17.1

We need to check how many iterations we need to get the index of all cones below 2:

After n iterations a cone D in our list has index

$$\operatorname{ind} D \leq (\operatorname{ind} C)^{\left(\frac{d-1}{d}\right)^n}$$

$\to$ need to find smallest n s.th. RHS is less than 2

(and thus $\operatorname{ind} D = 1$)

Consider:

$$\lg_2 \lg_2 (\operatorname{ind} C)^{\left(\frac{d-1}{d}\right)^n} = \lg_2 \left(\frac{d-1}{d}\right)^n \lg_2 \operatorname{ind} C$$

$$= n \lg_2 \left(\frac{d-1}{d}\right) + \lg_2 \lg_2 \operatorname{ind} C$$

$$= -n \lg_2 \left(\frac{d}{d-1}\right) + \text{———} \,''\, \text{———} \qquad (*)$$

$$\Rightarrow \text{if} \quad n > \frac{\lg_2 \lg_2 \operatorname{ind} C}{\lg_2 \left(\frac{d}{d-1}\right)} = \mathcal{O}(d \lg_2 \lg_2 \operatorname{ind} C)$$

then the RHS of $(*)$ is negative, so that

$$\lg_2 (\operatorname{ind} C)^{\left(\frac{d-1}{d}\right)^n} < 1$$

$$\Rightarrow (\operatorname{ind} C)^{\left(\frac{d-1}{d}\right)^n} < 2$$

$$\Rightarrow \operatorname{ind} D < 2 \quad \Rightarrow \operatorname{ind} D = 1$$

Hence, we need only a polynomial number of steps for fixed dimension in the size of the input.

For a polynomial time algorithm we also need to control the number of cones in our subdivision

We produce at most

$$(d\,2^d)^n = 2^{n d \lg_2 d} \leq 2^{\left(\frac{\lg_2 \lg_2 \operatorname{ind} C}{\lg_2 \left(\frac{d}{d-1}\right)} + 1\right) d \lg_2 d}$$

$$= 2^{d \lg_2 d} \cdot 2^{\frac{1}{\lg_2 \left(\frac{d}{d-1}\right)} d \lg_2 d \cdot \lg_2 \lg_2 \operatorname{ind} C}$$

$$= 2^{d \lg_2 d} \left(\lg_2 \operatorname{ind} C\right)^{\frac{1}{\lg_2 \left(d-1/d\right)} d \lg_2 d}$$

$$= O\left(\lg_2 \operatorname{ind} C\right)$$

$\to$ need only a polynomial number of cones in decomposition

<u>Thm</u> $d \in \mathbb{Z}_{\geq 1}$ fixed

There is a polynomial time algorithm that computes

$$g_C(x) = \sum_{i \in I} \varepsilon_i \frac{1}{\prod (1 - x^{v_{ij}})} + \left(\begin{array}{l}\text{lower dim}\\ \text{contributions}\end{array}\right)$$

for cones $C_i = \operatorname{cone}(v_{i1}, \ldots, v_{id})$ and $\varepsilon_i \in \{\pm 1\}$

$\to$ we can use this result to compute

(1) h^* for a decomposition of $C(P)$

(2) $|P \cap \mathbb{Z}^d|$ and $e_P(t)$ via the Thm of Brion

This usually requires us to evaluate sums of the form

$$\sum_{i \in I} \varepsilon_i \frac{x^{v_i}}{\prod (1-x^{a_{ij}})} \qquad \text{at } x = \mathbb{1} \text{ or } x = (t, \mathbb{1})$$

However, in most cases this is a pole of the sum.
We know that it must be removable as the Taylor series expansion produces the generating series, which we can evaluate.
Yet, this is <u>not</u> a polynomial time method to achieve this.

Here is one option:

choose a sufficiently generic vector $m = (m_1, \dots, m_d) \in \mathbb{R}^d$ st.

$$\langle m, v_i \rangle \neq 0 \quad , \quad \langle m, a_{ij} \rangle \neq 0$$

[e.g. use the moment curve $m(\lambda) := (1, \lambda, \dots, \lambda^{d-1}) \in (\mathbb{R}^d)^*$

Then $\lambda \longmapsto m(\lambda)(v_i), \alpha_\lambda(a_{ij})$

is a polynomial of degree $d-1$ but not $\equiv 0$

$\Rightarrow$ it has at most $(d-1) d |I|$ zeros, so we can just try sufficiently many different values for λ
This is polynomial in the input size.]

For $r \in \mathbb{C}$ consider

$$x(r) := (e^{rm_1}, \ldots, e^{rm_d})$$

We get our evaluation at $x = \mathbb{1}$ via

$$\lim_{r \to 0} g_C(x(r))$$

Let $\nu_i := \langle m, v_i \rangle$, $\alpha_{ij} := \langle m, a_{ij} \rangle$

Then

$$g_C(x(r)) = \sum_{i \in I} \varepsilon_i \underbrace{\frac{e^{\nu_i r}}{\prod (1 - e^{\alpha_{ij} r})}}_{\text{This is a meromorphic function}}$$

$\to$ want constant term of Laurent expansion at $r = 0$.

Consider a single fraction:

$$\frac{e^{\nu_i r}}{\prod_{j=1}^{k} (1 - e^{\alpha_{ij} r})} = \frac{1}{r^k} e^{\nu_i r} \prod \frac{r}{1 - e^{\alpha_{ij} r}}$$

Each $\frac{r}{1 - e^{\alpha_{ij} r}}$ is an analytic function, and we can compute the Taylor expansion up to degree $k+1$:

$$\frac{r}{1 - e^{\alpha_{ij} r}} = T_{ij}(r) + R_{ij}(r^{k+1})$$

Also:
$$e^{\nu_i r} = S_{ij}(r) + R'_i(r^{k+1})$$

We compute the product up to degree $k+1$:

$$P_i(r) = T_{i1}(r) \cdots T_{ik}(r)\, S_i(r) + R''_i(r^{k+1})$$

and
$$P(r) := \sum P_i(r) + R(r^{k+1})$$

Then the coefficient of r^k is the limit $\lim_{r\to 0} g_C(x(r))$

and thus the evaluation at $x = M$

Remark: Using the Todd-polynomials

$$\prod_{i=1}^{k} \frac{x\xi_i}{1-e^{-x\xi_i}} = \sum_{m=0}^{\infty} \underbrace{td_m(\xi_1, \dots, \xi_k)}_{m\text{-th Todd polynomial}} x^m$$

we can get a closed expression

$$g_C(M) = \sum_{i\in I} \prod \frac{1}{\nu_i} \sum_{k=0}^{d} \frac{\nu_i^k}{k!}\, td_{d-k}(-\alpha_{i1}, \dots, -\alpha_{id})$$

16. The Shortest Vector Problem

(SVP) Λ with basis $a_1, \ldots, a_d$ find, in polynomial time a vector $u \in \Lambda \setminus \{0\}$ of shortest possible length

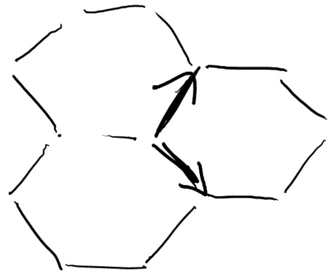
$$\|u\| = \lambda_1(\Lambda)$$

u need not be part of the given basis $a_1, \ldots, a_d$

$\rightarrow$ it will be if the lattice basis is orthogonal:

$$\|u\|^2 = \sum |\lambda_i|^2 \|a_i\|^2 \geq \min \|a_i\|^2$$

However (1) not all lattices have orthogonal bases
(2) we don't know how to find them.

hexagonal lattice: 

If we forget the lattice structure we can efficiently compute an orthogonal basis with the Gram-Schmidt algorithm.

$\rightarrow$ how close can we get?

$\rightarrow$ leads to reduced bases.

One such can be computed with the LLL-algorithm

($\rightarrow$ next section)

17.7

Assume bases are ordered.

Consider

$$V_0 := \{0\}, \quad V_k := \operatorname{lin}(v_1, \dots, v_k), \quad \Lambda_k := \Lambda \cap V_k$$

$$\pi_k : \text{orthogonal projection } \mathbb{R}^d \to V_k$$

The Gram-Schmidt orthogonalization $w_1, \dots, w_d$ is given by

$$w_k := v_k - \pi_{k-1}(v_k) = v_k - \sum_{j=1}^{k-1} \lambda_{jk} w_j \quad \text{for } \lambda_{jk} := \frac{\langle v_k, w_j \rangle}{\|w_j\|^2}$$

with

$$\langle w_i, w_j \rangle = \begin{cases} 1 & i = j \\ 0 & \text{else} \end{cases} \quad \text{and } d(v_k, V_{k-1}) = \|w_k\|$$

Ex: $v_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \quad v_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$

$$\Rightarrow w_1 = v_1, \quad w_2 = -\frac{4}{5} v_1 + v_2 = \begin{pmatrix} -3/5 \\ 6/5 \end{pmatrix}$$

Conversely $\quad v_k = w_k + \sum_{j=1}^{k-1} \lambda_{jk} w_j$

$$\Rightarrow \det \Lambda = \prod_{j=1}^{d} \|w_j\| \qquad \det \Lambda_k = \prod_{j=1}^{k} \|w_j\|$$

Def: The <u>orthogonality defect</u> of Λ is

$$\Pi := \frac{1}{\det \Lambda} \prod_{j=1}^{d} \| v_j \|$$

Note $v_1, \dots, v_d$ orthogonal $\iff \Pi = 1$

Def: $v_1, \dots, v_d$ is a <u>reduced basis</u> if the orthogonality defect is bounded by a constant Π_d depending only on the dimension d.