max planck institut informatik

Fast Algorithms for Integer Programming in Fixed Dimension

Friedrich Eisenbrand

Integer Programming I

- Variables: $x(1),\ldots,x(n)$
- Linear constraints: $a_{i1}x(1)+\cdots+a_{in}x(n) \leqslant b(i)$, for $i=1,\ldots,m$
- Linear objective function: $c(1)x(1)+\cdots+c(n)x(n)$

- Task: Find integer assignment to $x(1),\ldots,x(n)$ such that all constraints are satisfied and objective function is maximized.

GCDs and IP

Theorem. $\gcd(a,b) = \min\{xa+yb \mid x,y \in \mathbb{Z}, xa+yb \geqslant 1\}$

$$\begin{array}{ll} \textit{minimize} & xa+yb \\ \textit{condition} & xa+yb \geqslant 1 \\ & x,y \in \mathbb{Z}. \end{array}$$

GCDs and IP

Theorem. $\gcd(a,b) = \min\{xa+yb \mid x,y \in \mathbb{Z}, xa+yb \geqslant 1\}$

$$\begin{array}{ll} \textit{minimize} & xa+yb \\ \textit{condition} & xa+yb \geqslant 1 \\ & x,y \in \mathbb{Z}. \end{array}$$

Integer Programming: Combinatorics & Number Theory

Euclidean Algorithm

- Input: Integers $a \geqslant b > 0$
- while $b \neq 0$
  - Compute $q \geqslant 1$ and $0 \leqslant r < b$ with $a = qb+r$
  - $a \longleftarrow b$
  - $b \longleftarrow r$
- **return** a

Euclidean Algorithm

- Input: Integers $a \geqslant b > 0$
- while $b \neq 0$
  - Compute $q \geqslant 1$ and $0 \leqslant r < b$ with $a = qb+r$
  - $a \longleftarrow b$
  - $b \longleftarrow r$
- **return** a

Analysis:

- $r \leqslant a/2 \Longrightarrow$ running time is $O(\log a)$
- Running time depends on binary encoding length of numbers

Goals of this course

Primary goals:

- Develop algorithm for IP in fixed dimension with fixed number of constraints which runs in linear time (match complexity of Euclidean algorithm)
- Reduce the dependence of the running time on the number of constraints (Clarkson's algorithm)

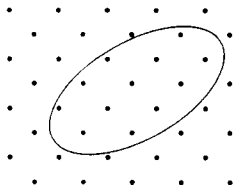
To achieve that we need to learn about:

- Lattices
- Basis reduction especially LLL algorithm
- Flatness theorem
- Integer feasibility in polynomial time

Integer feasibility

The feasibility problem for ellipsoids

- $A \in \mathbb{Q}^{n \times n}$ rational nonsingular matrix
- $a \in \mathbb{Q}^n$ rational vector
- $E(A,a) = \{x \in \mathbb{R}^n \mid \|A(x-a)\| \leqslant 1\}$ rational ellipsoid defined by A and a
- Question: $E(A,a) \cap \mathbb{Z}^n = \emptyset$?



A re-formulation

- $A \in \mathbb{Q}^{n \times n}$ rational nonsingular matrix
- $w \in \mathbb{Q}^n$ rational vector ($w = Aa$)
- $\Lambda(A) = \{Ax \mid x \in \mathbb{Z}^n\}$
- Question: What is the point in $\Lambda(A)$ which is closest to w ?

Lattices

A Lattice is a set:

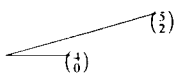
$$\Lambda(A) = \{Ax \mid x \in \mathbb{Z}^n\}$$

where $A \in \mathbb{Q}^{n \times n}$ is nonsingular rational matrix.

- A is basis of Λ.

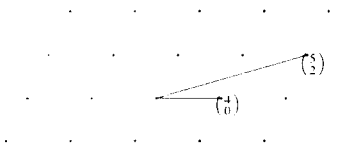
Lattices

$$A = \begin{pmatrix} 4 & 5 \\ 0 & 2 \end{pmatrix}$$



Lattices

$$A = \begin{pmatrix} 4 & 5 \\ 0 & 2 \end{pmatrix}$$



Exercise

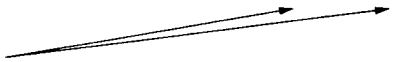
- Let A be a lattice basis of Λ. Suppose that B originates from A by:
  - Swapping columns.
  - Subtracting integer multiples of a column from another column.

  Show that B is also a basis of Λ.
- Show that $\left(\begin{smallmatrix} 4 & 5 \\ 0 & 2 \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} 4 & 1 \\ 0 & 2 \end{smallmatrix}\right)$ generate the same lattice. What is the shortest vector of this lattice?

The central lattice problems

- Given a nonsingular matrix $A \in \mathbb{Q}^{n \times n}$ and a vector $w \in \mathbb{Q}^n$
- Closest vector problem: Determine $v \in \Lambda(A)$ with $\|v - w\|$ minimal CV
- Shortest vector problem: Determine $v \in \Lambda(A) - \{0\}$ with $\|v\|$ minimal SV

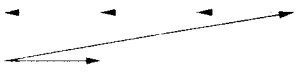
Quiz: What is the shortest vector?



Quiz: What is the shortest vector?

Quiz: What is the shortest vector?

Quiz: What is the shortest vector?



Quiz: What is the shortest vector?

Exercise

- Let A be an orthogonal matrix, i.e., columns are orthogonal to each other. Show that the shortest vector of $\Lambda(A)$ w.r.t. ℓ_2 is the shortest vector of the basis A.

The lattice determinant

Let A and B be bases of Λ

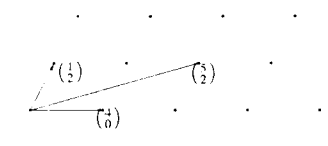
- There exists integer matrix Q_1 such that $B = AQ_1$
- There exists integer matrix Q_2 such that $A = BQ_2$
- Thus $A = Q_2 Q_1 A$, thus $Q_1 Q_2 = I_n$.
- $1 = \det(Q_2 Q_1) = \det(Q_2)\det(Q_1)$
- Q_2 and Q_1 integer matrices: $\det(Q_1), \det(Q_2) = \pm 1$
- $|\det(A)| = |\det(B)|$.

Lattice determinant:

$\det(\Lambda) = |\det(A)|$, where A is basis of Λ.

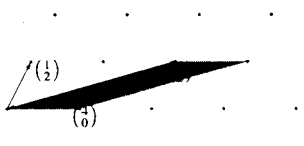
Example lattice determinant

- Lattice determinant is volume of parallelepiped of basis elements.
- The two bases below are $\begin{pmatrix} 4 & 5 \\ 0 & 2 \end{pmatrix}$ and $\begin{pmatrix} 4 & 1 \\ 0 & 2 \end{pmatrix}$.



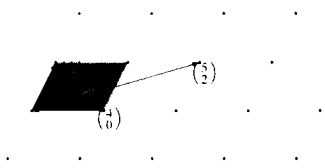
Example lattice determinant

- Lattice determinant is volume of parallelepiped of basis elements.
- The two bases below are $\begin{pmatrix} 4 & 5 \\ 0 & 2 \end{pmatrix}$ and $\begin{pmatrix} 4 & 1 \\ 0 & 2 \end{pmatrix}$.



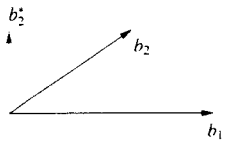
Example lattice determinant

- Lattice determinant is volume of parallelepiped of basis elements.
- The two bases below are $\begin{pmatrix} 4 & 5 \\ 0 & 2 \end{pmatrix}$ and $\begin{pmatrix} 4 & 1 \\ 0 & 2 \end{pmatrix}$.



Gram-Schmidt orthogonalization

- b_1 and b_2 vectors.
- We search b_2^* orthogonal to b_1 s.t. (b_1, b_2^*) generate same vectorspace as (b_1, b_2)



- $b_2^* = b_2 - \mu b_1$
- $0 = \langle b_2^*, b_1 \rangle = \langle b_1, b_2 \rangle - \mu \langle b_1, b_1 \rangle$
- $\mu = \langle b_1, b_2 \rangle / \langle b_1, b_1 \rangle$

Gram-Schmidt orthogonalization

- Input: $b_1, \ldots, b_n \in \mathbb{R}^n$
- Output: $b_1^*, \ldots, b_n^* \in \mathbb{R}^n$ pairwise orthogonal and $\langle b_1^*, \ldots, b_k^* \rangle = \langle b_1, \ldots, b_k \rangle$ for all $k = 1, \ldots, n$
- $b_1^* \leftarrow b_1$
- For $i = 2, \ldots, k$
  - $b_i^* \leftarrow b_i - \sum_{j=1}^{i-1} \mu(i,j) b_j^*$, where $\mu(i,j)$ satisfies $\langle b_i - \mu(i,j) b_j^*, b_j^* \rangle = 0$

Gram-Schmidt orthogonalization

- Decomposes matrix $B \in \mathbb{Z}^{n \times n}$ into

$$B = B^* \begin{pmatrix} 1 & & \mu(i,j) \\ & \ddots & \\ 0 & & 1 \end{pmatrix},$$

where B^* is matrix with pairwise orthogonal columns.

Exercise

- Let $B = (b_1, \ldots, b_{i-1}, b_i, b_{i+1}, b_{i+2}, \ldots, b_n)$ and $\bar{B} = (b_1, \ldots, b_{i-2}, b_{i+1}, b_i, b_{i+2}, \ldots, b_n)$ be two lattice bases. Notice that $\bar{B}$ originates from B via swapping the i-th and $i+1$-st column. Prove that B^* and $\bar{B}^*$ only differ in the i-th and $i+1$-st column.

Exercise

Let $B = B^* \begin{pmatrix} 1 & & \mu(i,j) \\ & \ddots & \\ 0 & & 1 \end{pmatrix}$ be the GSO of B

- Show that $\|b_i^*\| \leq \|b_i\|$ for $i = 1, \ldots, n$
- Show that $|\det(B)| \leq \|b_1\| \cdots \|b_n\|$ where equality holds if and only if B is orthogonal (Hadamard inequality)

Orthogonality defect

Let $A \in Q^{n \times n}$ be a nonsingular matrix. The number $\gamma \geqslant 1$ with $|\det(A)| \cdot \gamma = \|a_1\| \cdot \cdots \|a_n\|$ is the orthogonality defect of A.

Theorem. *A shortest vector of $\Lambda(A)$ is of the form*

$$\sum_{i=1}^{n} \lambda(i) a_i, \quad \text{where} \quad \lambda(i) \in \mathbb{Z} \quad \text{and} \quad -\gamma \leqslant \lambda(i) \leqslant \gamma.$$

Proof

- Suppose that $|\lambda(n)| > \gamma$.
- Let $B = B^* \cdot R$ be the GSO of B
- Since $\|b_i\| \geqslant \|b_i^*\|$ and $\|b_1\| \cdots \|b_n\| = \gamma \cdot \|b_1^*\| \cdots \|b_n^*\|$ one has $\|b_n\| \leqslant \gamma \cdot \|b_n^*\|$
- $B\lambda = B^* \cdot R\lambda = u + \lambda(n) b_n^*$, where $\langle u, b_n^* \rangle = 0$
- Thus $\|B\lambda\| \geqslant \lambda(n) \|b_n^*\| > \gamma \cdot \|b_n^*\| \geqslant \|b_n\|$ which is a contradiction to $B\lambda$ being shortest vector

Small defect means SV is simple

Consequence

Theorem. *Let $A \in \mathbb{Q}^{n \times n}$ be a rational lattice basis with orthogonality defect γ, then a shortest vector can be computed in time $O((2\gamma+1)^n)$.*

Lattice basis reduction is a way to compute a basis B for $\Lambda(A)$ which has orthogonality defect $\leqslant d(n)$, where $d(n)$ is a number which depends only on the dimension.

Exercise

- Let A be an orthogonal matrix, i.e., columns are orthogonal to each other. Let $w \in \mathbb{Q}^n$. Suppose that $w = \sum_{i=1}^{n} \mu(i) a_i$. Show that the closest vector of $\Lambda(A)$ to w is the vector $\sum_{i=1}^{n} \lfloor \mu(i) \rceil a_i$.

Small defect means CV is simple

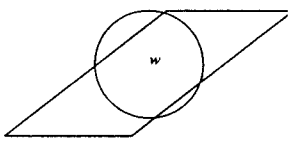
Theorem.

- *Let $A \in \mathbb{Q}^{n \times n}$ be a lattice basis with orthogonality defect γ.*
- *Suppose w.l.o.g. that last column a_n of A has largest norm*
- *Let $w \in \mathbb{Q}^n$ and let $B_{w,\varepsilon}$ be the ball of radius ε around w.*

If $B_{w,\varepsilon} \cap \Lambda(A) = \emptyset$, then $\|a_n^*\| \geqslant \varepsilon/(\gamma \cdot n)$

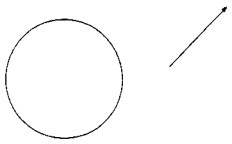
Proof

- $\|a_n^*\| \geqslant \|a_n\|/\gamma$
- Suppose $w = \sum_{i=1}^{n} \mu(i) a_i$ $\mu(i) \in \mathbb{R}$, $1 \leqslant i \leqslant n$
- $B_{w,\varepsilon} \cap \Lambda(A) = \emptyset \implies \sum_{i=1}^{n} (\mu(i) - \lfloor \mu(i) \rfloor) a_i \notin B_{0,\varepsilon}$.
- Since a_n is largest basis vector $\implies \|a_n\| \geqslant \varepsilon/n$
- $\implies \|a_n^*\| \geqslant \varepsilon/(\gamma \cdot n)$



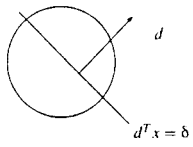
Searching the closest vector

- Let $d = a_n^* / \|a_n\|^2$
- $v \in \Lambda(A) \implies v = A^* \cdot R\lambda$, where $\lambda \in \mathbb{Z}^n$
- $\implies d^T v = d^T A^* \quad R\lambda = (0,\ldots,0,1) R\lambda = \lambda(n) \in \mathbb{Z}$
- $\max\{d^T x \mid x \in B_{\varepsilon,w}\} - \min\{d^T x \mid x \in B_{\varepsilon,w}\} = 2\varepsilon \|d\| \leqslant 2\gamma n$
- Recursively search for lattice vector in $B_{\varepsilon,w} \cap (d^T x = \delta)$, where $\delta \in \mathbb{Z}$ and $\max\{d^T x \mid x \in B_{\varepsilon,w}\} \geqslant \delta \geqslant \min\{d^T x \mid x \in B_{\varepsilon,w}\}$



Searching the closest vector

- Let $d = a_n^* / \|a_n\|^2$
- $v \in \Lambda(A) \implies v = A^* \cdot R\lambda$, where $\lambda \in \mathbb{Z}^n$
- $\implies d^T v = d^T A^* \quad R\lambda = (0,\ldots,0,1) R\lambda = \lambda(n) \in \mathbb{Z}$
- $\max\{d^T x \mid x \in B_{\varepsilon,w}\} - \min\{d^T x \mid x \in B_{\varepsilon,w}\} = 2\varepsilon \|d\| \leqslant 2\gamma n$
- Recursively search for lattice vector in $B_{\varepsilon,w} \cap (d^T x = \delta)$, where $\delta \in \mathbb{Z}$ and $\max\{d^T x \mid x \in B_{\varepsilon,w}\} \geqslant \delta \geqslant \min\{d^T x \mid x \in B_{\varepsilon,w}\}$



Basis reduction makes CV simple

Theorem (LLL Algorithm). *Let $A \in \mathbb{Q}^{n \times n}$ be a lattice basis There exists an algorithm which runs in polynomial time (in binary input encoding) which computes a lattice basis $B \in \mathbb{Q}^{n \times n}$ with*

- $\Lambda(A) = \Lambda(B)$
- *orthogonality defect of B is $\leqslant 2^{n(n-1)/4}$*

If the dimension is fixed, the algorithm runs in linear time.

Solving CV

Theorem (Flatness theorem, Lenstra's algorithm).
Let $A \in \mathbb{Q}^{n \times n}$, $w \in \mathbb{Q}^n$, $\varepsilon \in \mathbb{Q}_{>0}$.
There exists a polynomial algorithm which computes either

- *$v \in \Lambda(A) \cap B_{\varepsilon,w}$ or*
- *$d \in \mathbb{Q}^n$ with $d^T v \in \mathbb{Z}$ for each $v \in \Lambda(A)$ and*

$$\max\{d^T x \mid x \in B_{\varepsilon,w}\} - \min\{d^T x \mid x \in B_{\varepsilon,w}\} \leqslant 2n2^{n(n-1)/4}.$$

If n is fixed the algorithm runs in linear time.

Proof

- $A \longrightarrow LLL \longrightarrow B$, let b_n be largest vector of B
- orthogonality defect $\gamma \leqslant 2^{n(n-1)/4}$
- $w = \sum_{i=1}^n \mu(i) b_i$
- $v = \sum_{i=1}^n \lfloor \mu(i) \rfloor b_i$
- $d = b_n^* / \|b_n^*\|^2$

Solving IP feasibility for ellipsoids

Theorem (Flatness theorem, Lenstra's algorithm).
Let $E(A,a)$ be a rational ellipsoid. There exists a polynomial algorithm which computes either

- *an integer point $x \in E(A,a) \cap \mathbb{Z}^n$*
- *or an integer vector $d \in \mathbb{Z}^n$ with*
  $\max\{d^T x \mid x \in E(A,a)\} - \min\{d^T x \mid x \in E(A,a)\} \leqslant 2n\,2^{n(n-1)/4}$

If the dimension n is fixed, the algorithm runs in linear time.

proof

- Find vector in $\Lambda(A) \cap B_{1,n}$ or
- Find vector f with $f^T A \in \mathbb{Z}^n$ and

$$\max\{f^T x \mid x \in B_{1,n}\} - \min\{f^T x \mid x \in B_{1,n}\} \leqslant 2n2^{n(n-1)/4}.$$

- $f^T x = f^T A A^{-1} x$
- $E(A,a) = A^{-1}B_{1,n}$
- $d^T = f^T A$

Solving IP feasibility for ellipsoids

Theorem. *If the dimension n is fixed, there exists a linear-time algorithm to solve the IP-feasibility problem for ellipsoids.*

- Algorithm above either determines integer point or determines $d \in \mathbb{Z}^n$ with $\max\{d^T x \mid x \in E(A,a)\} - \min\{d^T x \mid x \in E(A,a)\} \leqslant 2n\, 2^{n(n-1)/4}$
- We can assume $\gcd(d) = 1$
- Compute unimodular matrix U with $d^T U = (1,0,\ldots,0)$
- $E(A,a) \cap (d^T x = \delta)$ contains integer point if and only if $E(AU^{-1},a) \cap x(1) = \delta$ contains integer point. (ellipsoid in lower dimension)
- Exercise: Give a closed formula for the ellipsoid above in $n-1$ variables.

The flatness theorem for convex bodies

Max. volume ellipsoids

- Each convex body $K \subseteq \mathbb{R}^n$ has a unique max. volume ellipsoid $E(A,b)$ contained in K.
- $E(A/n,a) \supseteq K \supseteq E(A,a)$

Theorem (Flatness theorem convex bodies). *Let $K \subseteq \mathbb{R}^n$ be a convex body. If $K \cap \mathbb{Z}^n \neq \emptyset$ then there exists a $d \in \mathbb{Z}^n - \{0\}$ such that*

$$\max\{d^T x \mid x \in K\} - \min\{d^T x \mid x \in K\} \leqslant 2n^2 2^{n(n-1)/4}.$$

Better flatness constants

- For Ellipsoids: $O(n)$ [Ban96]
- Simplices: $O(n \log n)$ [BLPS99]

The LLL algorithm

A lower bound on SV_2

Theorem. *Let B be a lattice basis and let $B^* = (b_1^*, \ldots, b_n^*)$ be its Gram-Schmidt orthogonalization, then* $\mathrm{SV}_2(\Lambda(B)) \geqslant \min_{j=1,\ldots,n} \|b_j^*\|_2$.

Proof of theorem

- $0 \neq v \in \Lambda$, then $v = \sum_{j=1,\ldots,k} \lambda(j) b_j$, where $k \leqslant n$, $\lambda(j) \in \mathbb{Z}$, $j = 1, \ldots, k$ and $x_k \neq 0$.
- Using GSO:

$$\begin{aligned} v &= \sum_{j=1,\ldots,k} \left(\lambda(j) \left(b_j^* + \sum_{i=1}^{j-1} \mu_{ij} b_i^* \right) \right) \\ &= \lambda(k) b_k^* + \sum_{i=1}^{k-1} x(i) b_i^*, \text{ for some } x(i) \in \mathbb{R}. \end{aligned}$$

$\|v\| = |\lambda(k)| \|b_k^*\| + \sum_{i=1}^{k-1} |x(i)| \|b_i^*\| \geqslant \|b_k^*\|$.

Summary of insight progress

- If lattice basis is orthogonal, shortest vector is easy.
- The Gram-Schmidt orthogonalization of lattice basis provides lower bound on shortest vector.
- First vector of GSO is first vector of basis.
- Typically vectors in GSO B^* of B are decreasing rapidly, thus spoiling the lower bound.

Natural conclusion

- Given a basis, turn it into something which resembles as much as possible to its GSO.
- Try to assure that the vectors in GSO do not decrease fast, so that first vector is about the size of the minimum in GSO.

The LLL Algorithm

- Normalize: Subtract integer multiples of columns from another column so that $|\mu_{ij}| \leqslant 1/2$ for every $1 \leqslant i < j \leqslant n$ in GSO decomposition

$$B = B^* \begin{pmatrix} 1 & & \mu \\ & \ddots & \\ 0 & & 1 \end{pmatrix}.$$

- Swap (fight the decrease of the $\|b_j^*\|$): If there exists a j such that

$$\|b_{j+1}^* + \mu_{j,j+1} b_j^*\|^2 < 3/4 \|b_j^*\|^2.$$

swap b_j and b_{j+1}. Goto Normalize

Swap: Explanation

- The vector $b_{j+1}^* + \mu_{j,j+1} b_j^*$ is the new j-th vector of B^* after the swap because
  - $b_{j+1}^* = b_{j+1} - \sum_{i=1}^{j} \mu_{i,j+1} b_i^*$.
  - The vector $b_{j+1}^* + \mu_{j,j+1} b_j^*$ is projection of b_{j+1} into orthogonal complement of $b_1, \ldots, b_{j-1}$.
  - The vector $b_{j+1}^* + \mu_{j,j+1} b_j^*$ is new j-th column of B^* after the swap.
  - The j-th column decreases by $3/4$
  - The only possible side effect is an increase of the $j+1$-st column. The rest of the GSO remains unchanged.

A potential function

- $\phi(B) = \|b_1^*\|^{2n} \|b_2^*\|^{2(n-1)} \|b_3^*\|^{2(n-2)} \cdots \|b_n^*\|^2$
- Define $B_j = (b_1, \ldots, b_j)$
- $\|b_1^*\| \cdots \|b_j^*\|$ is j-dimensional volume of parallelepiped of $b_1, \ldots, b_j$
- $\det(B_j^T B_j) = \|b_1^*\|^2 \cdots \|b_j^*\|^2 \in \mathbb{Z}$
- $\phi(B) = \prod_{j=1}^{n} \det(B_j^T B_j) \in \mathbb{Z}$
- A swap of b_j and b_{j+1} of LLL does not change volume for $i = 1, \ldots, j-1, j+1, \ldots, n$, and decreases $\det(B_j^T B_j)$ by a factor of $3/4$.
- Since $\phi(B)$ is an integer (all bases remain integral during LLL) the algorithm terminates in a polynomial number of steps.

Termination of the LLL

We just proved:

Theorem. *Given an integer lattice basis B, the LLL algorithm performs a polynomial number of steps.*

What remains to be done:

- Need to argue that the binary encoding length of numbers involved remains polynomial.

The first vector is short

Let B be a basis returned by LLL:

- $$\begin{aligned} 3/4\,\|b_j^*\|^2 &\leqslant \|b_{j+1} + \mu_{j,j+1} b_j^*\|^2 \\ &\leqslant \|b_{j+1}\|^2 + 1/4\,\|b_j^*\|^2 \end{aligned}$$
- Thus $\|b_j^*\|^2 \leqslant 2\,\|b_{j+1}^*\|^2$ (we successfully fought the rapid decrease!)
- $\|b_1\| = \|b_1^*\| \leqslant 2^{(n-1)/2} \min\{\|b_i^*\| \mid i = 1, \ldots, n\}$
- Thus $\|b_1\| \leqslant 2^{(n-1)/2} \mathrm{SV}(\Lambda(B))$
- Thus SV can be approximated within a factor of in polynomial time

The binary encoding length

What follows is a sketch of polynomiality.

- After normalization:

$$\|b_j\|^2 = \sum_{i=1}^{j} \mu_{ij}^2 \|b_i^*\|^2 \leqslant \sum_{i=1}^{j} \|b_i^*\|^2 \leqslant n \det(\Lambda)$$

- b_j are integral vectors, together with fact above, their encoding length is polynomial in input. (Remember Hadamard bound)!
- GSO is polynomial operation.
- The normalization is polynomial, because it operates on upper-right matrix in GSO decomposition.

The complexity of the LLL

Conclusion:

- The LLL algorithm is polynomial in the bit model of computation.
- If the dimension is fixed, it runs in linear time in binary encoding length (as the Euclidean algorithm)

Orthogonality defect; Exercise

- Show that LLL basis B satisfies

$$\prod_{i=1}^{n} \|b_i\| \leqslant 2^{\binom{n}{2}/2} |\det(B)|.$$

Basis reduction, historical notes

- Lattice basis reduction has its origin in the work of Lagrange on binary quadratic forms.
- With a technical, but algorithmic proof, Gauß [Gau01] showed that a 3-dimensional lattice Λ has a nonzero vector of length $(4/3)^{1/2}\det(\Lambda)^{1/3}$
- Hermite [Her50] generalized this result by showing that each n-dimensional lattice Λ has a nonzero vector v, such that $\|v\|_2 \leq (4/3)^{(n-1)/4}\det(\Lambda)^{1/n}$.

Basis reduction, historical notes

- The LLL algorithm is by Lenstra, Lenstra and Lovász [LLL82].
- A non-algorithmic and beautiful proof of these facts was given by Minkowski [Min68], who opened the stage for a new discipline of mathematics, the geometry of numbers.

Exercise

- Show that, given a lattice basis B, one can in polynomial time compute a nonzero vector $v \in \Lambda(B) - \{0\}$, such that $\|v\| \leq 2^{n-1/4}\sqrt[n]{|\det(B)|}$.

Computing the width of a simplex

The width of a simplex

- Since translation leaves flatness invariant 0 is a vertex
- $\Sigma = \text{conv}\{0, v_1, \ldots, v_n\}$ Simplex.
- A matrix with rows $v_1^T, \ldots, v_n^T$
- width of Σ along c:

$$\|Ac\|_\infty \leq w_c(\Sigma) \leq 2\|Ac\|_\infty$$

Minimal width along integer $c \in \mathbb{Z}^n - \{0\}$ $\approx$ length of shortest vector of $\Lambda(A) = \{Ac \mid c \in \mathbb{Z}^n\}$.

Exercise

- Let Σ be a simplex in fixed dimension. Show that one can determine an integer direction $d \in \mathbb{Z}^n - \{0\}$ with

$$\max\{d^T x \mid x \in \Sigma\} - \min\{d^T x \mid x \in \Sigma\}$$

in linear time.

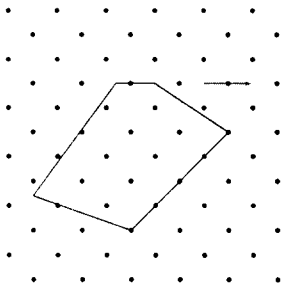
- Hint: Given an LLL-reduced basis in fixed dimension and a constant α, one can enumerate all vectors whose length is at most α times the shortest vector length in constant time.

Solving the optimization problem efficiently

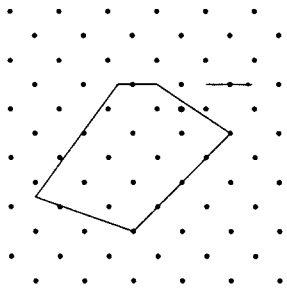
Integer Programming II

- Variables: $x(1), \ldots, x(n)$
- Linear constraints: $a_{i1}x(1) + \cdots + a_{in}x(n) \leqslant b(i)$, for $i = 1, \ldots, m$
- Task: Find integer assignment to $x(1), \ldots, x(n)$ with largest value of $x(1)$.

Integer Programming



Integer Programming



Binary search

- Given a polyhedron in fixed dimension with m constraints, each of binary encoding length s one can solve the integer feasibility problem in time $O(m+s)$
- Via binary search the optimization problem can be solved in time $O((m+s)s)$ [Len83]

Effient algorithms for the plane

History

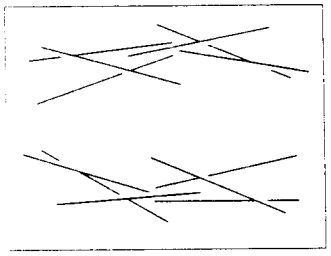
m: Number of constraints
s: largest binary encoding length of coefficient

Method	Complexity
Kannan 1980, Scharf 1981	polynomial
Lenstra 1983	$O(ms+s^2)$
Feit 1984	$O(m\log m+ms)$
Zamanskij and Cherkasskij 1984	$O(m\log m+ms)$
Kanamaru, Nishizeki and Asano 1994	$O(m\log m+s)$
E. and Rote 2000	$O(m+(\log m)s)$
E. 2003	$O(m+(\log m)s)$
E. & Lau	$O(m+s)$
Feasibility test + Euclidean algorithm	$O(m+s)$

any fixed dimension

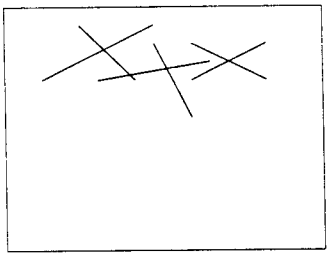
Prune & Search: Dealing with the combinatorics

Megiddo's Algorithm for LP in the plane



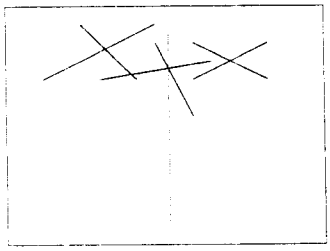
- Partition constraints into "down" and "up" constraints

Megiddo's Algorithm for LP in the plane



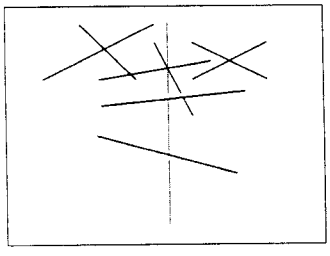
- Partition constraints into "down" and "up" constraints
- Pair "up-constraints" arbitrarily

Megiddo's Algorithm for LP in the plane



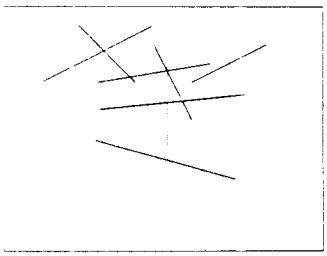
- Partition constraints into "down" and "up" constraints
- Pair "up-constraints" arbitrarily
- Compute median of intersections

Megiddo's Algorithm for LP in the plane



- Partition constraints into "down" and "up" constraints
- Pair "up-constraints" arbitrarily
- Compute median of intersections
- Decide whether optimum is left or right

Megiddo's Algorithm for LP in the plane



- Partition constraints into "down" and "up" constraints
- Pair "up-constraints" arbitrarily
- Compute median of intersections
- Decide whether optimum is left or right
- Prune $1/4$-th of constraints

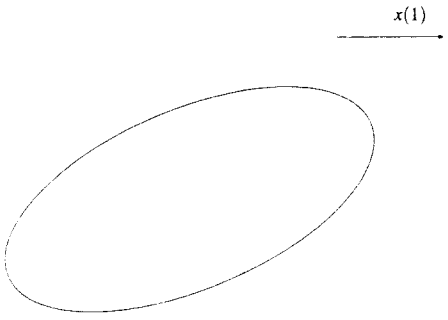
Megiddo's Algorithm for LP in the plane

- Each round at least $1/4$-th of the constraints pruned
- Each round costs linear time
- Overall cost is linear

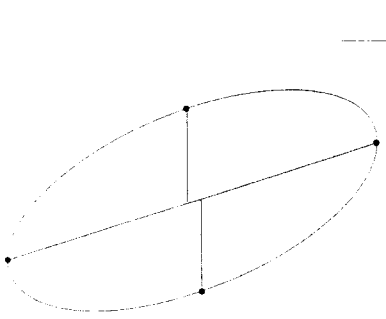
Theorem ([Meg83]). *A linear program in the plane with m constraints can be solved in $O(m)$.*

Combining Prune&Search with feasibility algorithm

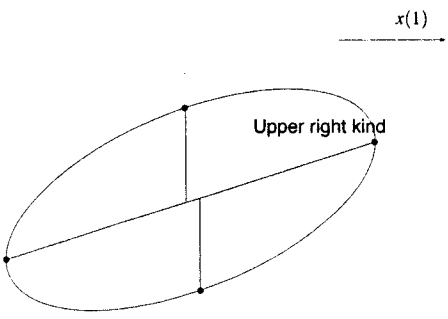
Partitioning the Polygon



Partitioning the Polygon

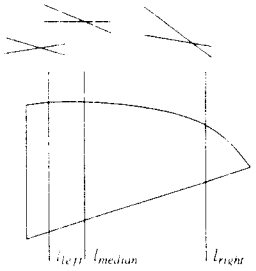


Partitioning the Polygon



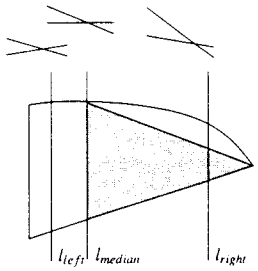
15

Prune & Search



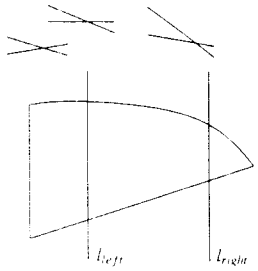
- Principle: Improve l_{left} and l_{right}
- Pair constraints arbitrarily
- Compute median of intersections

Prune & Search



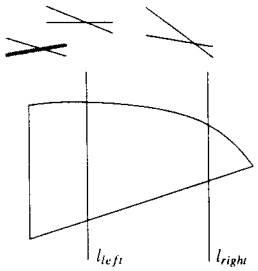
- Principle: Improve l_{left} and l_{right}
- Pair constraints arbitrarily
- Compute median of intersections
- Compute width of triangle defined by median

Prune & Search



- Principle: Improve l_{left} and l_{right}
- Pair constraints arbitrarily
- Compute median of intersections
- Compute width of triangle defined by median
- Update bounds

Prune & Search



- Principle: Improve l_{left} and l_{right}
- Pair constraints arbitrarily
- Compute median of intersections
- Compute width of triangle defined by median
- Update bounds
- Prune $1/4$-th of constraints

Analysis

- Each round $1/4$-th of constraints pruned

$O(m+s)$ is also possible [EL05]

Analysis

- Each round $1/4$-th of constraints pruned
- Computing median is linear

$O(m+s)$ is also possible [EL05]

Analysis

- Each round $1/4$-th of constraints pruned
- Computing median is linear
- Running time without width checking: $O(m)$

$O(m+s)$ is also possible [EL05]

Analysis

- Each round $1/4$-th of constraints pruned
- Computing median is linear
- Running time without width checking: $O(m)$
- Number of checked triangles: $O(\log m)$

$O(m+s)$ is also possible [EL05]

Roadmap

- Show that a problem with m constraints can be solved in expected time $O(m)$ + running time to solve $O(\log m)$ problems with a fixed number of constraints (Clarkson's algorithm)
- In total: Expected $O(m + s \log m)$ algorithm

Efficient algorithms for arbitrary fixed dimension

Integer Programming III

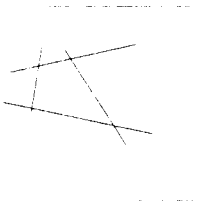
- Variables: $x(1),\ldots,x(n)$
- Set H of rational linear constraints
- Explicit box constraints: $0 \leqslant x \leqslant M$

- Task: Compute $x^*(H)$: Unique integer point which satisfies all constraints in H and the box-constraints which is lexicographically maximal (linear objective)

A theorem of Bell and Scarf

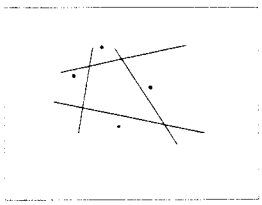
Theorem ([Bel77],[Sca77]). *Let H be a set of rational linear constraints in $\mathbb{R}^n$. If there does not exist an integer point which satisfies all constraints, then there exists a subset $B \subseteq H$ with $|B| \leqslant 2^n$ such that there does not exist an integer point which satisfies all constraints in B.*

Proof



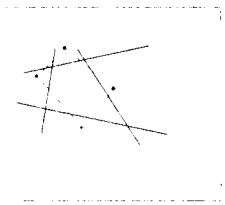
- Let H be minimal such that H has no feasible integer point
- Assume constraints are $a_i^T x \leqslant \beta_i$ $i = 1,\ldots,m$, where a_i and β_i are integers

Proof



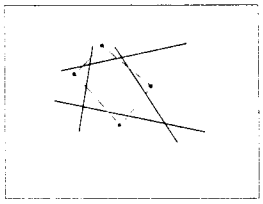
- Let H be minimal such that H has no feasible integer point
- Assume constraints are $a_i^T x \leqslant \beta_i$ $i = 1,\ldots,m$, where a_i and β_i are integers
- For each $a_i^T x \leqslant \beta_i$, there exists an integer solution which satisfies all but the i-th constraint. Let y_i be such an integer solution with $a_i^T y_i$ minimal

Proof



- Let H be minimal such that H has no feasible integer point
- Assume constraints are $a_i^T x \leqslant \beta_i$ $i = 1,\ldots,m$, where a_i and β_i are integers
- For each $a_i^T x \leqslant \beta_i$, there exists an integer solution which satisfies all but the i-th constraint. Let y_i be such an integer solution with $a_i^T y_i$ minimal
- $Z = \mathrm{conv}(\{y_1,\ldots,y_m\} \cap \mathbb{Z}^n)$

Proof



- Let $\gamma_1,\ldots,\gamma_m \in \mathbb{Z}$ s.t. $a_i^T x \leqslant \gamma_i$ has no solution in Z and $\gamma_1 + \cdots + \gamma_m$ is maximal
- For each i there exists a $z_i \in Z$ s.t. $a_i^T z_i = \gamma_i + 1$ and $a_j^T z_i \leqslant \gamma_j$ for each $j \neq i$
- Since $m > 2^n$ there exist $i \neq j$ with $z_i \equiv z_j \pmod 2 \implies 1/2(z_i + z_j) \in Z$ and satisfies all constraints which is a contradiction

Exercise

Prove the following theorem

Theorem ([Sca77]). *Let H be a set of linear constraints. If $x^*(H)$ exists then there exists a subset B of H with $|B| \leqslant 2^n - 1$ with $x^*(H) = x^*(B)$.*

- This B is called a basis of H.
- $D = 2^n - 1$ is combinatorial dimension

Clarkson 1

1. Input: H with $|H| = m$
2. Output: Basis B with $x^*(B) = x^*(H)$
3. $r \leftarrow D\sqrt{m}$, $G \leftarrow \emptyset$
4. `REPEAT`
   (a) Choose random $R \in \binom{H}{r}$
   (b) Compute $x^* = x^*(G \cup R)$ with Clarkson 2
   (c) $V \leftarrow \{h \in H \mid x^* \text{ violates } h\}$
   (d) `IF` $|V| \subseteq 2\sqrt{m}$, `THEN` $G \leftarrow G \cup V$, Augmentation step
5. `UNTIL` $V = \emptyset$
6. `RETURN` x^*

Analysis

Lemma. *In Step (4.c):* $E(|V|) = \sqrt{m}$.

Let B be optimal basis.

- Each augmentation step, a new element of B enters G
- Thus at most d augmentation steps
- $P(|V| > 2\sqrt{m}) \leqslant 1/2$ Markow inequality
- Expected number of draws is $2d$

Clarkson 1 performs:

- Expected $2d$ integer linear programs with Clarkson 2 on $3D\sqrt{m}$ constraints

Sampling Lemma

Lemma. *Let G and H (multi-)sets of constraints $|H| = m$ and let $1 \leqslant r \leqslant m$. Then for random $R \in \binom{H}{r}$:*

$$E(|V_R|) \leqslant d(m-r)/(r+1).$$

where $V_R = \{h \in H \mid x^*(G \cup R)$ violates $h\}$.

This lemma establishes our desired bound because $r = \lceil D\sqrt{m} \rceil$ and thus

$$D(m-r)/(r+1) \leqslant Dm/r \leqslant \sqrt{m}. \qquad (-2)$$

Proof

- See [GW96]
- $E(|V_R|) = \left(\sum_{R \in \binom{H}{r}} |V_R|\right) / \binom{m}{r}$
- $\chi_G(Q,h) = \begin{cases} 1 & \text{if } x^*(G \cup Q) \text{ violates } h, \\ 0 & \text{otherwise.} \end{cases}$
-

$$\begin{aligned} \binom{m}{r} E(|V_R|) &= \sum_{R \in \binom{H}{r}} \sum_{h \in H \setminus R} \chi_G(R,h) \\ &= \sum_{Q \in \binom{H}{r+1}} \sum_{h \in Q} \chi_G(Q-h,h) \\ &\leqslant \sum_{Q \in \binom{H}{r+1}} D \\ &= \binom{m}{r+1} D. \end{aligned}$$

Clarkson 2

- Each $h \in H$ is assigned a multiplicity μ_h.
- In the beginning $\mu_h = 1$ for all $h \in H$.

Clarkson 2

1. `INPUT:` c and H, $|H| = m$
2. `OUTPUT:` $x^*(H)$
3. $r \leftarrow 6d^2$
4. `REPEAT:`
   (a) Choose random $R \in \binom{H}{r}$
   (b) Compute $x^* = x^*(R)$, Base Case
   (c) $V \leftarrow \{h \in H \mid x^*$ violates $h\}$
   (d) `IF` $\mu(V) \leqslant 1/(3d)\mu(H)$ `THEN` for all $h \in V$ do $\mu_h \leftarrow 2\mu_h$
5. `UNTIL` $V = \emptyset$
6. `RETURN` x^*

Lemma. *After kd successful iterations (entering re-weighting step):*

$$2^k \leqslant \mu(B) \leqslant me^{k/3}, \text{ for basis } B \text{ of } H.$$

Lemma. *Clarkson 2 requires $O(d^2 m \log m)$ arithmetic operations and expected $6d \ln m$ base case computations.*

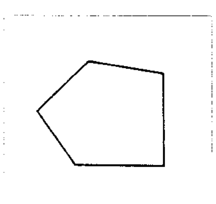
Result of combining 1 and 2

Theorem. *An integer linear program can be solved with expected $O(m)$ arithmetic operations and $O(\log m)$ oracle calls to solve an IP with a fixed number of constraints*

- So far IP with fixed number of constraints $O(s^2)$
- With Clarkson: IP with m constraints costs expected $O(m + s \cdot \log m)$

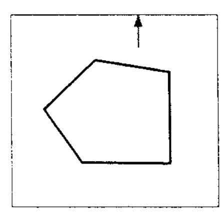
Solving IP with fixed number of constraints in linear time

Lenstra's IP algorithm



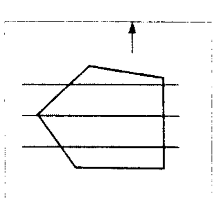
- Lenstra's algorithm is an algorithm for IP feasibility
- Computes width of polyhedron
- If width is to large, then return **feasible**
- Otherwise, **recursively** search for integer point on one of the constant number of hyper-planes (lower dimension)

Lenstra's IP algorithm



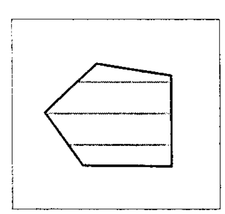
- Lenstra's algorithm is an algorithm for IP feasibility
- Computes width of polyhedron
- If width is to large, then return **feasible**
- Otherwise, **recursively** search for integer point on one of the constant number of hyper-planes (lower dimension)

Lenstra's IP algorithm



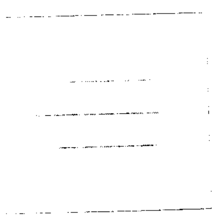
- Lenstra's algorithm is an algorithm for IP feasibility
- Computes width of polyhedron
- If width is to large, then return **feasible**
- Otherwise, **recursively** search for integer point on one of the constant number of hyper-planes (lower dimension)

Lenstra's IP algorithm



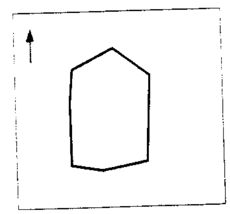
- Lenstra's algorithm is an algorithm for IP feasibility
- Computes width of polyhedron
- If width is to large, then return **feasible**
- Otherwise, **recursively** search for integer point on one of the constant number of hyper-planes (lower dimension)

Lenstra's IP algorithm



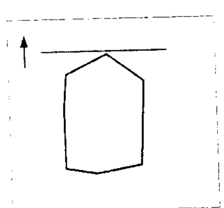
- Lenstra's algorithm is an algorithm for IP feasibility
- Computes width of polyhedron
- If width is to large, then return feasible
- Otherwise, recursively search for integer point on one of the constant number of hyper-planes (lower dimension)

Sliding objective



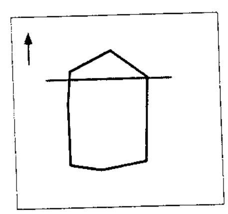
- Let objective function slide into polyhedron
- Until truncation is not flat anymore
- **Optimum** lies on one of a constant number of hyperplanes
- Continue search for optimum in the hyperplanes

Sliding objective



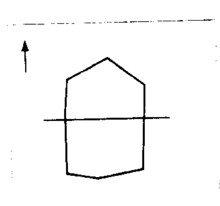
- Let objective function slide into polyhedron
- Until truncation is not flat anymore
- **Optimum** lies on one of a constant number of hyperplanes
- Continue search for optimum in the hyperplanes

Sliding objective



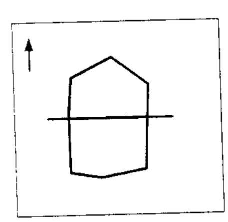
- Let objective function slide into polyhedron
- Until truncation is not flat anymore
- **Optimum** lies on one of a constant number of hyperplanes
- Continue search for optimum in the hyperplanes

Sliding objective



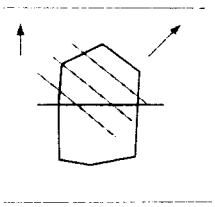
- Let objective function slide into polyhedron
- Until truncation is not flat anymore
- **Optimum** lies on one of a constant number of hyperplanes
- Continue search for optimum in the hyperplanes

Sliding objective



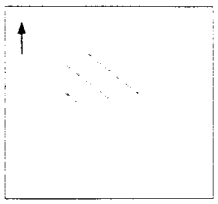
- Let objective function slide into polyhedron
- Until truncation is not flat anymore
- **Optimum** lies on one of a constant number of hyperplanes
- Continue search for optimum in the hyperplanes

Sliding objective



- Let objective function slide into polyhedron
- Until truncation is not flat anymore
- **Optimum** lies on one of a constant number of hyperplanes
- Continue search for optimum in the hyperplanes

Sliding objective



- Let objective function slide into polyhedron
- Until truncation is not flat anymore
- **Optimum** lies on one of a constant number of hyperplanes
- Continue search for optimum in the hyperplanes

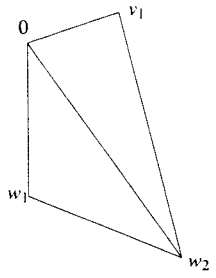
Problem: Geometry of truncation changes too much in the sliding process

Key idea: Restrict to two-layer simplices

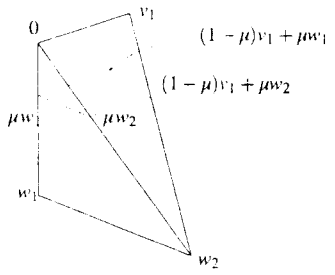
- Σ is two-layer simplex if vertices partition into two sets V and W such that

$$c^T v = c^T v' \text{ and } c^T w = c^T w' \text{ for all } v, v' \in V, w, w' \in W.$$

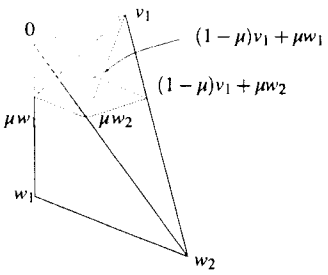
Example in 3D



Example in 3D



Example in 3D



Width of truncation is approximately width of simplex spanned by V and μW.

Width and shortest vector

Width of $\Sigma_{v,w}$ is [illegible] the length of shortest vector in $\Lambda(A_{\mu,k})$, where

- A is matrix with rows $w_1^T, \ldots, w_k^T, v_1^T, \ldots, v_{n-k}^T$
- $A_{\mu,k}$ results from A by scaling first k rows with μ

Parametric shortest vector

The following problem has to be solved:

PARAMETRIC SHORTEST VECTOR:
Given matrix $A \in \mathbb{Z}^{n \times n}$ and parameter $U \in \mathbb{N}$, find parameter $\mu \in \mathbb{N}$ such that $U \leqslant \mathrm{SV}(\Lambda(A_{\mu,k})) \leqslant 2^{n+1/2} \cdot U$

Theorem (E. 2003). *The parametric shortest vector problem can be solved in linear time in fixed dimension.*

Algorithm for PSV

$p \leftarrow 2^{\lceil \log t \rceil}$
$B \leftarrow A_{p,k}$
repeat
  if $p = 1$
    return $\mathrm{SV}(\Lambda) > U$
  $B \leftarrow B_{1/2,k}$
  $p \leftarrow p/2$
  $B \leftarrow \mathrm{LLL}(B)$
until $\|b_1\| \leqslant 2^{(n-1)/2} \cdot U$
return $2p$

Running time

- Potential of basis: $\phi(B) = \|b_1^*\|^{2n} \|b_2^*\|^{2(n-1)} \cdots \|b_1^*\|^2$
- Potential strictly decreases
- $B_1 \rightarrow \mathrm{LLL} \rightarrow B_2$: $\log \phi(B_1) - \log \phi(B_2)$ iterations
- $\phi(A_{U,k}) \leqslant \phi(A_{U,n}) \leqslant U^{2n^2} (\|a_1\| \cdots \|a_n\|)^{2n}$
- In fixed dimension $O(\mathrm{size}(U) + \mathrm{size}(A))$ iterations, linear

Complexity of IP any fixed dimension

Using Clarkson's algorithm for LP-type problems one then obtains:

Theorem (E. 2003). *An integer program with m constraints, each involving coefficients of size at most s can be solved in expected time $O(m + (\log m)s)$.*

What have we learned in these lectures ?

- We know why and how to reduce a basis
- We know how to compute shortest and closest vectors in fixed dimension in linear time
- We have learned about Prune&Search and about Clarkson's random sampling algorithm
- We have seen that IP in fixed dimension can be solved with an almost optimal algorithm

Research problems

- Is there a deterministic $O(m + s \log m)$ algorithm ?
- Is there a $O(m - s)$ algorithm ?

Bibliography

References

[Ban96] W. Banaszczyk. Inequalities for convex bodies and polar reciprocal lattices in $\mathbf{R}^n$. II. Application of K-convexity. *Discrete Comput. Geom.*, 16(3):305–311, 1996.

[Bel77] David E. Bell. A theorem concerning the integer lattice. *Studies in Appl. Math.*, 56(2):187–188, 1976/77.

[BLPS99] Wojciech Banaszczyk, Alexander E. Litvak, Alain Pajor, and Stanislaw J. Szarek. The flatness theorem for nonsymmetric convex bodies via the local theory of Banach spaces. *Mathematics of Operations Research*, 24(3):728–750, 1999.

[EL05] F. Eisenbrand and S. Laue. A linear algorithm for integer programming in the plane. *Mathematical Programming*, 102(2):249 – 259, 2005.

[Gau01] C. F. Gauß. *Disquisitiones arithmeticae*. Gerh. Fleischer Iun., 1801.

[GW96] Bernd Gärtner and Emo Welzl. Linear programming—randomization and abstract frameworks. In *STACS 96 (Grenoble, 1996)*, volume 1046 of *Lecture Notes in Comput. Sci.*, pages 669–687. Springer, Berlin, 1996.

[Her50] Ch. Hermite. Extraits de lettres de M. Ch. Hermite à M. Jacobi sur différents objets de la théorie des nombres. *Journal für die reine und ang[illegible]*