol{Q}$ and $\boldsymbol{\lambda_1(Q)}$.
- Facets adjacent to $\boldsymbol{Q} \leftrightarrow \pm\boldsymbol{x} \in$ Min $\boldsymbol{Q}$

  $\implies$ $|\text{Min } \boldsymbol{Q}| \geq \boldsymbol{2n = d(d+1)}$
- Voronoi's algorithm: enumerate all perfect forms (up to equivalence/similarity)

Equivalence and similarity

- $\boldsymbol{B}$ and $\boldsymbol{BU}$ generate the same lattice for $\boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$.

Equivalence and similarity

- $\boldsymbol{B}$ and $\boldsymbol{BU}$ generate the same lattice for $\boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$.
- $\boldsymbol{B}$ and $\boldsymbol{OBU}$ generate **isomorphic** lattices for $\boldsymbol{O} \in \mathbb{O}_{\boldsymbol{d}}(\mathbb{R})$. (same density)

Equivalence and similarity

- $\boldsymbol{B}$ and $\boldsymbol{BU}$ generate the same lattice for $\boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$.
- $\boldsymbol{B}$ and $\boldsymbol{OBU}$ generate **isomorphic** lattices for $\boldsymbol{O} \in \mathbb{O}_{\boldsymbol{d}}(\mathbb{R})$. (same density)
- **Arithmetically equivalence:** $\exists \boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$ s.t. $\boldsymbol{Q}' = \boldsymbol{U}^t \boldsymbol{Q} \boldsymbol{U}$.

Equivalence and similarity

- $\boldsymbol{B}$ and $\boldsymbol{BU}$ generate the same lattice for $\boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$.
- $\boldsymbol{B}$ and $\boldsymbol{OBU}$ generate **isomorphic** lattices for $\boldsymbol{O} \in \mathbb{O}_{\boldsymbol{d}}(\mathbb{R})$. (same density)
- **Arithmetically equivalence:** $\exists \boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$ s.t. $\boldsymbol{Q}' = \boldsymbol{U}^t \boldsymbol{Q} \boldsymbol{U}$.
- **Similarity:** Arithmetical equivalence up to positive scaling.

Equivalence and similarity

- $\boldsymbol{B}$ and $\boldsymbol{BU}$ generate the same lattice for $\boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$.
- $\boldsymbol{B}$ and $\boldsymbol{OBU}$ generate **isomorphic** lattices for $\boldsymbol{O} \in \mathbb{O}_{\boldsymbol{d}}(\mathbb{R})$. (same density)
- **Arithmetically equivalence:** $\exists \boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$ s.t. $\boldsymbol{Q}' = \boldsymbol{U}^t \boldsymbol{Q} \boldsymbol{U}$.
- **Similarity:** Arithmetical equivalence up to positive scaling.

Theorem: Voronoi (**1908**)

Up to similarity there are only a finite number of perfect forms in each dimension

Equivalence and similarity

- B and BU generate the same lattice for $U \in \mathrm{GL}_d(\mathbb{Z})$.
- B and OBU generate **isomorphic** lattices for $O \in \mathbb{O}_d(\mathbb{R})$. (same density)
- **Arithmetically equivalence:** $\exists U \in \mathrm{GL}_d(\mathbb{Z})$ s.t. $Q' = U^t Q U$.
- **Similarity:** Arithmetical equivalence up to positive scaling.

Theorem: Voronoi (1908)

Up to similarity there are only a finite number of perfect forms in each dimension

- Automorphism group $\mathrm{Aut}(Q) = \{U \in \mathrm{GL}_d(\mathbb{Z}) : U^t Q U = Q\}$.

Equivalence and similarity

- B and BU generate the same lattice for $U \in \mathrm{GL}_d(\mathbb{Z})$.
- B and OBU generate **isomorphic** lattices for $O \in \mathbb{O}_d(\mathbb{R})$. (same density)
- **Arithmetically equivalence:** $\exists U \in \mathrm{GL}_d(\mathbb{Z})$ s.t. $Q' = U^t Q U$.
- **Similarity:** Arithmetical equivalence up to positive scaling.

Theorem: Voronoi (1908)

Up to similarity there are only a finite number of perfect forms in each dimension

- Automorphism group $\mathrm{Aut}(Q) = \{U \in \mathrm{GL}_d(\mathbb{Z}) : U^t Q U = Q\}$.
- We have Min $U^t Q U = U^{-1} \cdot$ Min Q. ($\mathrm{GL}_d(\mathbb{Z})$ acts on $\mathcal{P}_\lambda$)

Number of perfect forms

- p_d := number of non-similar d-dimensional perfect forms.

Number of perfect forms

- $p_d :=$ number of non-similar d-dimensional perfect forms.

In theory..

$$p_d < e^{O(d^4 \log(d))} \quad \text{(C. Soulé, 1998)}$$

$$e^{\Omega(d)} < p_d < e^{O(d^3 \log(d))} \quad \text{(R. Bacher, 2017)}$$

Number of perfect forms

- $p_d :=$ number of non-similar d-dimensional perfect forms.

In theory..

$$p_d < e^{O(d^4 \log(d))} \qquad \text{(C. Soulé, 1998)}$$

$$e^{\Omega(d)} < p_d < e^{O(d^3 \log(d))} \qquad \text{(R. Bacher, 2017)}$$

Theorem: vW, 2020

$$p_d < e^{O(d^2 \log(d))}$$

Number of perfect forms

- $p_d :=$ number of non-similar d-dimensional perfect forms.

In theory..

$$p_d < e^{O(d^4 \log(d))} \qquad \text{(C. Soulé, 1998)}$$

$$e^{\Omega(d)} < p_d < e^{O(d^3 \log(d))} \qquad \text{(R. Bacher, 2017)}$$

Theorem: vW, 2020

$$p_d < e^{O(d^2 \log(d))}$$

In practice..

d	# p_d
2	1 (Lagrange, 1773)
3	1 (Gauss, 1840)
4	2 (Korkine & Zolotarev, 1877)
5	3 (Korkine & Zolotarev, 1877)
6	7 (Barnes, 1957)
7	33 (Jaquet, 1993)
8	10916 (DSV, 2005)
9	$\geq$ 500.000 (DSV, 2005)
	$\geq$ 23.000.000 (**vW**, 2018)

Number of perfect forms

- $p_d :=$ number of non-similar d-dimensional perfect forms.

In theory..

$$p_d < e^{O(d^4 \log(d))} \qquad \text{(C. Soulé, } \mathbf{1998}\text{)}$$

$$e^{\Omega(d)} < p_d < e^{O(d^3 \log(d))} \qquad \text{(R. Bacher, } \mathbf{2017}\text{)}$$

Theorem: vW, 2020

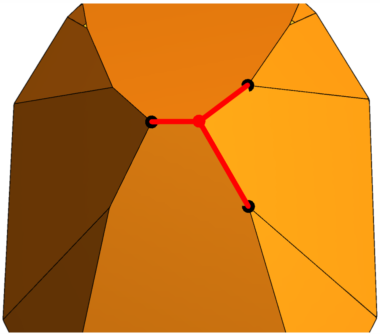
$$p_d < e^{O(d^2 \log(d))}$$

In practice..

d	# p_d
2	1 (Lagrange, 1773)
3	1 (Gauss, 1840)
4	2 (Korkine & Zolotarev, 1877)
5	3 (Korkine & Zolotarev, 1877)
6	7 (Barnes, 1957)
7	33 (Jaquet, 1993)
8	10916 (DSV, 2005)
9	$\geq$ 500.000 (DSV, 2005)
	$\geq$ 23.000.000 (**vW**, 2018)
	Many more, to be continued...

Voronoi's Algorithm
Challenges & Solutions

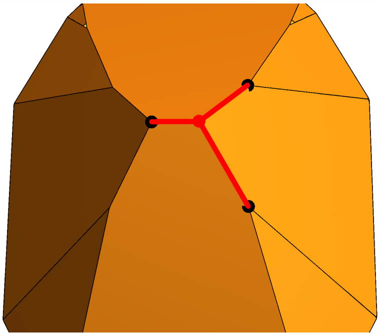
- Voronoi's Algorithm finds all d-dimensional perfect forms.



Voronoi's Algorithm

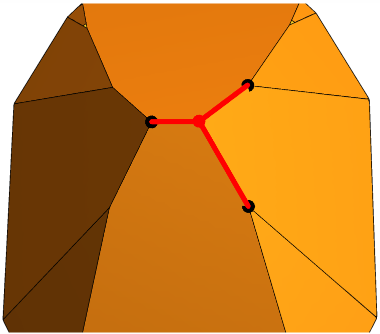
- Voronoi's Algorithm finds all $\boldsymbol{d}$-dimensional perfect forms.

1. Start at a single vertex of $\mathcal{P}_1$.



- Voronoi's Algorithm finds all $\boldsymbol{d}$-dimensional perfect forms.

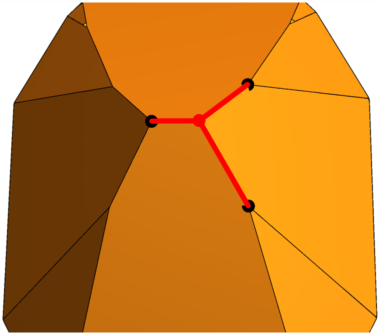
1. Start at a single vertex of $\mathcal{P}_1$.
2. Determine all neighbouring perfect forms.



Voronoi's Algorithm

- Voronoi's Algorithm finds all $\boldsymbol{d}$-dimensional perfect forms.

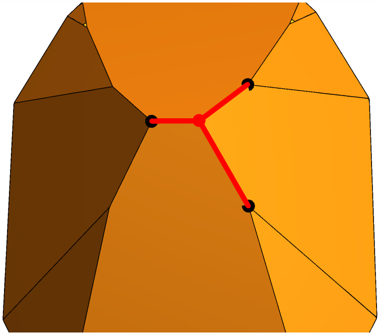
1. Start at a single vertex of $\mathcal{P}_{\mathbf{1}}$.
2. Determine all neighbouring perfect forms.
3. Keep those that are new.



Voronoi's Algorithm

- Voronoi's Algorithm finds all $\boldsymbol{d}$-dimensional perfect forms.

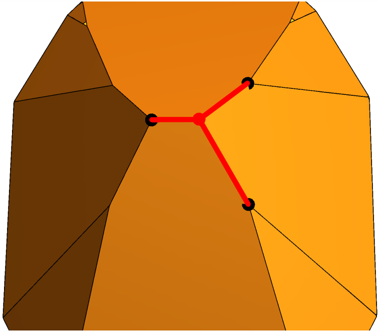
1. Start at a single vertex of $\mathcal{P}_{\mathbf{1}}$.
2. Determine all neighbouring perfect forms.
3. Keep those that are new.
4. Repeat for each perfect form.



Voronoi's Algorithm

- Voronoi's Algorithm finds all $\boldsymbol{d}$-dimensional perfect forms.

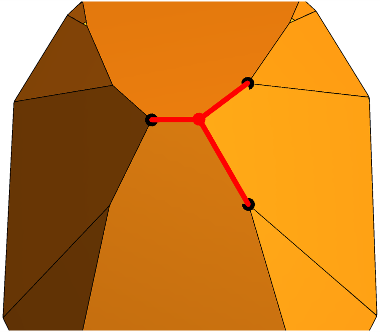
1. Start at a single vertex of $\mathcal{P}_1$.
2. Determine all neighbouring perfect forms. ← Dual Description Problem
3. Keep those that are new.
4. Repeat for each perfect form.



Voronoi's Algorithm

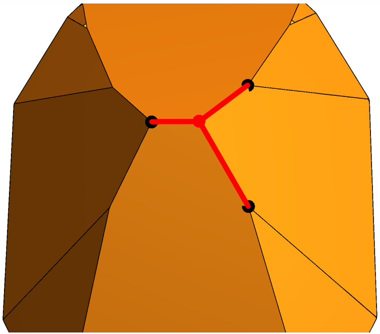
- Voronoi's Algorithm finds all $\boldsymbol{d}$-dimensional perfect forms.

1. Start at a single vertex of $\mathcal{P}_1$.
2. Determine all neighbouring perfect forms. ← Dual Description Problem
3. Keep those that are new. ← Testing Equivalence
4. Repeat for each perfect form.



- Voronoi's Algorithm finds all $\boldsymbol{d}$-dimensional perfect forms.

1. Start at a single vertex of $\mathcal{P}_{\mathbf{1}}$.
2. Determine all neighbouring perfect forms. ← Dual Description Problem
3. Keep those that are new. ← Testing Equivalence
4. Repeat for each perfect form. ↑ #Perfect forms



Canonical functions

- Group G acting on a set X, orbit equality determines an equivalence relation on X.
  ($x \sim y \Leftrightarrow \mathrm{Orbit}(G, x) = \mathrm{Orbit}(G, y)$)

Canonical functions

- Group G acting on a set X, orbit equality determines an equivalence relation on X.
  $(x \sim y \Leftrightarrow \text{Orbit}(G, x) = \text{Orbit}(G, y))$
- **Problem:** given $S \subset X$, determine all orbit equivalence classes under the action G.

Canonical functions

- Group G acting on a set X, orbit equality determines an equivalence relation on X.
  $(x \sim y \Leftrightarrow \text{Orbit}(G, x) = \text{Orbit}(G, y))$
- **Problem:** given $S \subset X$, determine all orbit equivalence classes under the action G.
- **Naive approach:** up to $O(|S|^2)$ orbit equivalence checks.

Canonical functions

- Group G acting on a set X, orbit equality determines an equivalence relation on X.
  $(x \sim y \Leftrightarrow \text{Orbit}(G, x) = \text{Orbit}(G, y))$
- **Problem:** given $S \subset X$, determine all orbit equivalence classes under the action G.
- **Naive approach:** up to $O(|S|^2)$ orbit equivalence checks.
- $|S|$ can be of order 10^9 in our work.

Canonical functions

- Group G acting on a set X, orbit equality determines an equivalence relation on X.
  $(x \sim y \Leftrightarrow \mathrm{Orbit}(G, x) = \mathrm{Orbit}(G, y))$
- **Problem:** given $S \subset X$, determine all orbit equivalence classes under the action G.
- **Naive approach:** up to $O(|S|^2)$ orbit equivalence checks.
- $|S|$ can be of order 10^9 in our work.

Definition: canonical function

We call $\Theta : X \to X$ a **canonical function** if $\Theta(x) \sim x$, and

$$x \sim y \Leftrightarrow \Theta(x) = \Theta(y) \quad \text{for all } x, y \in X.$$

Canonical functions

- Group G acting on a set X, orbit equality determines an equivalence relation on X.
  $(x \sim y \Leftrightarrow \mathrm{Orbit}(G, x) = \mathrm{Orbit}(G, y))$
- **Problem:** given $S \subset X$, determine all orbit equivalence classes under the action G.
- **Naive approach:** up to $O(|S|^2)$ orbit equivalence checks.
- $|S|$ can be of order 10^9 in our work.

Definition: canonical function

We call $\Theta : X \to X$ a **canonical function** if $\Theta(x) \sim x$, and

$$x \sim y \Leftrightarrow \Theta(x) = \Theta(y) \quad \text{for all } x, y \in X.$$

- $|S|$ canonical function evaluations, keep unique ones in $O(|S|)$ using hashmap.

Canonical functions

- Group G acting on a set X, orbit equality determines an equivalence relation on X.
  $(x \sim y \Leftrightarrow \text{Orbit}(G, x) = \text{Orbit}(G, y))$
- **Problem:** given $S \subset X$, determine all orbit equivalence classes under the action G.
- **Naive approach:** up to $O(|S|^2)$ orbit equivalence checks.
- $|S|$ can be of order 10^9 in our work.

Definition: canonical function

We call $\Theta : X \to X$ a **canonical function** if $\Theta(x) \sim x$, and

$$x \sim y \Leftrightarrow \Theta(x) = \Theta(y) \quad \text{for all } x, y \in X.$$

- $|S|$ canonical function evaluations, keep unique ones in $O(|S|)$ using hashmap.
- Used for: **PQF**, **face** and **polyhedral** equivalence.

Example: Arithmetical Equivalence

- Arithmetical equivalence: $\exists \boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$ s.t. $\boldsymbol{Q}' = \boldsymbol{U}^t \boldsymbol{Q} \boldsymbol{U}$. $(\boldsymbol{G} = \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z}), \boldsymbol{X} = \mathcal{S}^d_{>0})$

Example: Arithmetical Equivalence

- Arithmetical equivalence: $\exists \boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$ s.t. $\boldsymbol{Q}' = \boldsymbol{U}^t \boldsymbol{Q} \boldsymbol{U}$. $(\boldsymbol{G} = \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z}), \boldsymbol{X} = \mathcal{S}^d_{>0})$
- Gives an **isometry**: $\boldsymbol{U} \cdot \mathrm{Min}\,\boldsymbol{Q}' = \mathrm{Min}\,\boldsymbol{Q}$. (w.r.t. $\boldsymbol{Q}'$ and $\boldsymbol{Q}$ respectively)

Example: Arithmetical Equivalence

- Arithmetical equivalence: $\exists \boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$ s.t. $\boldsymbol{Q}' = \boldsymbol{U}^t \boldsymbol{Q} \boldsymbol{U}$. $(\boldsymbol{G} = \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z}), \boldsymbol{X} = \mathcal{S}^d_{>0})$
- Gives an **isometry**: $\boldsymbol{U} \cdot \mathrm{Min}\,\boldsymbol{Q}' = \mathrm{Min}\,\boldsymbol{Q}$. (w.r.t. $\boldsymbol{Q}'$ and $\boldsymbol{Q}$ respectively)
- If $\mathrm{span}_{\mathbb{Z}}(\mathrm{Min}\,\boldsymbol{Q}) = \mathbb{Z}^{\boldsymbol{d}}$, then reverse implication is also true. (assume for now)

Example: Arithmetical Equivalence

- Arithmetical equivalence: $\exists \boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$ s.t. $\boldsymbol{Q}' = \boldsymbol{U}^t \boldsymbol{Q} \boldsymbol{U}$. $(\boldsymbol{G} = \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z}), \boldsymbol{X} = \mathcal{S}^{d}_{>0})$
- Gives an **isometry**: $\boldsymbol{U} \cdot \mathrm{Min}\,\boldsymbol{Q}' = \mathrm{Min}\,\boldsymbol{Q}$. (w.r.t. $\boldsymbol{Q}'$ and $\boldsymbol{Q}$ respectively)
- If $\mathrm{span}_{\mathbb{Z}}(\mathrm{Min}\,\boldsymbol{Q}) = \mathbb{Z}^{\boldsymbol{d}}$, then reverse implication is also true. (assume for now)
- Complete graph $\mathcal{G}_{\boldsymbol{Q}}$ with vertices Min $\boldsymbol{Q}$, and weight $\boldsymbol{x}^t \boldsymbol{Q} \boldsymbol{y}$ on each edge $(\boldsymbol{x}, \boldsymbol{y})$.

Example: Arithmetical Equivalence

- Arithmetical equivalence: $\exists U \in \mathrm{GL}_d(\mathbb{Z})$ s.t. $Q' = U^t Q U$. $(G = \mathrm{GL}_d(\mathbb{Z}), X = \mathcal{S}^d_{>0})$
- Gives an **isometry**: $U \cdot \mathrm{Min}\, Q' = \mathrm{Min}\, Q$. (w.r.t. Q' and Q respectively)
- If $\mathrm{span}_{\mathbb{Z}}(\mathrm{Min}\, Q) = \mathbb{Z}^d$, then reverse implication is also true. (assume for now)
- Complete graph $\mathcal{G}_Q$ with vertices Min Q, and weight $x^t Q y$ on each edge (x, y).
- Then

$$Q \sim Q' \Leftrightarrow \mathcal{G}_Q \cong \mathcal{G}_{Q'} \qquad \text{(graph isomorphism)}$$

Example: Arithmetical Equivalence

- Arithmetical equivalence: $\exists U \in \mathrm{GL}_d(\mathbb{Z})$ s.t. $Q' = U^t Q U$. $(G = \mathrm{GL}_d(\mathbb{Z}), X = \mathcal{S}^d_{>0})$
- Gives an **isometry**: $U \cdot \mathrm{Min}\, Q' = \mathrm{Min}\, Q$. (w.r.t. Q' and Q respectively)
- If $\mathrm{span}_{\mathbb{Z}}(\mathrm{Min}\, Q) = \mathbb{Z}^d$, then reverse implication is also true. (assume for now)
- Complete graph $\mathcal{G}_Q$ with vertices Min Q, and weight $x^t Q y$ on each edge (x, y).
- Then

$$Q \sim Q' \Leftrightarrow \mathcal{G}_Q \cong \mathcal{G}_{Q'} \qquad \text{(graph isomorphism)}$$

- Construct a canonical form using canonical graph labeling algorithms.

Example: Arithmetical Equivalence

- Arithmetical equivalence: $\exists U \in \mathrm{GL}_d(\mathbb{Z})$ s.t. $Q' = U^t Q U$. $(G = \mathrm{GL}_d(\mathbb{Z}), X = \mathcal{S}^d_{>0})$
- Gives an **isometry**: $U \cdot \mathrm{Min}\, Q' = \mathrm{Min}\, Q$. (w.r.t. Q' and Q respectively)
- If $\mathrm{span}_{\mathbb{Z}}(\mathrm{Min}\, Q) = \mathbb{Z}^d$, then reverse implication is also true. (assume for now)
- Complete graph $\mathcal{G}_Q$ with vertices Min Q, and weight $x^t Q y$ on each edge (x, y).
- Then

$$Q \sim Q' \Leftrightarrow \mathcal{G}_Q \cong \mathcal{G}_{Q'} \qquad \text{(graph isomorphism)}$$

- Construct a canonical form using canonical graph labeling algorithms.
- With more improvements: ± 0.3ms per perfect form in dimension 9.

Example: Arithmetical Equivalence

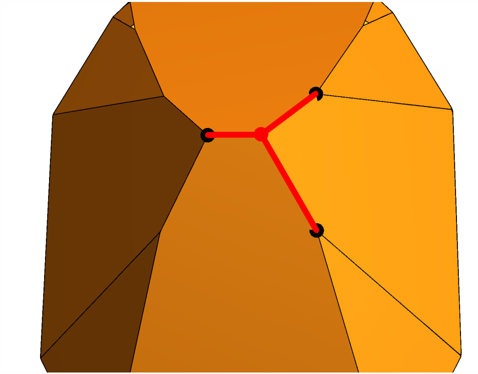
- Arithmetical equivalence: $\exists \boldsymbol{U} \in \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z})$ s.t. $\boldsymbol{Q}' = \boldsymbol{U}^t \boldsymbol{Q} \boldsymbol{U}$. $(\boldsymbol{G} = \mathrm{GL}_{\boldsymbol{d}}(\mathbb{Z}), \boldsymbol{X} = \mathcal{S}^d_{>0})$
- Gives an **isometry**: $\boldsymbol{U} \cdot \mathrm{Min}\,\boldsymbol{Q}' = \mathrm{Min}\,\boldsymbol{Q}$. (w.r.t. $\boldsymbol{Q}'$ and $\boldsymbol{Q}$ respectively)
- If $\mathrm{span}_{\mathbb{Z}}(\mathrm{Min}\,\boldsymbol{Q}) = \mathbb{Z}^{\boldsymbol{d}}$, then reverse implication is also true. (assume for now)
- Complete graph $\mathcal{G}_{\boldsymbol{Q}}$ with vertices Min $\boldsymbol{Q}$, and weight $\boldsymbol{x}^t \boldsymbol{Q} \boldsymbol{y}$ on each edge $(\boldsymbol{x}, \boldsymbol{y})$.
- Then

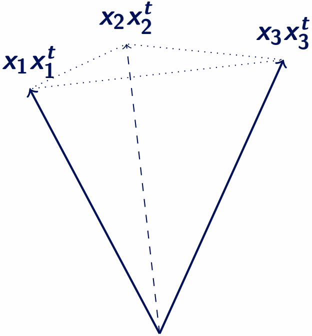
$$\boldsymbol{Q} \sim \boldsymbol{Q}' \Leftrightarrow \mathcal{G}_{\boldsymbol{Q}} \cong \mathcal{G}_{\boldsymbol{Q}'} \qquad \text{(graph isomorphism)}$$

- Construct a canonical form using canonical graph labeling algorithms.
- With more improvements: $\pm\mathbf{0.3}$ms per perfect form in dimension $\mathbf{9}$.
- **Details:** A canonical form for positive definite matrices. [ANTS 2020, **DS**HV**v**W]

Dual Description Problem

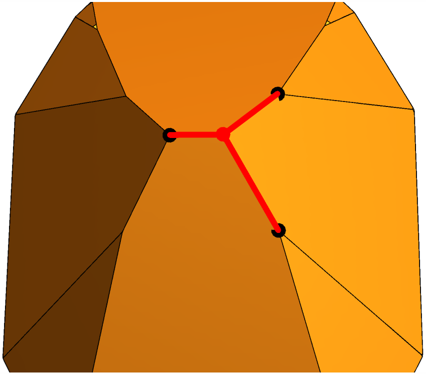
- A (pointed) polyhedral cone $\mathcal{C} \subset \mathbb{R}^n$ can either be given by **facet inequalities** or by **extreme rays**.

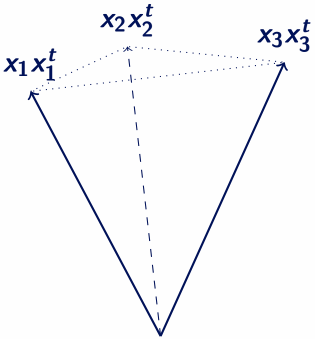




Dual Description Problem

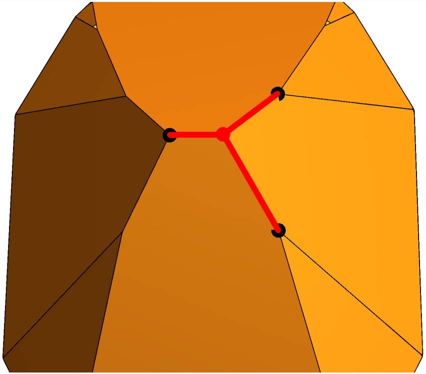
- A (pointed) polyhedral cone $\mathcal{C} \subset \mathbb{R}^n$ can either be given by **facet inequalities** or by **extreme rays**.
- Dual Description problem: facets $\Leftrightarrow$ extreme rays.

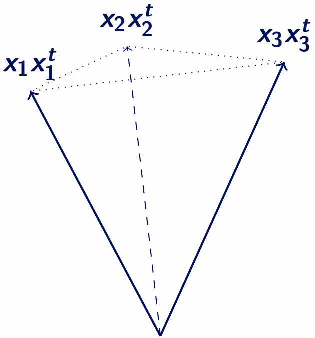




Dual Description Problem

- A (pointed) polyhedral cone $\mathcal{C} \subset \mathbb{R}^n$ can either be given by **facet inequalities** or by **extreme rays**.
- Dual Description problem: facets $\Leftrightarrow$ extreme rays.
- The two directions are equivalent by duality.





Too many neighbours

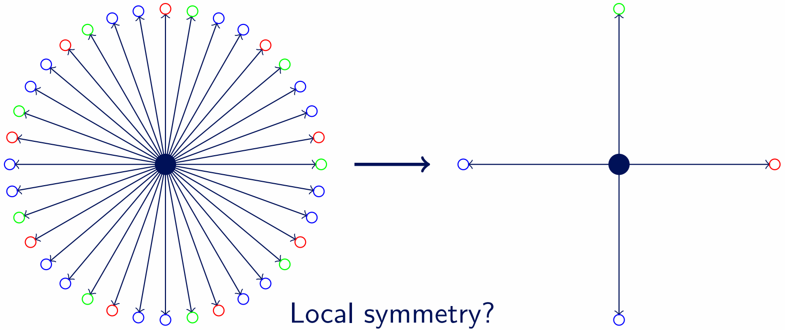
- Let $\mathcal{P}(\boldsymbol{Q})$ be the local pointed cone at $\boldsymbol{Q}$.

Too many neighbours

- Let $\mathcal{P}(\boldsymbol{Q})$ be the local pointed cone at $\boldsymbol{Q}$.
- $\mathcal{P}(\boldsymbol{Q_{E_8}})$: **120** facets in **36** dimensional space: **25.075.566.937.584** extreme rays...

Too many neighbours

- Let $\mathcal{P}(Q)$ be the local pointed cone at Q.
- $\mathcal{P}(Q_{E_8})$: **120** facets in **36** dimensional space: **25.075.566.937.584** extreme rays...
- Many rays point to equivalent forms: $Q + \alpha_1 R_1 \sim Q + \alpha_2 R_2$



Local symmetry

- Aut$\boldsymbol{Q}$ induces linear symmetries on $\mathcal{P}(\boldsymbol{Q})$. (Aut$\boldsymbol{Q}/\{\pm\} \subset$ Aut$(\mathcal{P})$)

Local symmetry

- $\mathrm{Aut}\boldsymbol{Q}$ induces linear symmetries on $\boldsymbol{\mathcal{P}(Q)}$. $(\mathrm{Aut}\boldsymbol{Q}/\{\pm\} \subset \mathrm{Aut}(\boldsymbol{\mathcal{P}}))$
- For all $\boldsymbol{U} \in \mathrm{Aut}\boldsymbol{Q}$, $\boldsymbol{R}$ is a ray if and only if $\boldsymbol{U^t R U}$ is a ray, and:

$$\boldsymbol{Q + R \sim U^t(Q + R)U = Q + U^t R U}$$

Local symmetry

- $\mathrm{Aut}\,\boldsymbol{Q}$ induces linear symmetries on $\mathcal{P}(\boldsymbol{Q})$. ($\mathrm{Aut}\,\boldsymbol{Q}/\{\pm\} \subset \mathrm{Aut}(\mathcal{P})$)
- For all $\boldsymbol{U} \in \mathrm{Aut}\,\boldsymbol{Q}$, $\boldsymbol{R}$ is a ray if and only if $\boldsymbol{U}^t\boldsymbol{R}\boldsymbol{U}$ is a ray, and:

$$\boldsymbol{Q} + \boldsymbol{R} \sim \boldsymbol{U}^t(\boldsymbol{Q} + \boldsymbol{R})\boldsymbol{U} = \boldsymbol{Q} + \boldsymbol{U}^t\boldsymbol{R}\boldsymbol{U}$$

Problem: Dual description problem under symmetry

Compute all **orbits** of extreme rays under some symmetry group $\boldsymbol{G} \subset \mathrm{Aut}(\mathcal{P})$.

Local symmetry

- Aut$\boldsymbol{Q}$ induces linear symmetries on $\mathcal{P}(\boldsymbol{Q})$. (Aut$\boldsymbol{Q}/\{\pm\} \subset \mathrm{Aut}(\mathcal{P})$)
- For all $\boldsymbol{U} \in \mathrm{Aut}\boldsymbol{Q}$, $\boldsymbol{R}$ is a ray if and only if $\boldsymbol{U}^t\boldsymbol{R}\boldsymbol{U}$ is a ray, and:

$$\boldsymbol{Q} + \boldsymbol{R} \sim \boldsymbol{U}^t(\boldsymbol{Q} + \boldsymbol{R})\boldsymbol{U} = \boldsymbol{Q} + \boldsymbol{U}^t\boldsymbol{R}\boldsymbol{U}$$

Problem: Dual description problem under symmetry

Compute all **orbits** of extreme rays under some symmetry group $\boldsymbol{G} \subset \mathrm{Aut}(\mathcal{P})$.

Theorem: Dutour, Schürmann, Vallentin, 2005

$\mathcal{P}(\boldsymbol{Q}_{\boldsymbol{E}_8})$ with **120** facets has **25.075.566.937.584** extreme rays, but 'only' **83.092** orbits under Aut$\boldsymbol{Q}_{\boldsymbol{E}_8}$.

Local symmetry

- $\mathrm{Aut}\, Q$ induces linear symmetries on $\mathcal{P}(Q)$. $(\mathrm{Aut}\, Q/\{\pm\} \subset \mathrm{Aut}(\mathcal{P}))$
- For all $U \in \mathrm{Aut}\, Q$, R is a ray if and only if $U^t R U$ is a ray, and:

$$Q + R \sim U^t (Q + R) U = Q + U^t R U$$

Problem: Dual description problem under symmetry

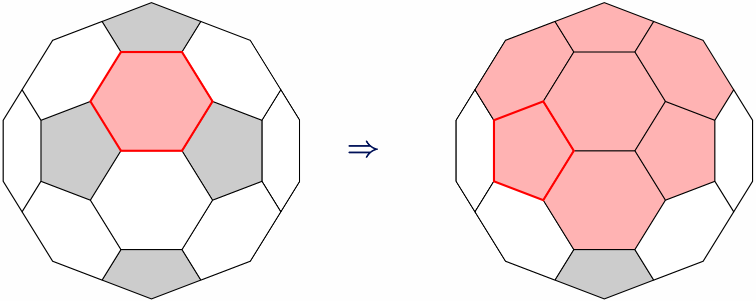
Compute all **orbits** of extreme rays under some symmetry group $G \subset \mathrm{Aut}(\mathcal{P})$.

Theorem: Dutour, Schürmann, Vallentin, 2005

$\mathcal{P}(Q_{E_8})$ with **120** facets has **25.075.566.937.584** extreme rays, but 'only' **83.092** orbits under $\mathrm{Aut}\, Q_{E_8}$.

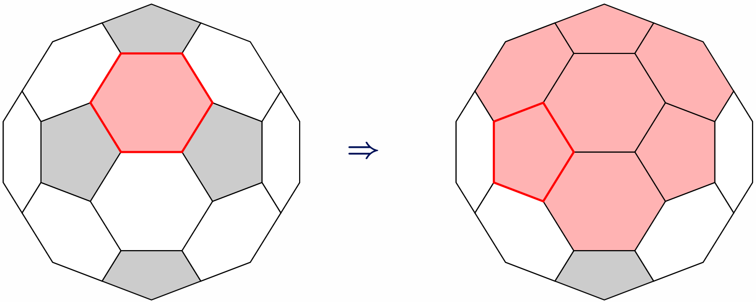
- **Even harder:** $\mathcal{P}(Q_{\Lambda_9})$ has **136** facets in a **45**-dimensional space.

Adjacency Decomposition Method



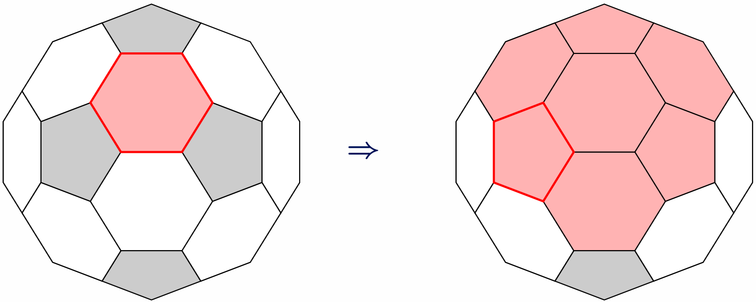
- Two $\boldsymbol{k}$-dimensional faces $\boldsymbol{F_1}, \boldsymbol{F_2}$ are **adjacent** if $\mathbf{dim}(\boldsymbol{F_1} \cap \boldsymbol{F_2}) = \boldsymbol{k} - \mathbf{1}$.

Adjacency Decomposition Method



- Two k-dimensional faces F_1, F_2 are **adjacent** if $\mathbf{dim}(F_1 \cap F_2) = k - 1$.
- Enumerate adjacency graph up to equivalence (just like Voronoi's algorithm!)

Adjacency Decomposition Method



- Two k-dimensional faces F_1, F_2 are **adjacent** if $\dim(F_1 \cap F_2) = k - 1$.
- Enumerate adjacency graph up to equivalence (just like Voronoi's algorithm!)
- $\{F_2 : \text{adjacent to } F_1\} \leftrightarrow \{\text{facets } H \text{ of } F_1\}$ $(H = F_1 \cap F_2)$.

Adjacency Decomposition Method

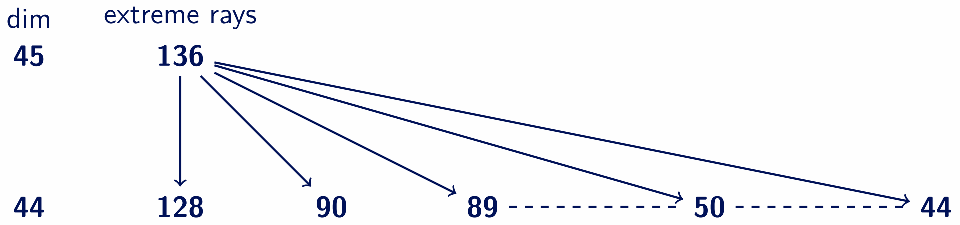
- Best explained in dual setting: $\mathcal{C} = \text{cone}([y_1, \dots, y_m\}) \subset \mathbb{R}^n$ with $G \subset \text{Aut}(\mathcal{C})$.

Algorithm: Adjacency Decomposition Method

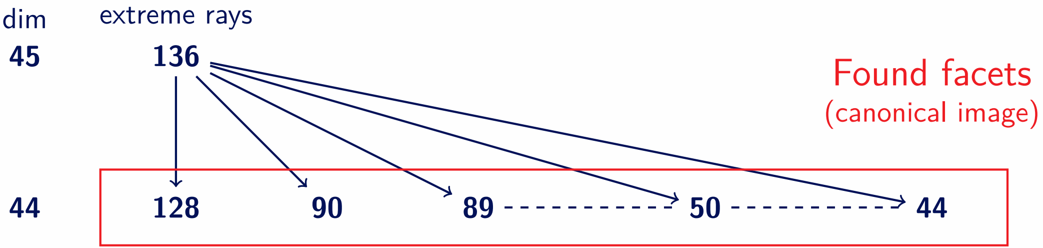
1. Find at least one facet F.
2. Determine facets $H_1, \dots, H_k$ of F, .i.e. ridges of $\mathcal{C}$ contained in F.
3. For all i
   - compute facet F_i of $\mathcal{C}$ such that $H_i = F \cap F_i$.
   - Keep F_i if G-inequivalent to all found facets.
4. Repeat **(2)** and **(3)** for each new facet.

- Step **(2)** is again Dual Description problem but dimension $n - 1$ and only with **extreme rays contained in F**.
- If still difficult, **recurse**: $G' = \text{Stab}(G, F)$.

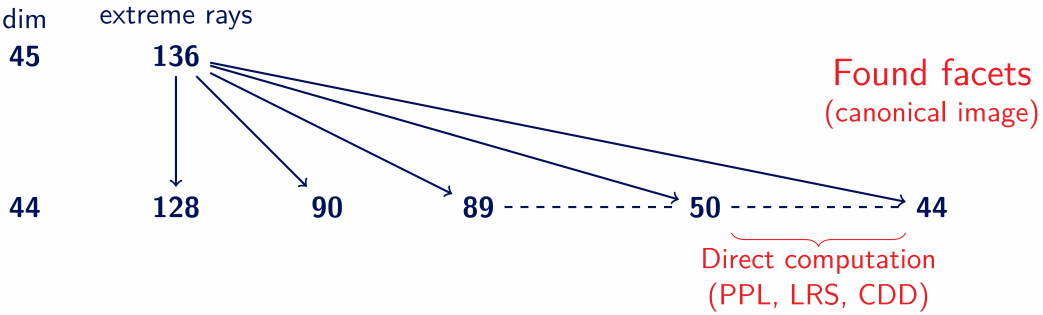
Recursive Adjacency Decomposition Method



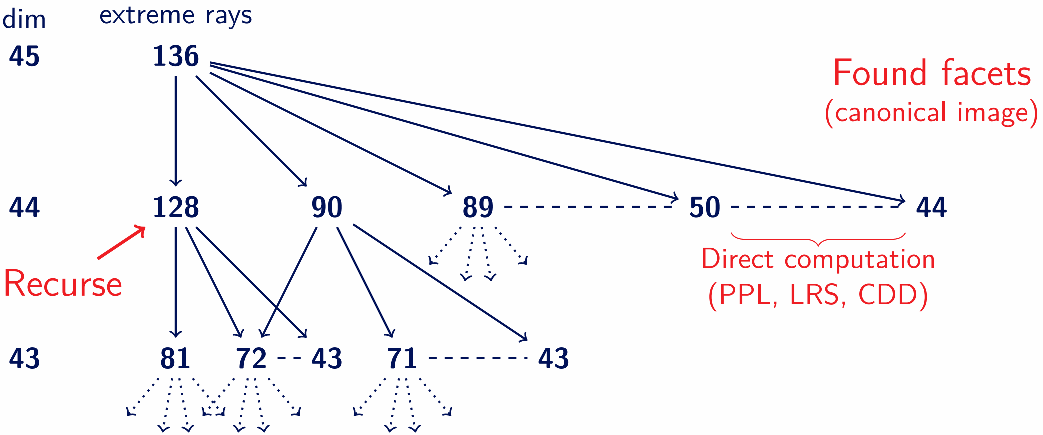
Recursive Adjacency Decomposition Method



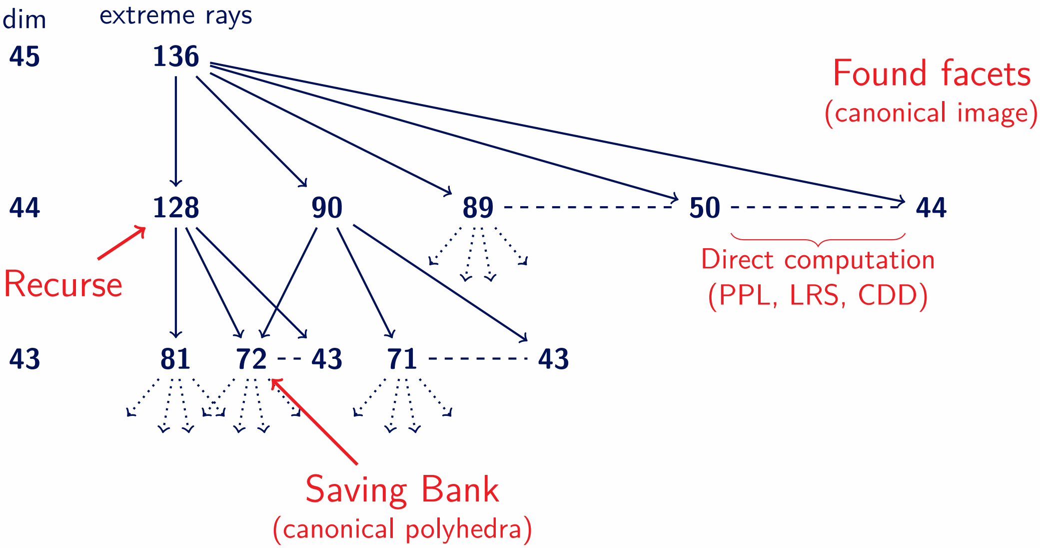
Recursive Adjacency Decomposition Method



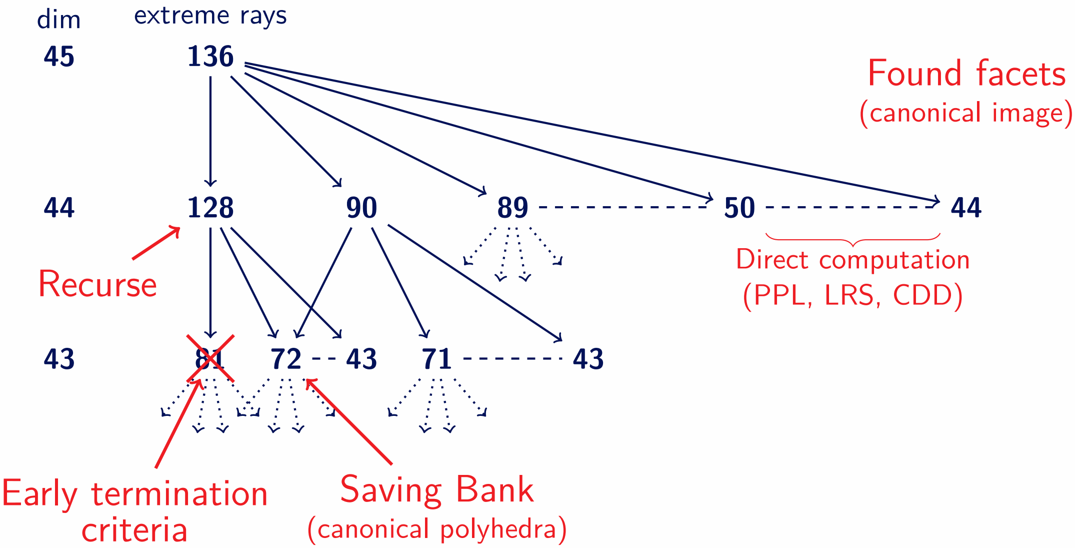
Recursive Adjacency Decomposition Method



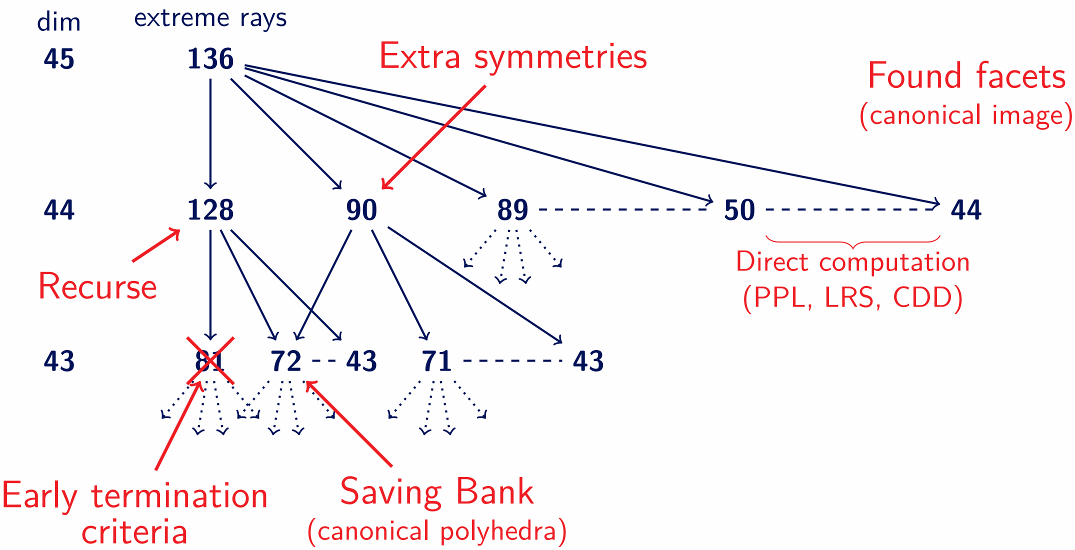
Recursive Adjacency Decomposition Method



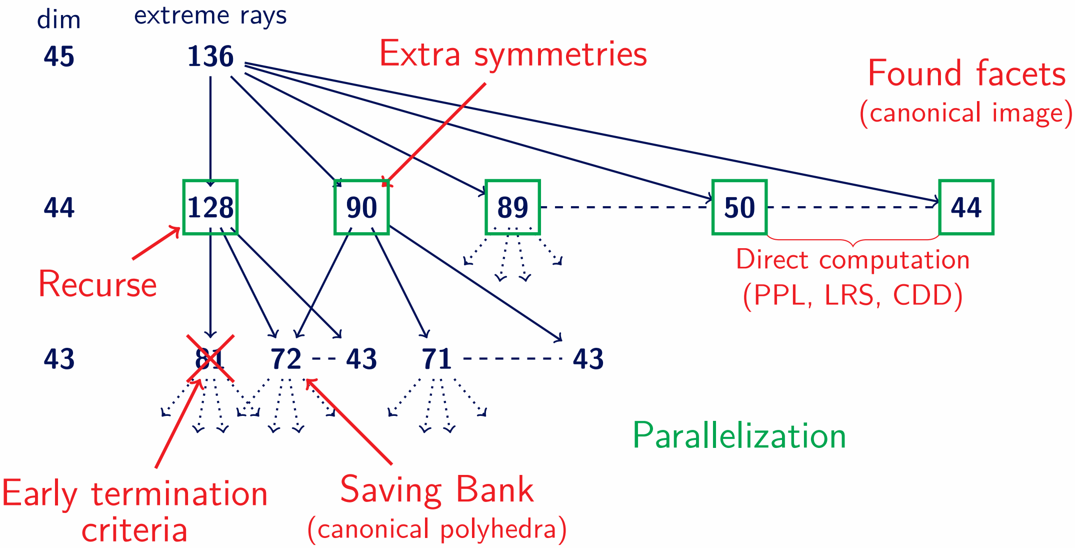
Recursive Adjacency Decomposition Method



Recursive Adjacency Decomposition Method



Recursive Adjacency Decomposition Method



Results

Lattice packing problem in dimension 9

8 years and **$\pm$ 3 000 000 core hours later...**

Lattice packing problem in dimension 9

8 years and **± 3 000 000 core hours later...**

Theorem: Main result

There are precisely **2 237 251 040** non-similar perfect forms in dimension **9**.

Lattice packing problem in dimension 9

8 years and **$\pm$ 3 000 000 core hours later...**

Theorem: Main result

There are precisely **2 237 251 040** non-similar perfect forms in dimension **9**.

Corollary: Lattice Packing Problem in dimension **9**

The Laminated lattice $\boldsymbol{\Lambda_9}$ is the unique densest lattice packing in dimension **9**.

Lattice packing problem in dimension 9

8 years and $\pm$ **3 000 000 core hours later...**

Theorem: Main result

There are precisely **2 237 251 040** non-similar perfect forms in dimension **9**.

Corollary: Lattice Packing Problem in dimension **9**

The Laminated lattice $\boldsymbol{\Lambda_9}$ is the unique densest lattice packing in dimension **9**.

The Hermite constant in dimension **9** is $\gamma_9 = 2$.

Lattice packing problem in dimension 9

8 years and **± 3 000 000 core hours later...**

Theorem: Main result

There are precisely **2 237 251 040** non-similar perfect forms in dimension **9**.

Corollary: Lattice Packing Problem in dimension **9**

The Laminated lattice Λ_9 is the unique densest lattice packing in dimension **9**.

The Hermite constant in dimension **9** is $\gamma_9 = 2$.

Theorem: Kissing numbers

The set of possible kissing numbers $|\mathrm{Min}(L)|$, for a lattice $L \subset \mathbb{R}^9$ of dimension **9**, is $2 \cdot \{1, \ldots, 91, 99, 120, \ldots, 129, 136\}$.

All perfect forms by their kissing number

$\|\min(Q)\|/2$	#	$\|\min(Q)\|/2$	#	$\|\min(Q)\|/2$	#
45	1 353 947 672	**61**	2 244	**77**	1
46	471 756 975	**62**	1 713	**78**	1
47	267 588 732	**63**	641	**79**	2
48	84 473 357	**64**	634	**80**	12
49	37 278 163	**65**	236	**81**	3
50	13 324 560	**66**	203	**82**	4
51	5 299 974	**67**	172	**84**	2
52	2 009 292	**68**	74	**85**	2
53	903 943	**69**	44	**88**	1
54	366 796	**70**	42	**90**	2
55	155 182	**71**	26	**91**	1
56	78 919	**72**	21	**99**	1
57	31 113	**73**	7	**129**	1
58	17 207	**74**	3	**136**	1
59	8 231	**75**	4		
60	4 820	**76**	6		

All perfect forms by their kissing number

99.9991% of all forms

< 5% of runtime.

$\lvert\min(Q)\rvert/2$	#	$\lvert\min(Q)\rvert/2$	#	$\lvert\min(Q)\rvert/2$	#
45	1 353 947 672	**61**	2 244	**77**	1
46	471 756 975	**62**	1 713	**78**	1
47	267 588 732	**63**	641	**79**	2
48	84 473 357	**64**	634	**80**	12
49	37 278 163	**65**	236	**81**	3
50	13 324 560	**66**	203	**82**	4
51	5 299 974	**67**	172	**84**	2
52	2 009 292	**68**	74	**85**	2
53	903 943	**69**	44	**88**	1
54	366 796	**70**	42	**90**	2
55	155 182	**71**	26	**91**	1
56	78 919	**72**	21	**99**	1
57	31 113	**73**	7	**129**	1
58	17 207	**74**	3	**136**	1
59	8 231	**75**	4		
60	4 820	**76**	6		

High incidence cases

For comparison: $\mathcal{P}(Q_{E_8})$ now takes **9** core hours (before a few month).

High incidence cases

For comparison: $\mathcal{P}(Q_{E_8})$ now takes **9** core hours (before a few month).

Table: Cost of dual description cases with more than **50**k core hours. These cases account for **1.5** million of the total amount of **2** million core hours spent on dual description instances.

$\vert\min(Q)\vert/2$	Core hours	$\vert\mathrm{linaut}(P)\vert$	rays (orbits)	$\vert\mathrm{aut}(Q)\vert$	neighbours (orbits)
136	59 277	660 602 880	64 001 686	10 321 920	1 038 153 863
84	75 467	12 288	171 496 157	384	1 514 557 045
99	84 197	589 824	137 739 671	18 432	1 842 205 495
90	85 349	73 728	185 824 962	2 304	2 058 568 310
74	95 784	128	333 146 387	16	1 257 559 244
80	97 118	7 680	108 828 919	480	764 775 430
81	181 570	1 296	254 734 260	2 592	254 734 260
80	219 437	128	772 745 513	256	772 745 513
82	245 030	432	680 747 757	864	680 747 757
76	355 554	24	1 549 616 491	48	1 549 616 491

Hard to reach perfect forms

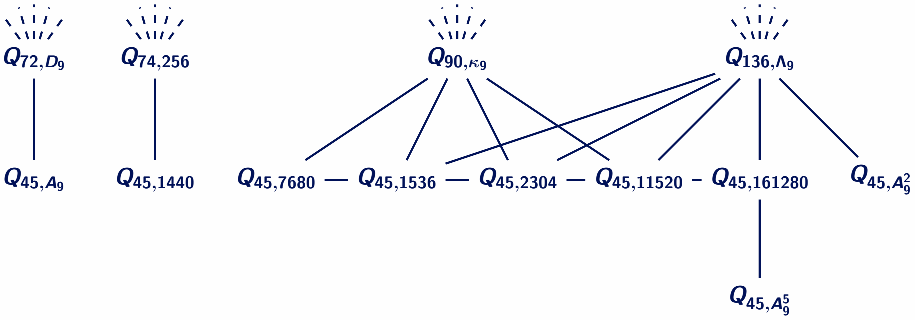


Figure: Part of Voronoi graph showing all perfect forms that are only connected via high-incidence perfect forms.

- All other forms are connected via forms with $|\mathrm{Min}\,\boldsymbol{Q}| \leq \mathbf{2 \cdot 58}$.

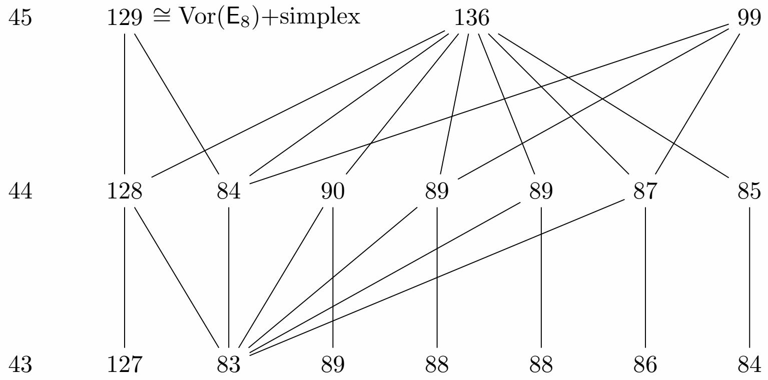
Thank you!

Preprint:

https://arxiv.org/abs/2508.20719

Thank you!

Kissing numbers



Partial face lattice:

- all facets: $|F| \geq 84$ (dim$= 44$)
- all faces: $|F| \geq 92$
- other relevant subfaces.

up to combinatorial equivalence

Canonical functions - Examples

Graph Isomorphism: $X = \{$n-vertex graphs $\mathcal{G} = (V, E)\}$, $G = \mathrm{Sym}(n)$.

Well researched area. Babai: canonical function in quasi-polynomial time.

Important: Many practically efficient canonical functions and libraries.

PQF equivalence: $X = S^d_{>0}(\mathbb{Q})$, $G = \mathrm{GL}_d(\mathbb{Z})$, $Q \circ U := U^t Q U$

Difficulty: infinite size orbits. **Idea:** G also acts on finite set $\mathrm{Min}\,(Q)$

"A canonical form for positive definite matrices" [**DS**HV**vW**20]. $\longrightarrow$ **GI**

Polyhedral Cone: $X = \{\{v_1, \ldots, v_m\} \subset \mathbb{R}^n\}$, $G = \mathrm{GL}_n(\mathbb{R})$

"Computing symmetry groups of polyhedra" [B**DS**PRS14] $\longrightarrow$ **GI**

Face equivalence: $X = \{$faces of $P\}$, $G \subset \mathrm{Aut}(P)$.

Permutation group acting on sets: "Minimal and Canonical images" [JJPW19]

Face equivalence

- Each face can be described by the set of rays $F \subset [m]$ contained in it.
- Polyhedral symmetry group can be described as a permutation group $G \subset \mathrm{Sym}_m$.
- $X = \{F \subset [m] : F \text{ is a face of } \mathcal{P}\}$, $\sigma \circ F = \sigma(F) = \{\sigma(x) : x \in F\}$.
- Define total ordering $\preccurlyeq$ on $\mathcal{P}([m])$, then

$$\theta_m(F) = \min_{\preccurlyeq}(\mathrm{Orb}(G, F))$$

  is canonical. Use stabilizer chain to calculate $\theta_m(F)$ without full enumeration.
- (*Minimal and Canonical images*, JJPW, 2017): dynamical ordering tailored for each orbit. Constructed in a canonical way during the algorithm.
- Up to multiple orders of magnitude faster. (**1 min. vs 2 ms** in GAP)
- **Mathieu** ported the GAP routines and the package to C++: **even faster**.