Quantum Lattice Enumeration and Tweaking Discrete Pruning

Yoshinori Aono *Phong Q. Nguyen* *Yixin Shen*

Context

- *NIST standardization of post-quantum cryptography:*
  - *Need to convince security estimates for lattice-based cryptosystems (especially in the quantum setting)*
  - *Typical attacks rely on a lattice reduction algorithm (BKZ)* $\longrightarrow$ *uses SVP as a subroutine*
- *Main approaches to solve SVP*
  - *Sieving:* $2^{O(n)}$ *time and* $2^{O(n)}$ *space*
    - *Best known classical heuristic time* $2^{0.292n+o(n)}$
    - *Best known quantum heuristic time* $2^{0.265n+o(n)}$
  - *Enumeration:* $2^{O(n\log(n))}$ *time and poly(n) space*
    - *Speed-up in quantum setting?*

SVP: the Shortest Vector Problem

n: dim of the lattice

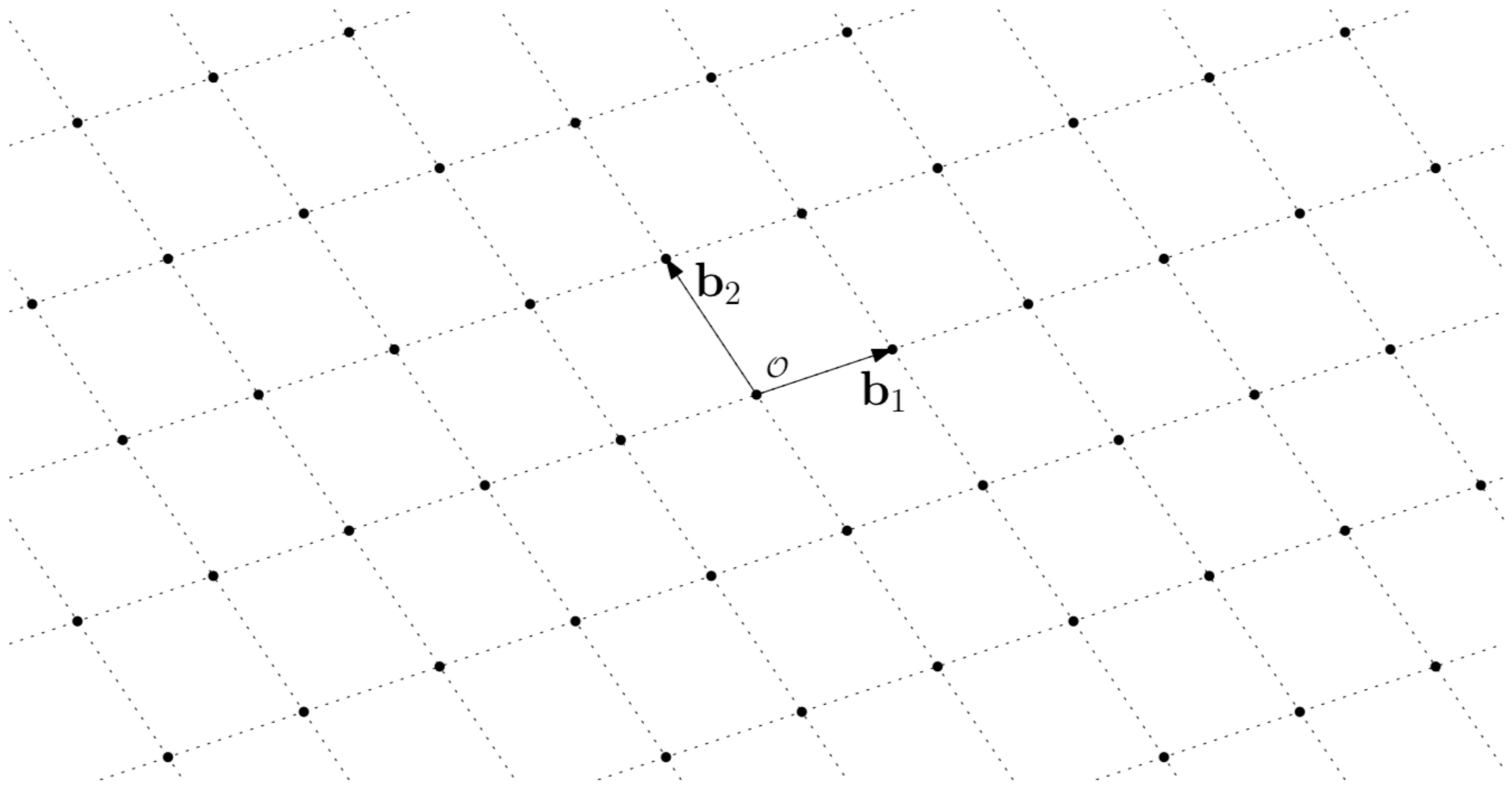
Contribution

- *Quasi-quadratic quantum speed-up for cylinder pruning and discrete pruning*
- *Optimizing discrete pruning preprocessing (open problem in [AN17])*

What is a lattice?

$$L(b_1, \cdots, b_n) = \{ \sum_{i=1}^{n} x_i b_i \,|\, x_i \in \mathbb{Z}, \forall 1 \leq i \leq n \}$$

where $(b_1, \cdots, b_n)$ *is a basis of* $\mathbb{R}^n$



Enumeration Algorithm

$S(R)$: a centered n-dimension ball of radius R

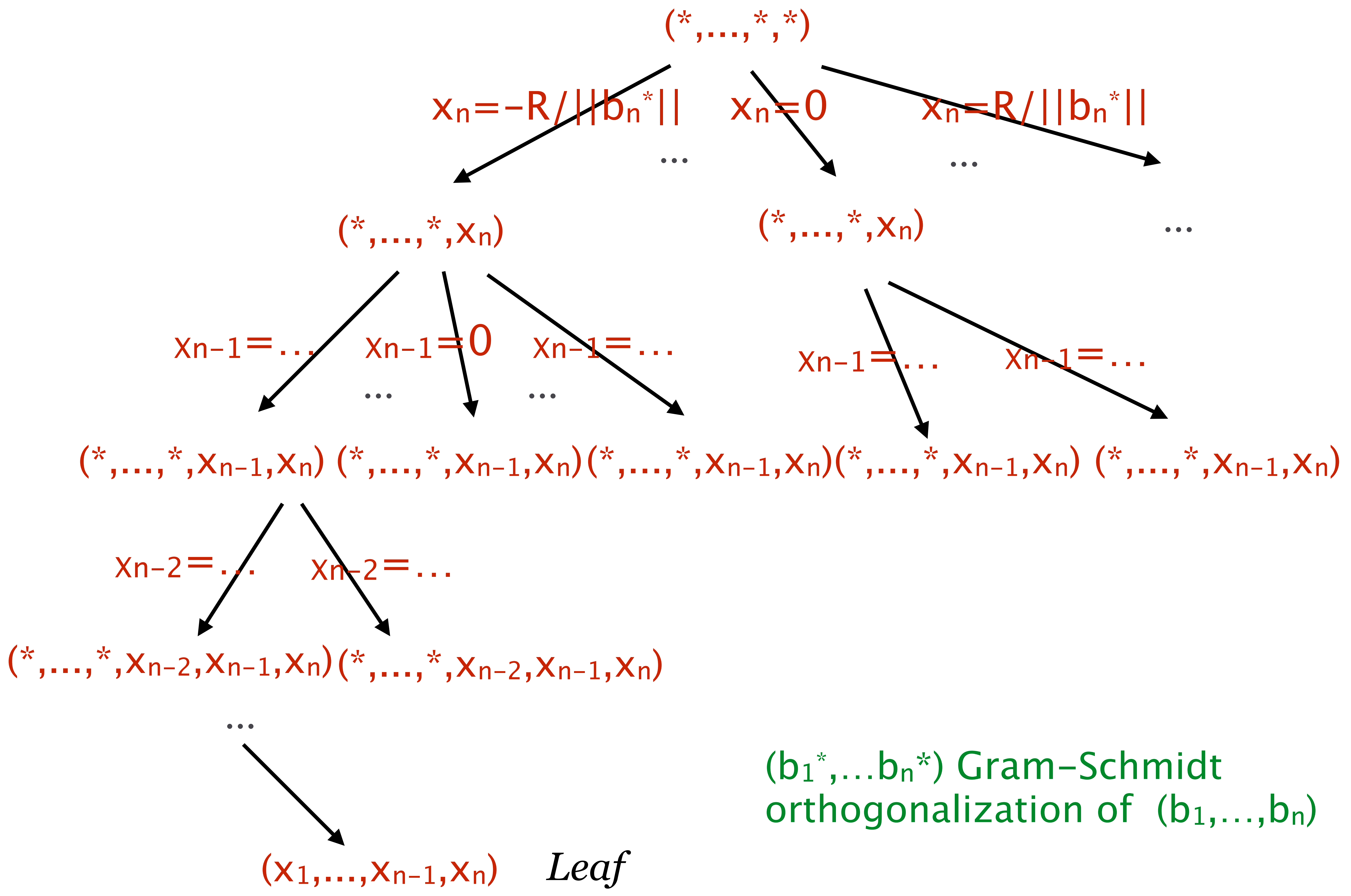
Search for all vectors $x = x_1 b_1 + x_2 b_2 \ldots + x_n b_n$ in $S(R)$

π_i :the orthogonal projection on

$\mathtt{span}(b_1, \cdots, b_{i-1})^{\perp}$

Given $x_n, \cdots, x_{i+1}$, $\|\pi_i(x)\| \leq R$

$\Longrightarrow$ *the integer x_i belongs to an interval of small length.*



$(b_1^*, \ldots b_n^*)$ Gram–Schmidt orthogonalization of $(b_1, \ldots, b_n)$

Quantum Speed-up for Enumeration

Implicit in [Alkim et al 2016] [Alkim et al 2017] [del Pino et al 2016]

Quantum backtracking *[Montanaro 2015]:*

- *A tree of size T, of depth n, of constant max degree, with marked nodes*
- *A blackbox which specifies the local structure of the tree*

$\Rightarrow$ $O^*(\sqrt{T})$ *queries for finding a marked node*

Application to the previous enumeration algorithm: (**Quantum Lattice Enumeration**)

Difficulties: If the basis is only LLL-reduced, max degree can be $2^{O(n)}$

Idea: Transform the tree into a binary one

$\Rightarrow$ $O^*(\sqrt{T})$ *time for finding one vector in* $L \cap S(R)$

$\Rightarrow$ $O^*(\#(L \cap S(R))\sqrt{T})$ *time for finding all vectors in* $L \cap S(R)$

Enumeration with Pruning [ScEu94, ScHo95, GNR10]

Enumeration with Pruning [ScEu94, ScHo95, GNR10]

Previous Enumeration algorithm:

- *Running-time depends on the quality of the basis*
- *Running-time typically superexponential, much larger than $\#(L \cap S(R))$.*

Enumeration with Pruning
[ScEu94, ScHo95, GNR10]

Previous Enumeration algorithm:

- *Running-time depends on the quality of the basis*
- *Running-time typically superexponential, much larger than #(L∩S(R)).*

Enumeration with Pruning:

$P \subseteq \mathbb{R}^n$ *a pruning set*

Search only the vectors in L∩S(R)∩P

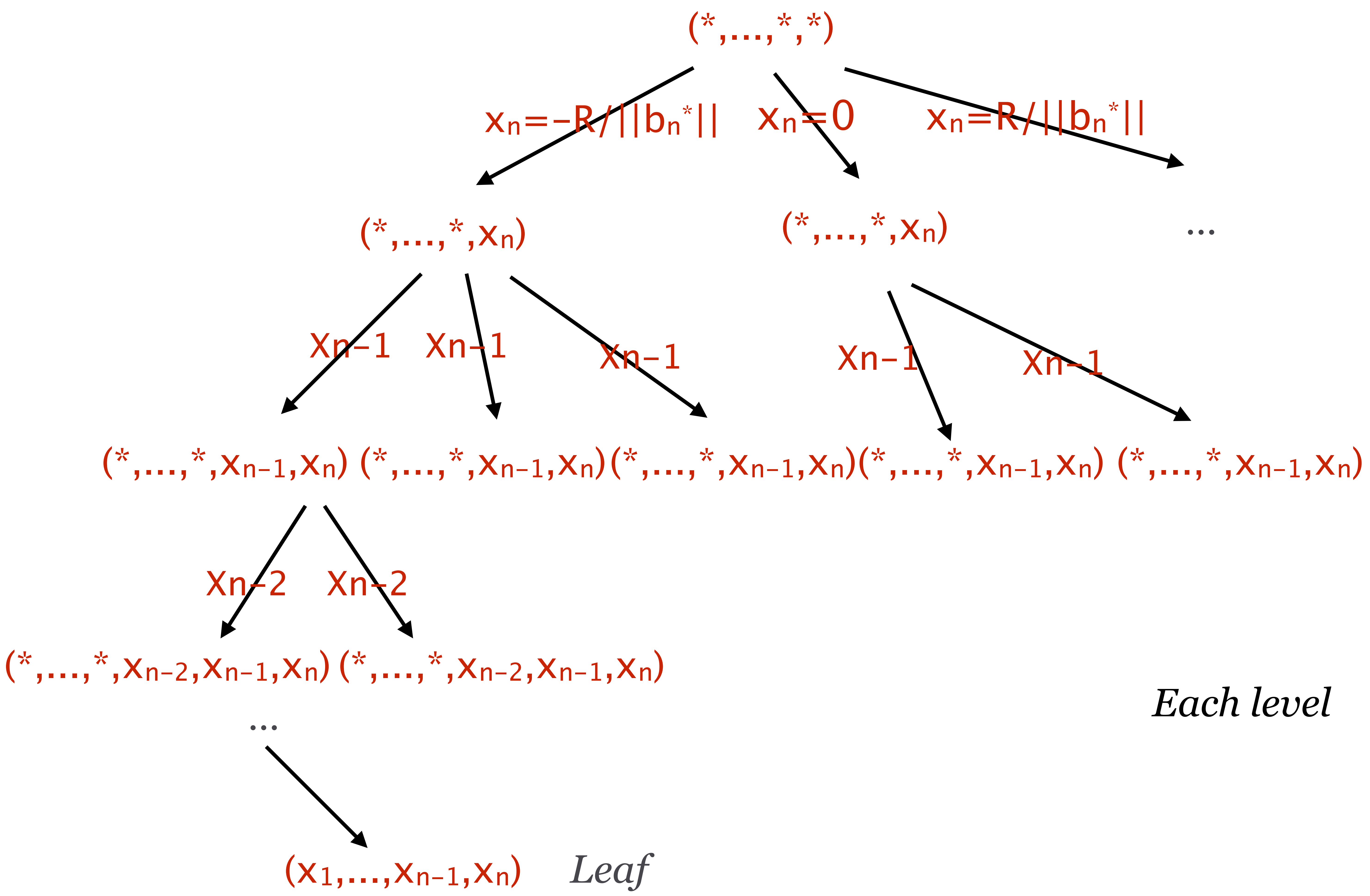
- *Pros: Enumerating Tree L∩S(R)∩P*
  - *can be much smaller than the one of L∩S(R)*
- *Cons: Maybe L∩S(R)∩P= ∅*

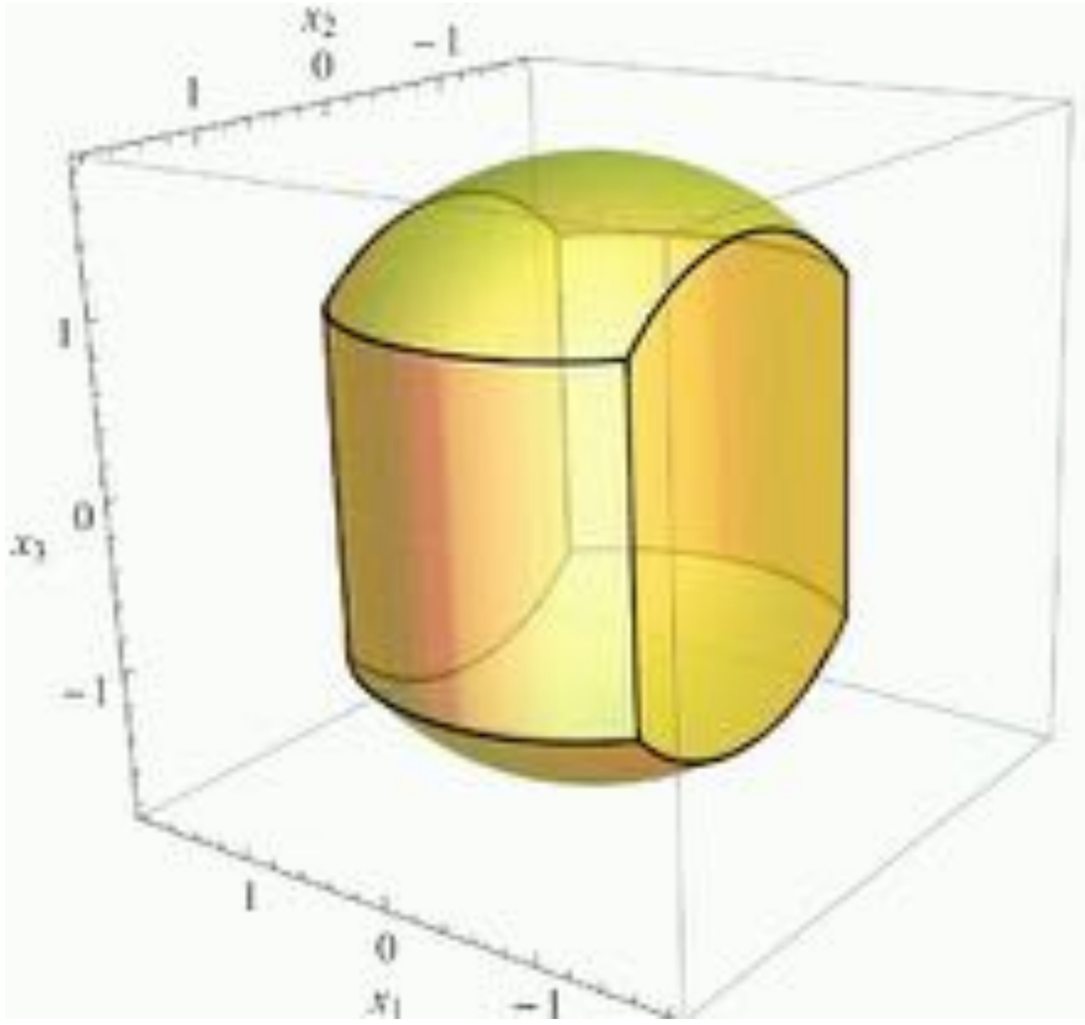
Extreme Pruning [GNR10]

- *Repeat until a vector is found*
  - *Generate a « random » basis and a pruning set P based on it*
  - ***Enumeration**(L∩S(R)∩P)*
- *Even if Pr(L∩S(R)∩P≠∅) is tiny, what matters is the trade-off:*

*Cost(**Enumeration**(L∩S(R)∩P))/Pr(L∩S(R)∩P≠∅)*

Cylinder Pruning
[ScEu94, ScHo95, GNR10]

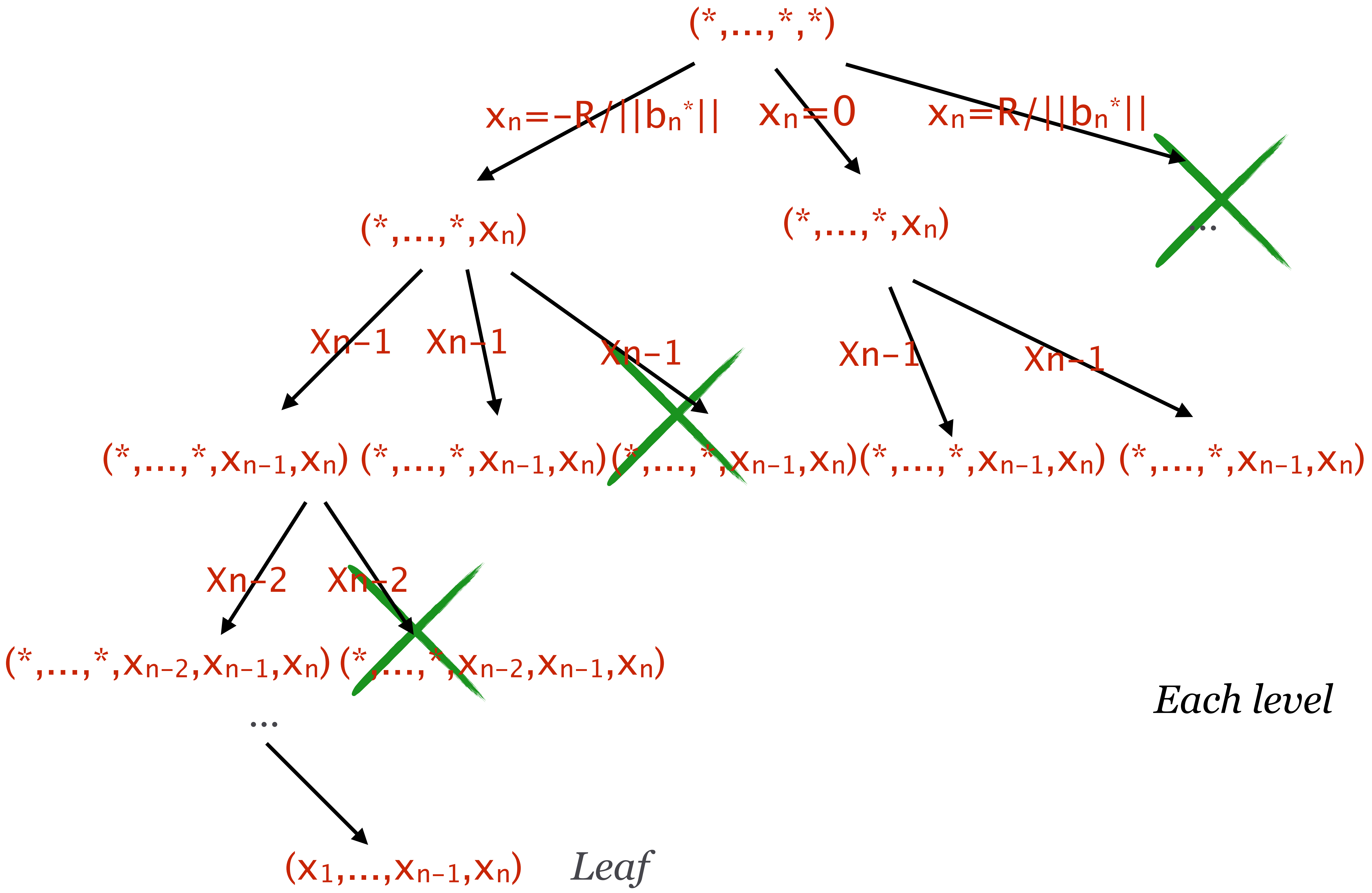


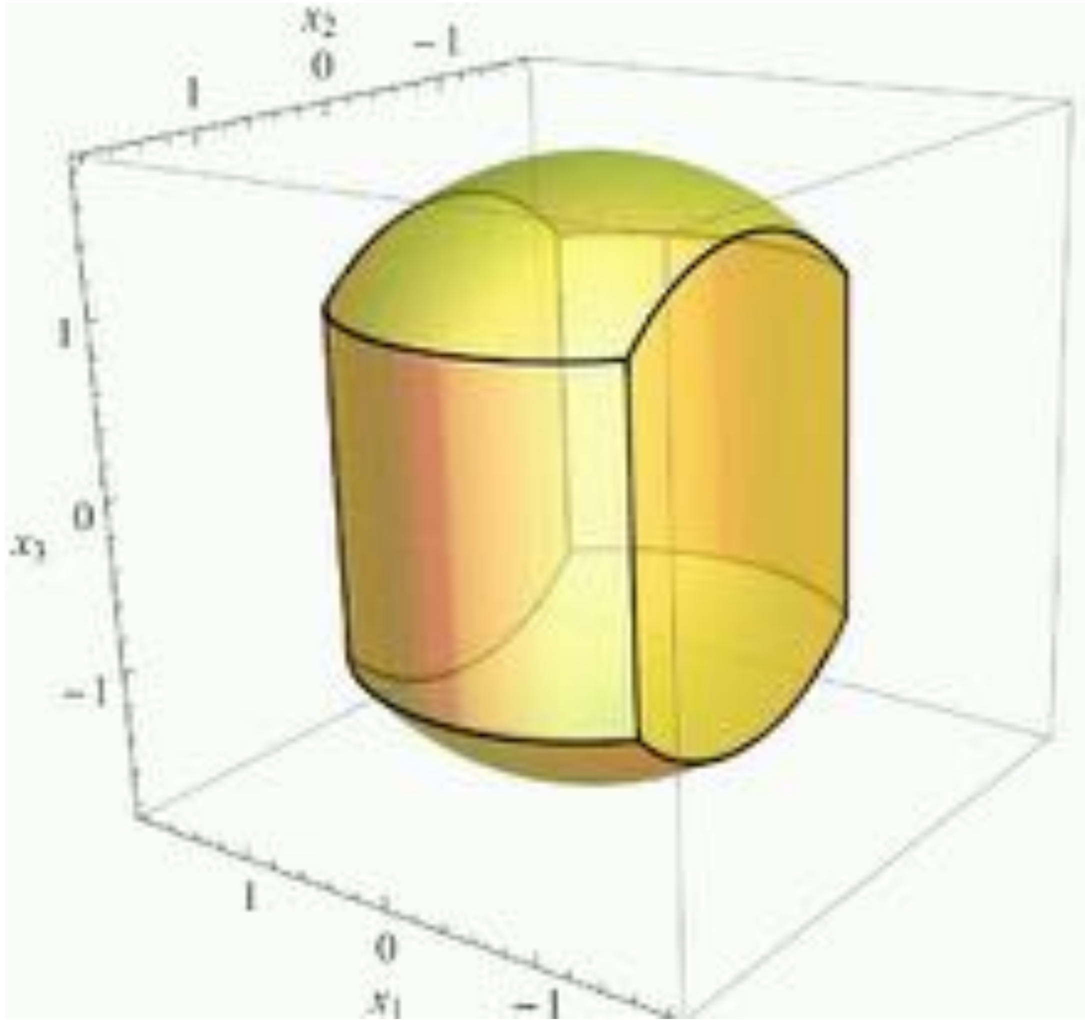


Each level $\|\pi_i(x)\| \leq R \longrightarrow \|\pi_i(x)\| \leq R_i R$

where $0 < R_i \leq 1$

Cylinder Pruning
[ScEu94, ScHo95, GNR10]





Each level $\|\pi_i(x)\| \leq R \longrightarrow \|\pi_i(x)\| \leq R_i R$

where $0 < R_i \leq 1$

Quantum Speed-up for Cylinder Pruning

In practice, L is an integer lattice. The basis is LLL-reduced $\longrightarrow$ $R = \|b_1\| \leq 2^{\frac{n-1}{2}} \lambda_1(L)$

Quantum Lattice Enumeration on the truncated tree:

$\rightarrow O^*(\sqrt{T})$ *time for finding one vector L∩S(R)∩P, if it's not empty*

+ dichotomy on R

$\rightarrow O^*(\sqrt{T})$ *time for finding the shortest vector in L∩S(R)∩P, if it's not empty*

Quantum Speed-up for Cylinder Pruning

In practice, L is an integer lattice. The basis is LLL-reduced $\longrightarrow$ $R = \|b_1\| \leq 2^{\frac{n-1}{2}} \lambda_1(L)$

Quantum Lattice Enumeration on the truncated tree:

$\rightarrow O^*(\sqrt{T})$ *time for finding one vector L∩S(R)∩P, if it's not empty*

+ dichotomy on R

$\rightarrow O^*(\sqrt{T})$ *time for finding the shortest vector in L∩S(R)∩P, if it's not empty*

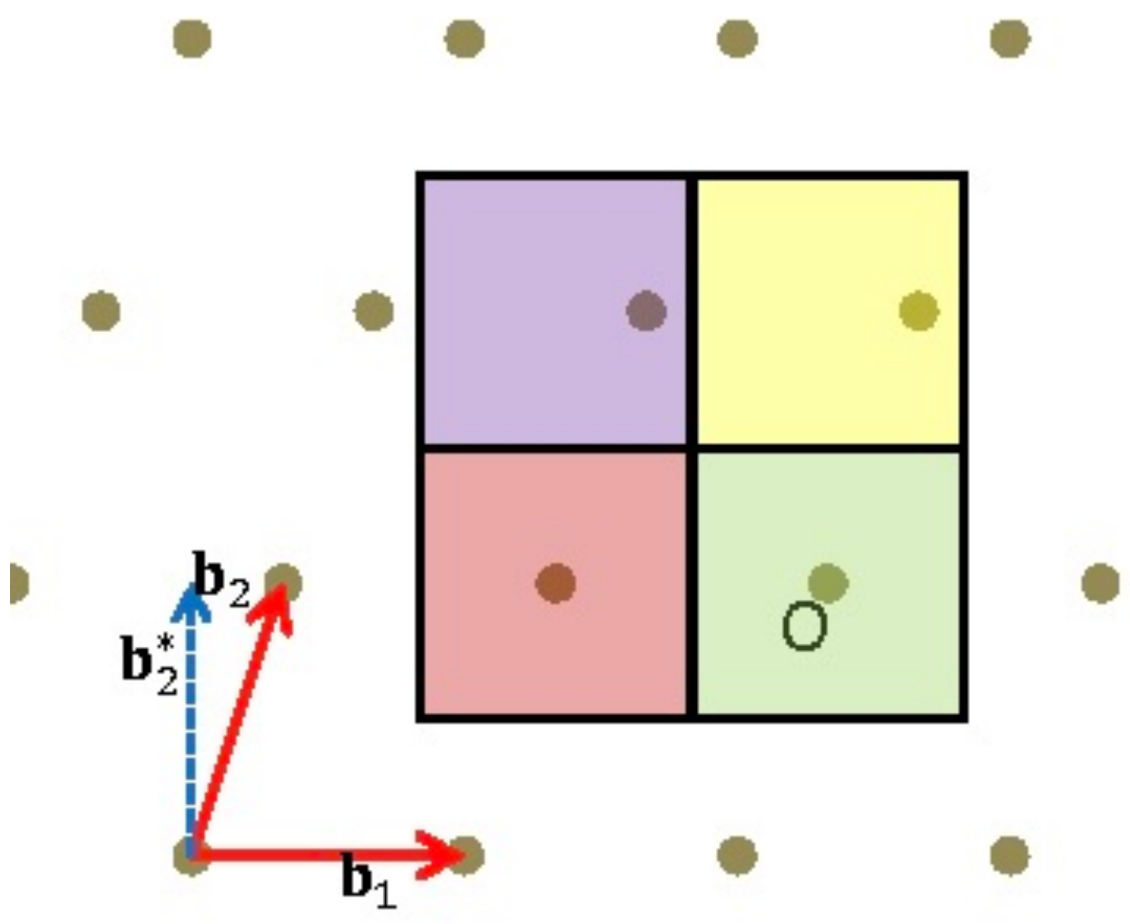
***Extreme Cylinder Pruning**: Given m LLL-reduced bases of the same lattice,* $T_1,\ldots,T_m$ *the corresponding enumeration tree sizes,* $O^*(\sqrt{\sum_{i=1}^{m} T_i})$ *time for finding the shortest vector among all the pruning sets.*

Discrete Pruning
[AN 2017]

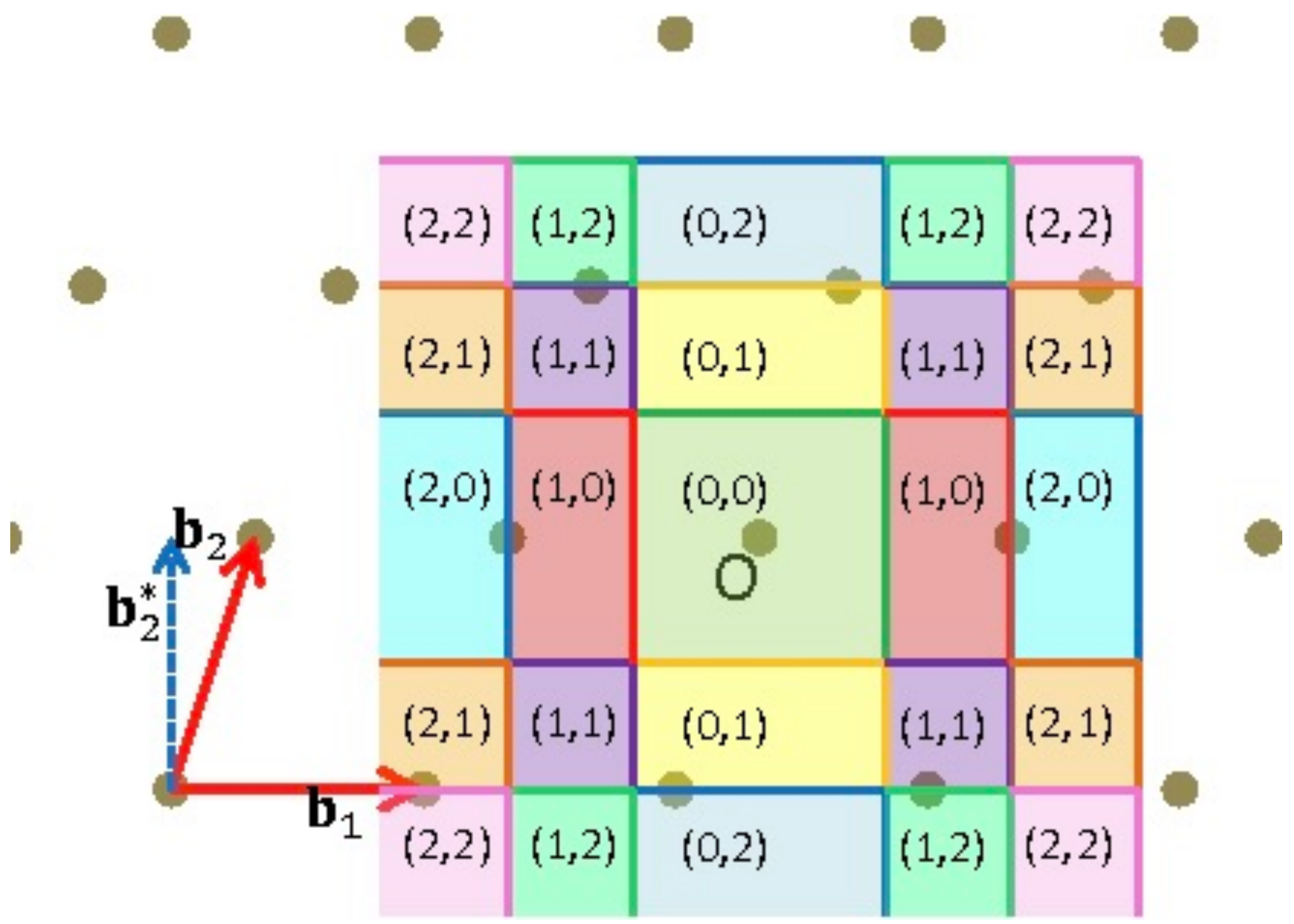
Lattice partition: $\mathbb{R}^n = \cup_{t=(t_1,\cdots,t_n)\in\mathbb{Z}^n} C(t)$

1 cell<-> 1 lattice vector

Two examples:



Babai's partition



The natural partition

The pruning set: $P = \cup_{t\in U} C_{\mathbb{N}}(t), U \subset \mathbb{Z}^n, |U| = \mathtt{poly}(n)\cdot M$

Discrete Pruning
[AN17]

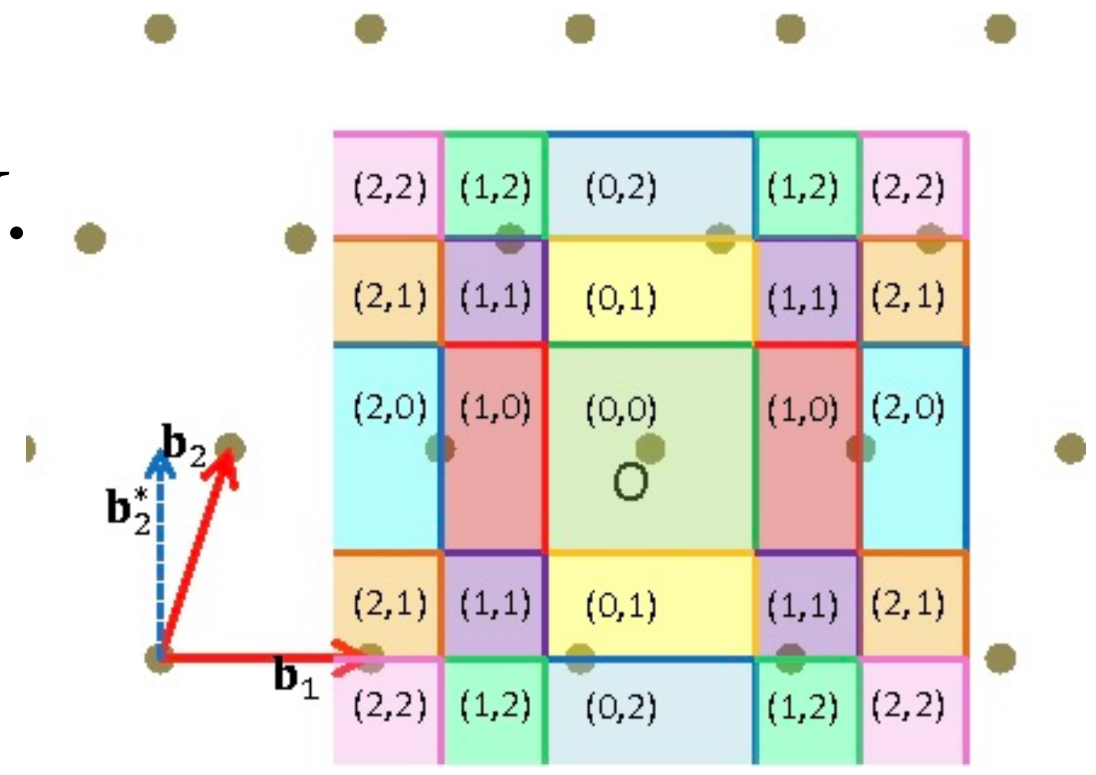
Step 1: *Find the pruning set*

- *Find approximatively M best cells minimizing* $\sum_{i=1}^{n} f(t_i)\|b_i^*\|^2$ *where* $f(t_i) = \frac{t_i^2}{4} + \frac{t_i}{4} + \frac{1}{12}$

  Roughly, the smaller $\sum_{i=1}^{n} f(t_i)\|b_i^*\|^2$, *the shorter the vector x inside* $C_{\mathbb{N}}(t)$
- *Equivalent to find R such that #Solutions of* $\sum_{i=1}^{n} f(t_i)\|b_i^*\|^2 \leq R$ *is close to M.*

Step 2: *Find the shortest vector among these cells*

- *Step 2 can also be seen as a depth-first search of a tree.*



1 lattice vector <-> 1 cell

Quantum Speed-up for Discrete Pruning

Step 1: Find R such that #Sol of $\sum_{i=1}^{n} f(t_i)\|b_i^*\|^2 \le R$ *is close to M (up to poly(n) factor).*

- ***TreeSizeEstimation*** *[Ambainis and Kokainis 2017]:*
  - *A blackbox which specifies the local structure of the tree*
  - *An estimation T of #nodes, δ: precision parameter*

  $\rightarrow O^*(\sqrt{T})$ *queries to give an estimate of #nodes within δ precision when T≤#nodes, or output T>#nodes*
- *Additional tweak:* $\sum_{i=1}^{n} f(t_i)\|b_i^*\| = \sum_{i=1}^{n} \left(\frac{t_i^2}{4} + \frac{t_i}{4} + \frac{1}{12}\right)\|b_i^*\|^2 \rightarrow C\sum_{i=1}^{n} (t_i^2 + t_i)\,\|b_i^*\|^2$
  - *Consequence: linear relation between #nodes and #leaves*

By dichotomy, we can find R such that $M \le \#Sol \le 32n^2M$ *in* $O^*(\sqrt{M})$ *time.*

Quantum Speed-up for Discrete Pruning

Step 2: Find the shortest vector among the cells corresponding to leaves satisfying $C\sum_{i=1}^{n}(t_i^2+t_i)\,\|\overrightarrow{b}_i^*\|^2 \le R$

- *Same as before:* ***Quantum backtracking*** *+ binary tree transformation + dichotomy*

Step 1+ Step 2 $\longrightarrow$ *In total,* $O^*(\sqrt{M})$ *time to find a shortest non-zero vector in* $L \cap P$

Quantum Speed-up for Discrete Pruning

Step 2: Find the shortest vector among the cells corresponding to leaves satisfying $C\sum_{i=1}^{n}(t_i^2+t_i)\,\|\overrightarrow{b}_i^*\|^2 \le R$

- *Same as before:* ***Quantum backtracking*** *+ binary tree transformation + dichotomy*

Step 1+ Step 2 $\longrightarrow$ *In total,* $O^*(\sqrt{M})$ *time to find a shortest non-zero vector in* $L \cap P$

Extreme Discrete Pruning*: Given m LLL-reduced bases of the same lattice, we can find a R such that the total number of cells such that at least one* $C\sum_{i=1}^{n}(t_i^2+t_i)\,\|\overrightarrow{b}_i^*\|^2 \le R$ *is satisfied is close to M, then find the shortest non-zero vector inside these cells.*

$\rightarrow$ $O^*(\sqrt{M})$ *times in total*

Our results

In this talk:

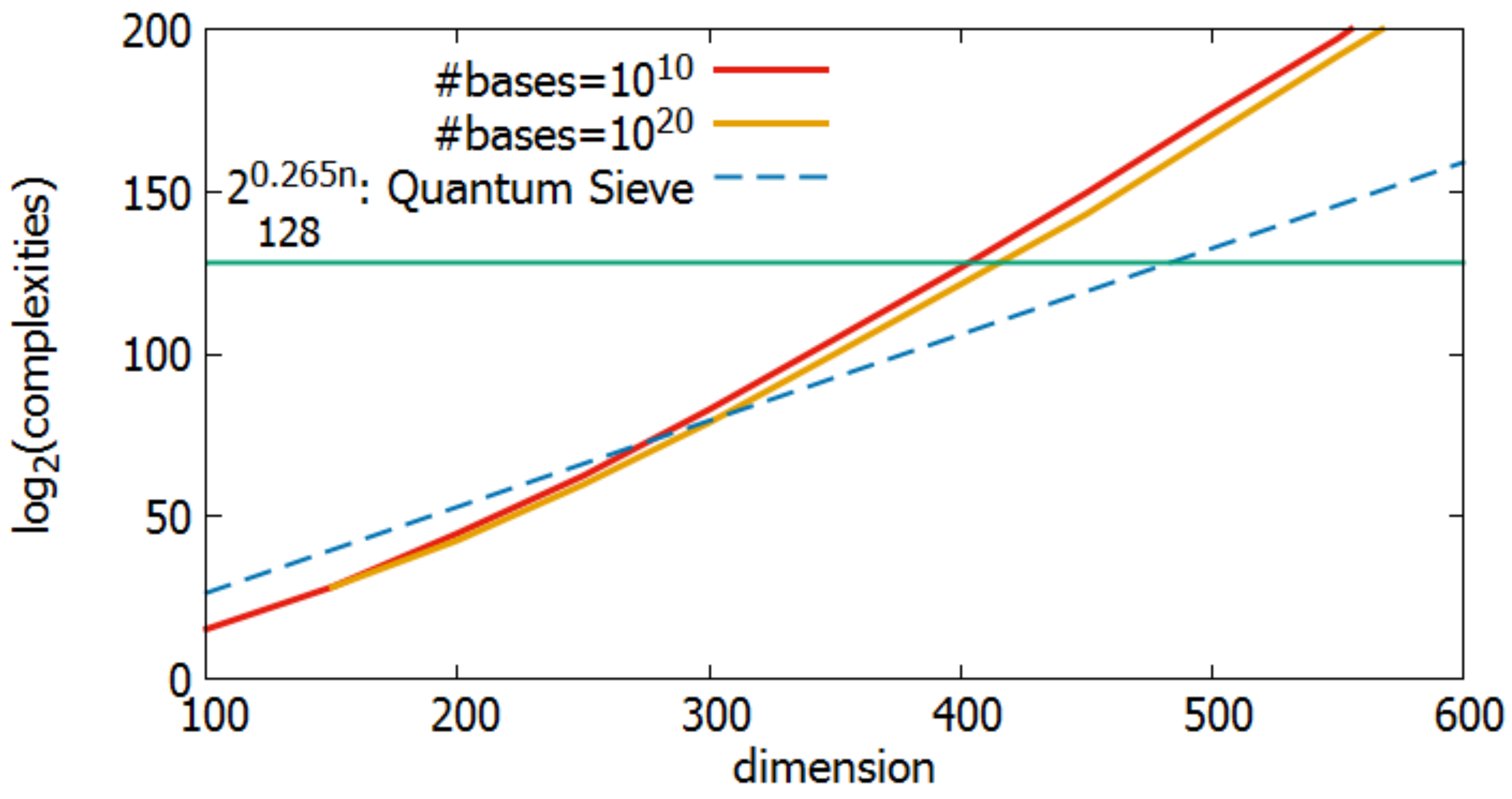
- *Quasi-quadratic speed-up for both cylinder and discrete pruning for SVP (for integer lattice)*
- *Speed-up applicable in the extreme pruning setting*

In the paper:

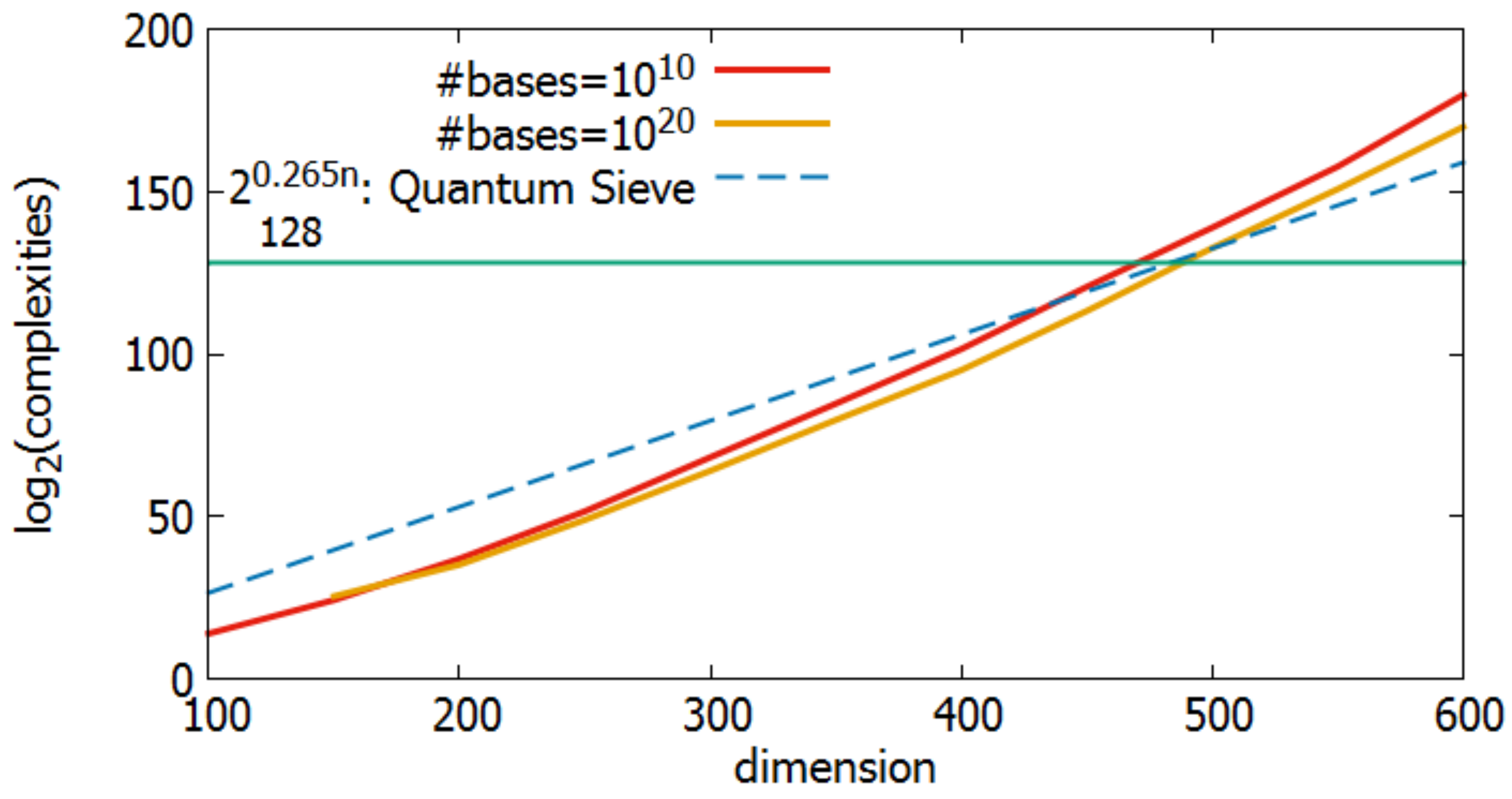
- *Quasi-quadratic speed-up for cylinder pruning for CVP (same as for SVP)*
- *Tweak which adapts discrete pruning to CVP*

$\longrightarrow$ *Quasi-quadratic speed-up for discrete pruning for CVP when the target has integer coordinates*

Revisiting Q-sieve vs Q-enum



quasi-HKZ bases



Rankin bases

Complexity: $\sqrt{\#bases * N}$, N: upper bound of the number of nodes of enumeration with extreme pruning with probability 1/#bases [ANSS18]

Quantum enumeration with extreme pruning would be faster than quantum sieve up to higher dimensions than previously thought!

Our results affect the security estimates of between 11 and 17 NIST submissions.

Thank you for your attention!