Cornell Bowers C·IS
Computer Science

Theory Seminar
Sep 25, 2023

Recursive Lattice Reduction

Spencer Peters
and

Divesh Aggarwal Thomas Espitau Noah S.D.

Lattices

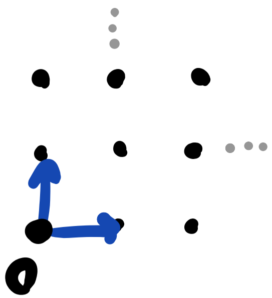
A *lattice* $\mathcal{L} = \mathcal{L}(B)$ is specified by a *basis*
$B = (b_1, b_2, \ldots, b_n)$ of linearly independent vectors $b_i \in \mathbb{R}^d$:

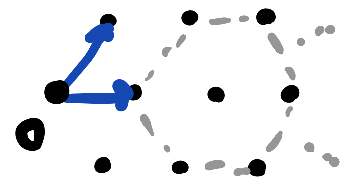
$$\boxed{\mathcal{L}(B) := \{ z_1 b_1 + z_2 b_2 + \cdots + z_n b_n \mid z_i \in \mathbb{Z} \}}$$

Lattices

A *lattice* $\mathcal{L} = \mathcal{L}(B)$ is specified by a *basis* $B = (b_1, b_2, \dots, b_n)$ of linearly independent vectors $b_i \in \mathbb{R}^d$:

$$\boxed{\mathcal{L}(B) := \{ z_1 b_1 + z_2 b_2 + \dots + z_n b_n \mid z_i \in \mathbb{Z} \}}$$

e.g., $\mathbb{Z}^2$: 

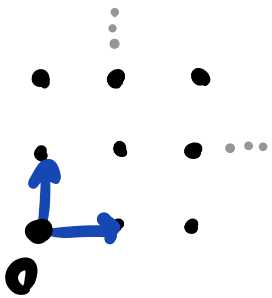
, $\mathbb{Z}^n$: 
, Hexagonal: 

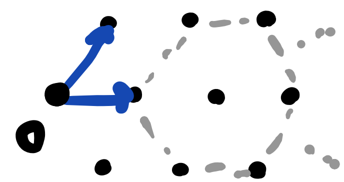
, ...

Lattices

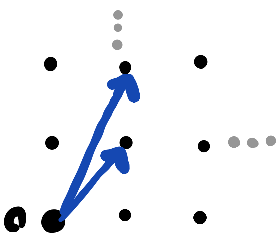
A *lattice* $\mathcal{L} = \mathcal{L}(B)$ is specified by a *basis* $B = (b_1, b_2, \dots, b_n)$ of linearly independent vectors $b_i \in \mathbb{R}^d$:

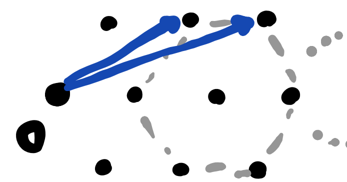
$$\boxed{\mathcal{L}(B) := \{ z_1 b_1 + z_2 b_2 + \dots + z_n b_n \mid z_i \in \mathbb{Z} \}}$$

e.g., $\mathbb{Z}^2$: 

, $\mathbb{Z}^n$: 
, Hexagonal: 

, ...

$\mathbb{Z}^2$: 

, $\mathbb{Z}^n$: 
, Hexagonal: 

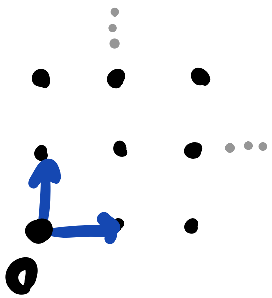
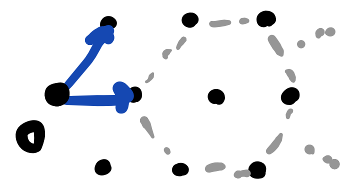
, ...

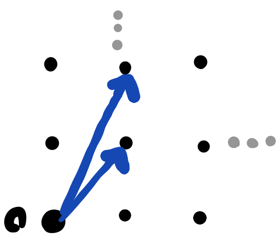
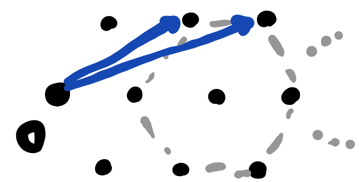
Lattices

A *lattice* $\mathcal{L} = \mathcal{L}(B)$ is specified by a *basis*

$B = (b_1, b_2, \dots, b_n)$ of linearly independent vectors $b_i \in \mathbb{R}^d$:

$$\boxed{\mathcal{L}(B) := \{ z_1 b_1 + z_2 b_2 + \cdots + z_n b_n \mid z_i \in \mathbb{Z} \}}$$

e.g., $\mathbb{Z}^2$:  , $\mathbb{Z}^n$:  , Hexagonal:  , ...

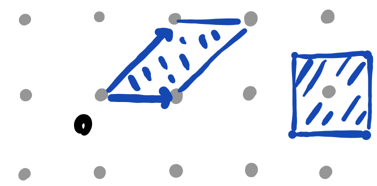
$\mathbb{Z}^2$:  , $\mathbb{Z}^n$:  , Hexagonal:  , ...

n is called the *rank* or *dimension* of $\mathcal{L}$.

Lattices

$$\boxed{\lambda_1(\mathcal{L}) := \min_{\substack{y \in \mathcal{L} \\ y \neq \vec{0}}} \|y\|}$$

Lattices

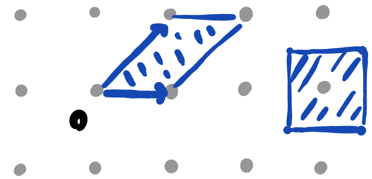


$$\lambda_1(\mathcal{L}) := \min_{\substack{y \in \mathcal{L} \\ y \neq \vec{0}}} \|y\|$$

$$\det(\mathcal{L}) := \sqrt{\det(B^T B)}$$

$$= \frac{\text{volume}}{\text{point}} = \frac{1}{\text{density}}.$$

Lattices



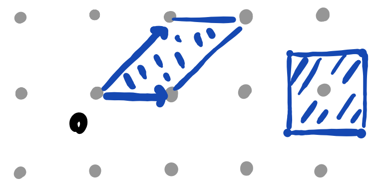
$$\lambda_1(\mathcal{L}) := \min_{\substack{y \in \mathcal{L} \\ y \neq \vec{0}}} \|y\|$$

$$\det(\mathcal{L}) := \sqrt{\det(B^T B)}$$

$$= \frac{\text{volume}}{\text{point}} = \frac{1}{\text{density}}.$$

$$\lambda_1(\mathcal{L}) \leq \sqrt{n}\, \det(\mathcal{L})^{1/n}$$

Lattices



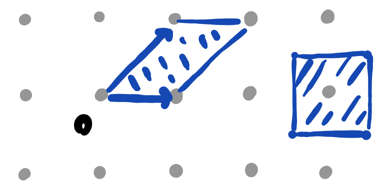
$$\lambda_1(\mathcal{L}) := \min_{\substack{y \in \mathcal{L} \\ y \neq \vec{0}}} \|y\|$$

$$\det(\mathcal{L}) := \sqrt{\det(B^T B)} = \frac{\text{volume}}{\text{point}} = \frac{1}{\text{density}}.$$

$$\lambda_1(\mathcal{L}) \leq \sqrt{n}\, \det(\mathcal{L})^{1/n}$$

SVP: Find $v \in \mathcal{L}$ with $\|v\| = \lambda_1(\mathcal{L})$.

Lattices



$$\lambda_1(\mathcal{L}) := \min_{\substack{y \in \mathcal{L} \\ y \neq \vec{0}}} \|y\|$$

$$\det(\mathcal{L}) := \sqrt{\det(B^T B)} = \frac{\text{volume}}{\text{point}} = \frac{1}{\text{density}}.$$

$$\lambda_1(\mathcal{L}) \leq \sqrt{n} \det(\mathcal{L})^{1/n}$$

SVP: Find $v \in \mathcal{L}$ with $\|v\| = \lambda_1(\mathcal{L})$.

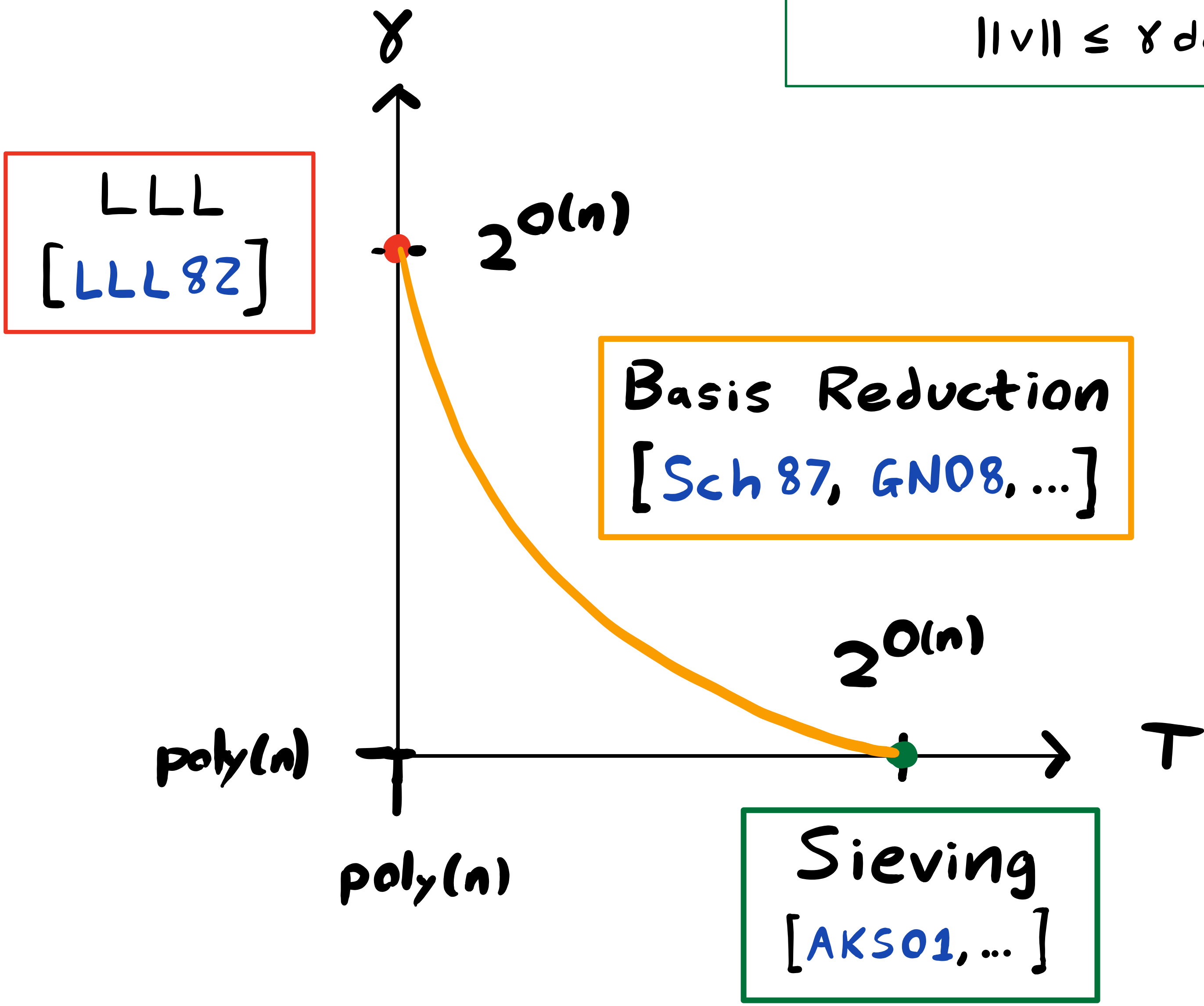
γ-approximate (H)SVP: Find $v \in \mathcal{L}$:

$$\|v\| \leq \gamma \det(\mathcal{L})^{1/n}$$

Time / Approximation Tradeoffs

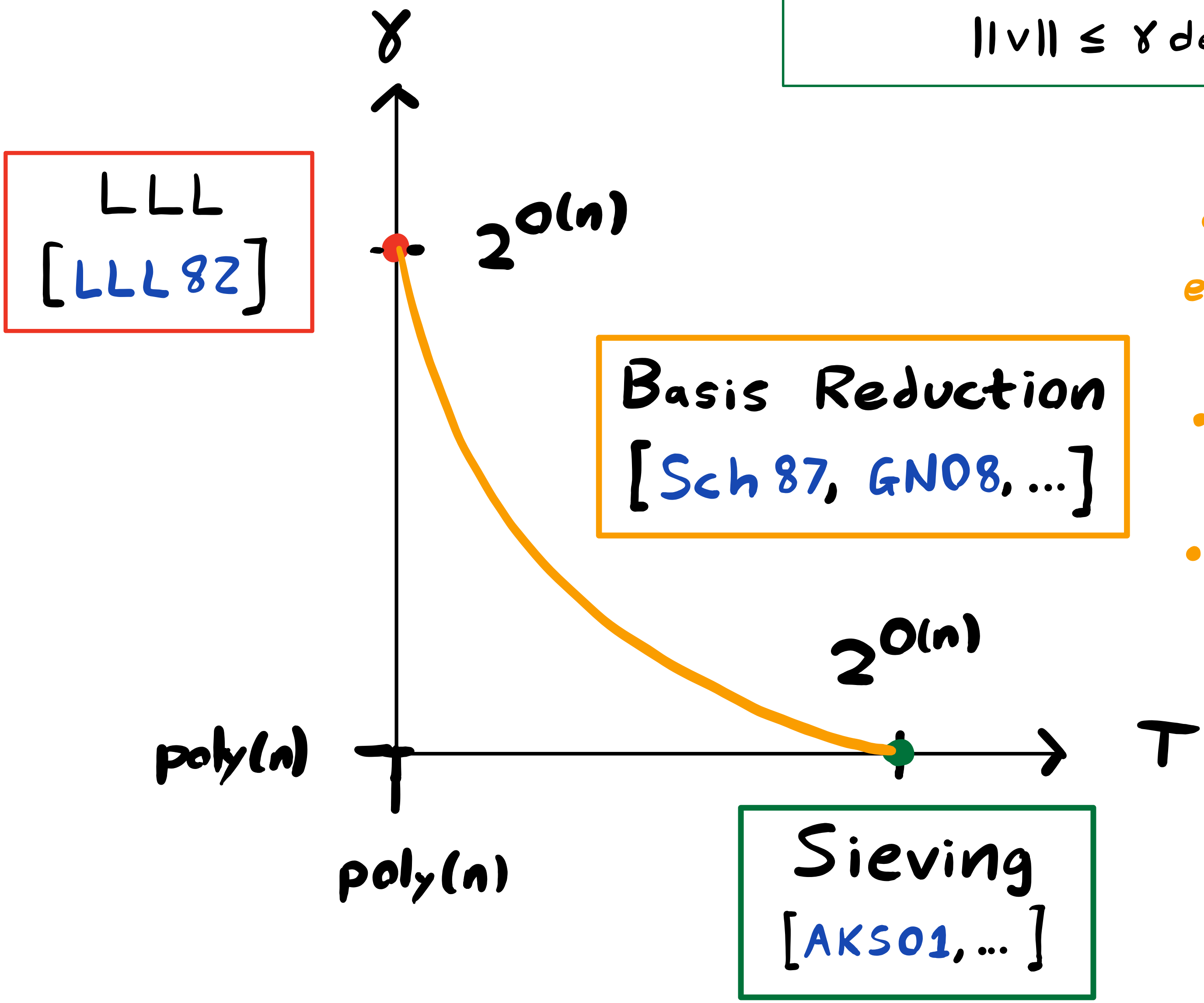
γ-approximate (H)SVP: Find $v \in \mathcal{L}$:

$$\|v\| \le \gamma \det(\mathcal{L})^{1/n}$$



Time / Approximation Tradeoffs

γ-approximate (H)SVP: Find $v \in \mathcal{L}$: $\|v\| \le \gamma \det(\mathcal{L})^{1/n}$



- Works by solving exact SVP in dimension k
- $T = \mathrm{poly}(n)\, 2^{O(k)}$
- $\gamma = k^{\frac{(n-1)}{2(k-1)}} \approx k^{\frac{n}{2k}}$.

Basis Reduction Algorithms

- All basis reduction algorithms follow the basic approach of LLL: iteratively improve the basis by solving SVP exactly in smaller dimension.

- Analysis is intricate
    - Our best (practical) algorithms are *heuristic*

Basis Reduction Algorithms

- All basis reduction algorithms follow the basic approach of LLL: iteratively improve the basis by solving SVP exactly in smaller dimension.

- Analysis is intricate
  - Our best (practical) algorithms are *heuristic*

- This talk: match Basis Reduction tradeoff with an "Algorithms 101"-style recursive algorithm.
  - "Recurse on a smaller-dimensional lattice!"

Basis Reduction Algorithms

- All basis reduction algorithms follow the basic approach of LLL: iteratively improve the basis by solving SVP exactly in smaller dimension.
- Analysis is intricate
  - Our best (practical) algorithms are *heuristic*
- This talk: match Basis Reduction tradeoff with an "Algorithms 101"-style recursive algorithm.
  - "Recurse on a smaller-dimensional lattice!"
  - Should be a *sublattice*

Basis Reduction Algorithms

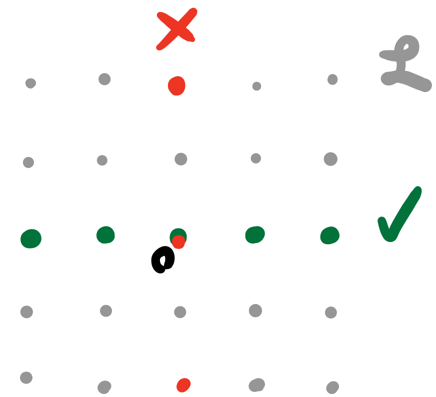
- All basis reduction algorithms follow the basic approach of LLL: iteratively improve the basis by solving SVP exactly in smaller dimension.

- Analysis is intricate
  - Our best (practical) algorithms are *heuristic*

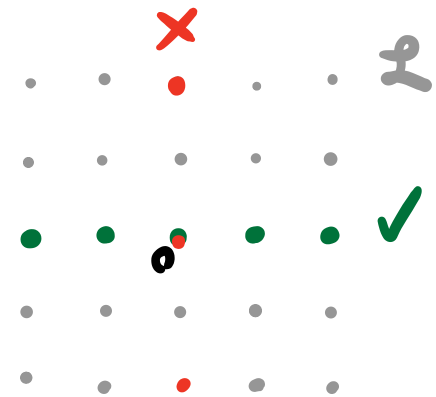
- This talk: match Basis Reduction tradeoff with an "Algorithms 101"-style recursive algorithm.
  - "Recurse on a smaller-dimensional lattice!"
  - Should be a *sublattice*
  - But should still have *short vectors* — i.e., small determinant.

The Densest Sublattice Problem

A sublattice $\mathcal{L}'$ of $\mathcal{L}$ is the intersection of $\mathcal{L}$ with a subspace.



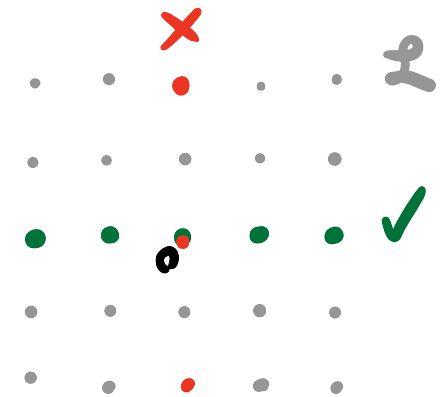
The Densest Sublattice Problem



A sublattice $\mathcal{L}'$ of $\mathcal{L}$ is the intersection of $\mathcal{L}$ with a subspace.

γ-$DSP_\ell(\mathcal{L})$: Find $\mathcal{L}' \subset \mathcal{L}$ of rank ℓ such that $\det(\mathcal{L}') \leq \gamma \cdot \det(\mathcal{L})^{\ell/n}$.

The Densest Sublattice Problem



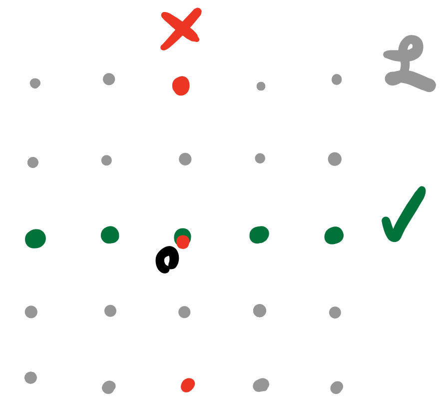
A sublattice $\mathcal{L}'$ of $\mathcal{L}$ is the intersection of $\mathcal{L}$ with a subspace.

γ-$DSP_\ell(\mathcal{L})$: Find $\mathcal{L}' \subset \mathcal{L}$ of rank ℓ such that $\det(\mathcal{L}') \le \gamma \cdot \det(\mathcal{L})^{\ell/n}$.

$$\gamma(\mathcal{L}', \mathcal{L}) := \frac{\det(\mathcal{L}')}{\det(\mathcal{L})^{\ell/n}}$$

$\sqrt{\delta_{n,\ell}}$ is the best γ possible in general (for worst-case $\mathcal{L}$)

The Densest Sublattice Problem



A sublattice $\mathcal{L}'$ of $\mathcal{L}$ is the intersection of $\mathcal{L}$ with a subspace.

γ-$\mathrm{DSP}_\ell(\mathcal{L})$: Find $\mathcal{L}' \subset \mathcal{L}$ of rank ℓ such that $\det(\mathcal{L}') \leq \gamma \cdot \det(\mathcal{L})^{\ell/n}$.

$$\gamma(\mathcal{L}', \mathcal{L}) := \frac{\det(\mathcal{L}')}{\det(\mathcal{L})^{\ell/n}}$$

$\sqrt{\delta_{n,\ell}}$ is the best γ possible in general (for worst-case $\mathcal{L}$)

γ-DSP with $\ell = 1$ is exactly γ-SVP.

An approach for solving SVP

$A(\mathcal{L})$:

1. Find dense sublattice $\mathcal{L}' \subset \mathcal{L}$
   (somehow)

2. Return $A(\mathcal{L}')$.

- For the base case, when $\mathrm{rank}(\mathcal{L}) = k$,
  output $SVP(\mathcal{L})$ — that is, use an exact algorithm.

γ-DSP is Composable

Consider $\mathcal{L}'' \subset \mathcal{L}' \subset \mathcal{L}$.

rank: $\ell < m < n$

γ-DSP is Composable

Consider $\mathcal{L}'' \subset \mathcal{L}' \subset \mathcal{L}$.

rank: $\ell < m < n$

$$\mathcal{L}'' \in \gamma_1\text{-}DSP_{\ell}(\mathcal{L}') \text{ AND } \mathcal{L}' \in \gamma_2\text{-}DSP_{m}(\mathcal{L})$$

γ-DSP is Composable

Consider $\mathcal{L}'' \subset \mathcal{L}' \subset \mathcal{L}$.

rank: $\ell < m < n$

$$\boxed{\begin{array}{l} \mathcal{L}'' \in \gamma_1\text{-}\mathrm{DSP}_\ell(\mathcal{L}') \text{ AND } \mathcal{L}' \in \gamma_2\text{-}\mathrm{DSP}_m(\mathcal{L}) \\ \Rightarrow \mathcal{L}'' \in \left(\gamma_1 \cdot \gamma_2^{\ell/m}\right)\text{-}\mathrm{DSP}_\ell(\mathcal{L}) \end{array}}$$

γ-DSP is Composable

Consider $\mathcal{L}'' \subset \mathcal{L}' \subset \mathcal{L}$.

rank: ℓ < m < n

$$\boxed{\begin{array}{l} \mathcal{L}'' \in \gamma_1\text{-}DSP_\ell(\mathcal{L}') \text{ AND } \mathcal{L}' \in \gamma_2\text{-}DSP_m(\mathcal{L}) \\ \Rightarrow \mathcal{L}'' \in \left(\gamma_1 \cdot \gamma_2^{\ell/m}\right)\text{-}DSP_\ell(\mathcal{L}) \end{array}}$$

<u>Proof.</u> $\det(\mathcal{L}'') \leq \gamma_1 \cdot \det(\mathcal{L}')^{\ell/m}$

$$= \gamma_1 \cdot \left(\gamma_2 \det(\mathcal{L})^{\ell/m}\right)^{m/n} = \gamma_1 \cdot \gamma_2^{\ell/m} \det(\mathcal{L})^{\ell/m}.$$

γ-DSP is Self-Dual

$$\boxed{\mathcal{L}^* := \{ w \in \mathrm{span}(\mathcal{L}) : \forall y \in \mathcal{L}, \langle w, y \rangle \in \mathbb{Z} \}}$$

$$\boxed{(\mathcal{L}^*)^* = \mathcal{L}} \qquad \boxed{\det(\mathcal{L}^*) = 1/\det(\mathcal{L})}$$

- There is a bijection from rank ℓ sublattices of $\mathcal{L}$ to rank $(n-\ell)$ sublattices of $\mathcal{L}^*$, preserving the approximation factor!

$$\boxed{\begin{gathered} \mathcal{L}' \in \gamma\text{-DSP}_\ell(\mathcal{L}) \\ \Longleftrightarrow \\ \mathcal{L}^* \cap (\mathcal{L}')^\perp \in \gamma\text{-DSP}_{n-\ell}(\mathcal{L}^*) \end{gathered}}$$

Important Special Case:

$$\boxed{\begin{gathered} w \in \gamma\text{-SVP}(\mathcal{L}^*) \\ \Longleftrightarrow \\ \mathcal{L} \cap w^\perp \in \gamma\text{-DSP}_{n-1}(\mathcal{L}). \end{gathered}}$$

γ-DSP is Self-Dual

$$\boxed{\mathcal{L}^* := \{ w \in \mathrm{span}(\mathcal{L}) : \forall y \in \mathcal{L}, \langle w, y \rangle \in \mathbb{Z} \}}$$

γ-DSP is Self-Dual

$$\boxed{\mathcal{L}^* := \{ w \in \mathrm{span}(\mathcal{L}) : \forall y \in \mathcal{L}, \langle w, y \rangle \in \mathbb{Z} \}}$$

$$\boxed{(\mathcal{L}^*)^* = \mathcal{L}}$$

$$\boxed{\det(\mathcal{L}^*) = 1 / \det(\mathcal{L})}$$

γ-DSP is Self-Dual

$$\boxed{\mathcal{L}^* := \{ w \in \operatorname{span}(\mathcal{L}) : \forall y \in \mathcal{L}, \langle w, y \rangle \in \mathbb{Z} \}}$$

$$\boxed{(\mathcal{L}^*)^* = \mathcal{L}} \qquad \boxed{\det(\mathcal{L}^*) = 1/\det(\mathcal{L})}$$

- There is a bijection from rank ℓ sublattices of $\mathcal{L}$ to rank $(n-\ell)$ sublattices of $\mathcal{L}^*$, preserving the approximation factor!

$$\boxed{\begin{gathered} \mathcal{L}' \in \gamma\text{-DSP}_\ell(\mathcal{L}) \\ \Leftrightarrow \\ \mathcal{L}^* \cap (\mathcal{L}')^\perp \in \gamma\text{-DSP}_{n-\ell}(\mathcal{L}^*) \end{gathered}}$$

γ-DSP is Self-Dual

$$\boxed{\mathcal{L}^* := \{ w \in \mathrm{span}(\mathcal{L}) : \forall y \in \mathcal{L}, \langle w, y \rangle \in \mathbb{Z} \}}$$

$$\boxed{(\mathcal{L}^*)^* = \mathcal{L}} \qquad \boxed{\det(\mathcal{L}^*) = 1/\det(\mathcal{L})}$$

- There is a bijection from rank ℓ sublattices of $\mathcal{L}$ to rank $(n-\ell)$ sublattices of $\mathcal{L}^*$, preserving the approximation factor!

$$\boxed{\begin{gathered} \mathcal{L}' \in \gamma\text{-DSP}_\ell(\mathcal{L}) \\ \Longleftrightarrow \\ \mathcal{L}^* \cap (\mathcal{L}')^\perp \in \gamma\text{-DSP}_{n-\ell}(\mathcal{L}^*) \end{gathered}}$$

Important Special Case:

$$\boxed{\begin{gathered} w \in \gamma\text{-SVP}(\mathcal{L}^*) \\ \Longleftrightarrow \\ \mathcal{L} \cap w^\perp \in \gamma\text{-DSP}_{n-1}(\mathcal{L}). \end{gathered}}$$

Warm-up: $\ell = 1$ (SVP $\Rightarrow$ SVP)

- Plan:
  1. $w \leftarrow A(\mathcal{L}^*)$
  2. $\mathcal{L}' := \mathcal{L} \cap w^\perp$
  3. Output $A(\mathcal{L}')$.

- Base case: if $\mathrm{rank}(\mathcal{L}) = k$, output $\mathrm{SVP}(\mathcal{L})$.

Warm-up: $\ell = 1$ (SVP $\Rightarrow$ SVP)

- Plan:
  1. $w \leftarrow A(\mathcal{L}^*)$
     1. $w \leftarrow A(\mathcal{L})$
        1. $w \leftarrow A(\mathcal{L}^*)$

           ...

Warm-up: $\ell = 1$ (SVP $\Rightarrow$ SVP)

- Plan:
  - Choose initial "depth parameter"

    $\tau = \tau(n) \geq 0.$

  1. $w \leftarrow A(\mathcal{L}^*, \tau - 1)$
  2. $\mathcal{L}' := \mathcal{L} \cap w^\perp$
  3. Output $A(\mathcal{L}', \tau)$

- Base cases:

  if $\tau = 0$, output $\mathrm{LLL}(\mathcal{L}, 1)$

  if $\mathrm{rank}(\mathcal{L}) = k$, output $\mathrm{SVP}(\mathcal{L})$.

Analysis

1. $w \leftarrow A(\mathcal{L}^*, \tau - 1)$
2. $\mathcal{L}' := \mathcal{L} \cap w^\perp$ (Dual)
3. Output $A(\mathcal{L}', \tau)$ (Composition)

Analysis

1. $\omega \leftarrow A(\mathcal{L}^*, \tau - 1)$
2. $\mathcal{L}' := \mathcal{L} \cap \omega^\perp$ (Dual)
3. Output $A(\mathcal{L}', \tau)$ (Composition)

$$\gamma(n, \tau) \leq \gamma(y, \mathcal{L}') \cdot \gamma(\mathcal{L}', \mathcal{L})^{\frac{1}{n-1}}$$

(Compositionality)
$l = 1, \quad m = n-1$

Analysis

1. $w \leftarrow A(\mathcal{L}^*, \tau - 1)$
2. $\mathcal{L}' := \mathcal{L} \cap w^\perp$ (Dual)
3. Output $A(\mathcal{L}', \tau)$ (Composition)

$$\gamma(n, \tau) \leq \gamma(y, \mathcal{L}') \cdot \gamma(\mathcal{L}', \mathcal{L})^{\frac{1}{n-1}}$$

(Compositionality) $\ell = 1$, $m = n-1$

$$= \gamma(y, \mathcal{L}') \cdot \gamma(w, \mathcal{L}^*)^{\frac{1}{n-1}}$$

(Duality)

Analysis

1. $w \leftarrow A(\mathcal{L}^*, \tau - 1)$
2. $\mathcal{L}' := \mathcal{L} \cap w^\perp$ (Dual)
3. Output $A(\mathcal{L}', \tau)$ (Composition)

$$\gamma(n, \tau) \leq \gamma(y, \mathcal{L}') \cdot \gamma(\mathcal{L}', \mathcal{L})^{\frac{1}{n-1}}$$ (Compositionality) $\ell = 1,\ m = n-1$

$$= \gamma(y, \mathcal{L}') \cdot \gamma(w, \mathcal{L}^*)^{\frac{1}{n-1}}$$ (Duality)

$$= \gamma(n-1, \tau) \cdot \gamma(n, \tau - 1)^{\frac{1}{n-1}}.$$ (By definition)

Analysis

$$\boxed{\gamma(n,\tau) \le \gamma(n-1,\tau)\cdot\gamma(n,\tau-1)^{\frac{1}{n-1}}}$$

$$\boxed{\gamma(n,0) = 2^n} \qquad \boxed{\gamma(k,\tau) = \sqrt{k}}$$

- Can check that by induction

$$\boxed{\gamma(n,\tau) \le k^{\frac{n-1}{2(k-1)}} \cdot \exp\left(n^3/2^\tau\right)}$$

- Taking $\tau = O(\log n)$ recovers block reduction:

$$\boxed{\gamma = (1+o(1))\, k^{\frac{n-1}{2(k-1)}}}$$

Analysis

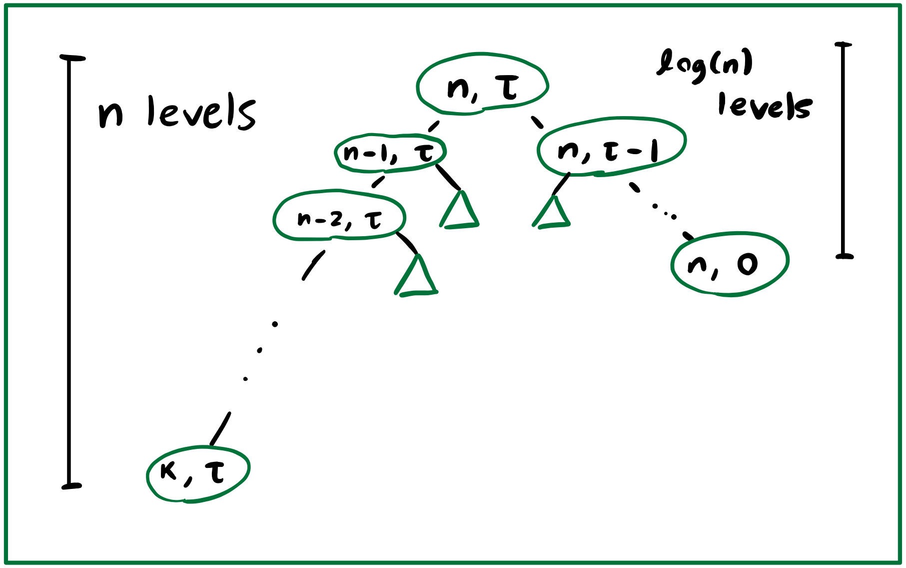
- SVP oracle calls dominate runtime.

$$\boxed{C(n, \tau) = C(n-1, \tau) + C(n, \tau-1)}$$

$$\boxed{C(k, \tau) = 1} \qquad \boxed{C(n, 0) = 0}$$

$$\boxed{C(n, \tau) = \binom{n-k+\tau-1}{\tau} \approx n^{\tau} = n^{O(\log n)}}$$

- Issue: Call tree highly unbalanced.



Take Two: $\ell > 1$ (DSP $\Rightarrow$ DSP)

- Plan:
  - Choose initial $\tau = O(\log n)$, $0 < \varepsilon < 1$.

1. $\hat{\mathcal{L}} \leftarrow A(\mathcal{L}^*, \varepsilon n, \tau - 1)$
2. $\mathcal{L}' := \mathcal{L} \cap (\hat{\mathcal{L}})^\perp$ // $\text{rank}(\mathcal{L}') = (1-\varepsilon)n$
3. Output $A(\mathcal{L}', \ell, \tau)$

- Base cases:

if $\tau = 0$, output $\text{LLL}(\mathcal{L}, \ell)$

if $\text{rank}(\mathcal{L}) = k$, output $\text{DSP}(\mathcal{L}, \ell)$.

Take Two: $\ell > 1$ (DSP $\Rightarrow$ DSP)

- Plan:
  - Choose initial $\tau = O(\log n)$, $0 < \varepsilon \ll 1$.

1. $\hat{\mathcal{L}} \leftarrow A(\mathcal{L}^*, \varepsilon n, \tau - 1)$
2. $\mathcal{L}' := \mathcal{L} \cap (\hat{\mathcal{L}})^\perp$ // $\mathrm{rank}(\mathcal{L}') = (1-\varepsilon)n$
3. Output $A(\mathcal{L}', \ell, \tau)$

$\boxed{\text{What if } \mathrm{rank}(\mathcal{L}') = (1-\varepsilon)n < \ell\text{? Can't find a larger-rank sublattice!}}$

Take Two: $\ell > 1$ (DSP $\Rightarrow$ DSP)

- Plan:
  - Choose initial $\tau = O(\log n)$, $0 < \varepsilon \ll 1$.

  0. If $\ell > n/2$,
     output $\mathcal{L} \cap (A(\mathcal{L}^*, n-\ell, \tau))^\perp$.
  1. $\hat{\mathcal{L}} \leftarrow A(\mathcal{L}^*, \lceil \varepsilon n \rceil, \tau - 1)$
  2. $\mathcal{L}' := \mathcal{L} \cap (\hat{\mathcal{L}})^\perp$ // $\mathrm{rank}(\mathcal{L}') = (1-\varepsilon)n$
  3. Output $A(\mathcal{L}', \ell, \tau)$

- Base cases:

  if $\tau = 0$, output $\mathrm{LLL}(\mathcal{L}, \ell)$.

  if $\mathrm{rank}(\mathcal{L}) = k$, output $\mathrm{DSP}(\mathcal{L}, \ell)$.

Analysis

- C = poly(n) oracle calls!

Analysis

- $C = \text{poly}(n)$ oracle calls!
- Recurrence becomes

$$\boxed{\gamma(n, \ell, \tau) \leq \gamma((1-\varepsilon)n, \ell, \tau) \cdot \gamma(n, \varepsilon n, \tau)^{\frac{\ell}{(1-\varepsilon)n}}}$$

Analysis

- $C = \text{poly}(n)$ oracle calls!
- Recurrence becomes

$$\boxed{\gamma(n, \ell, \tau) \leq \gamma((1-\varepsilon)n, \ell, \tau) \cdot \gamma(n, \varepsilon n, \tau)^{\frac{\ell}{(1-\varepsilon)n}}}$$

$$\boxed{\gamma(n, \ell, 0) = 2^{n\ell}}$$

$$\boxed{\gamma(k, \ell, \tau) = \sqrt{\delta_{k,\ell}} \approx k^{\frac{\ell(k-\ell)}{2(k-1)}}}$$

Analysis

- $C = \text{poly}(n)$ oracle calls!
- Recurrence becomes

$$\boxed{\gamma(n, \ell, \tau) \leq \gamma((1-\varepsilon)n, \ell, \tau) \cdot \gamma(n, \varepsilon n, \tau)^{\frac{\ell}{(1-\varepsilon)n}}}$$

$$\boxed{\gamma(n, \ell, 0) = 2^{n\ell}}$$

$$\boxed{\gamma(k, \ell, \tau) = \sqrt{\delta_{k,\ell}} \approx k^{\frac{\ell(k-\ell)}{2(k-1)}}}$$

- Can check

$$\boxed{\gamma(n, \ell, \tau) \leq k^{\frac{\ell(n-\ell)}{2(k-1)}} \cdot \exp\left(n^2 \ell(n-\ell)/2^{\tau}\right)}$$

Analysis

- $C = \text{poly}(n)$ oracle calls!
- Recurrence becomes

$$\boxed{\gamma(n,\ell,\tau) \leq \gamma((1-\varepsilon)n,\ \ell,\ \tau)\cdot\gamma(n,\ \varepsilon n,\tau)^{\frac{\ell}{(1-\varepsilon)n}}}$$

$$\boxed{\gamma(n,\ell,0) = 2^{n\ell}}$$

$$\boxed{\gamma(\kappa,\ \ell,\ \tau) = \sqrt{\delta_{\kappa,\ell}} \approx \kappa^{\frac{\ell(\kappa-\ell)}{2(\kappa-1)}}}$$

- Can check $\boxed{\gamma(n,\ell,\tau) \leq \kappa^{\frac{\ell(n-\ell)}{2(\kappa-1)}} \cdot \exp\left(n^2\ell(n-\ell)/2^{\tau}\right)}$

(To handle the duality step, note that the guess is symmetric under $\ell \mapsto (n-\ell)$.)

Analysis

- C = poly(n) oracle calls!
- Recurrence becomes

$$\boxed{\gamma(n,\ell,\tau) \le \gamma((1-\varepsilon)n,\ \ell,\ \tau)\cdot \gamma(n,\ \varepsilon n,\ \tau)^{\frac{\ell}{(1-\varepsilon)n}}}$$

$$\boxed{\gamma(n,\ell,0) = 2^{n\ell}}$$

$$\boxed{\gamma(K,\ell,\tau) = \sqrt{\delta_{K,\ell}} \approx K^{\frac{\ell(K-\ell)}{2(K-1)}}}$$

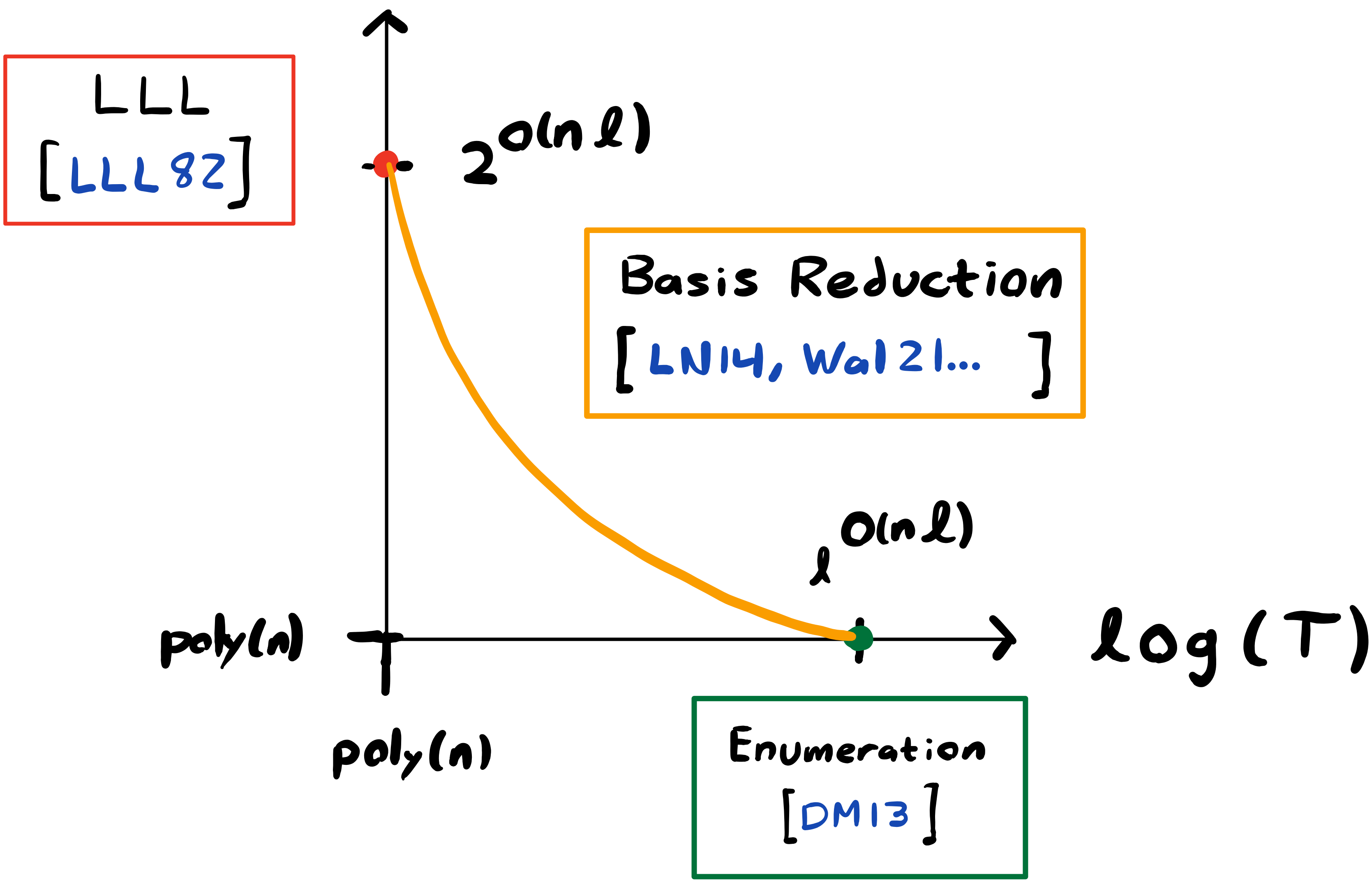
- Can check $\boxed{\gamma(n,\ell,\tau) \le K^{\frac{\ell(n-\ell)}{2(K-1)}} \cdot \exp\left(n^2\ell(n-\ell)/2^{\tau}\right)}$

(To handle the duality step, note that the guess is symmetric under $\ell \mapsto (n-\ell)$.)

$$\boxed{\gamma = (1+o(1))\,K^{\frac{\ell(n-\ell)}{2(K-1)}}}$$

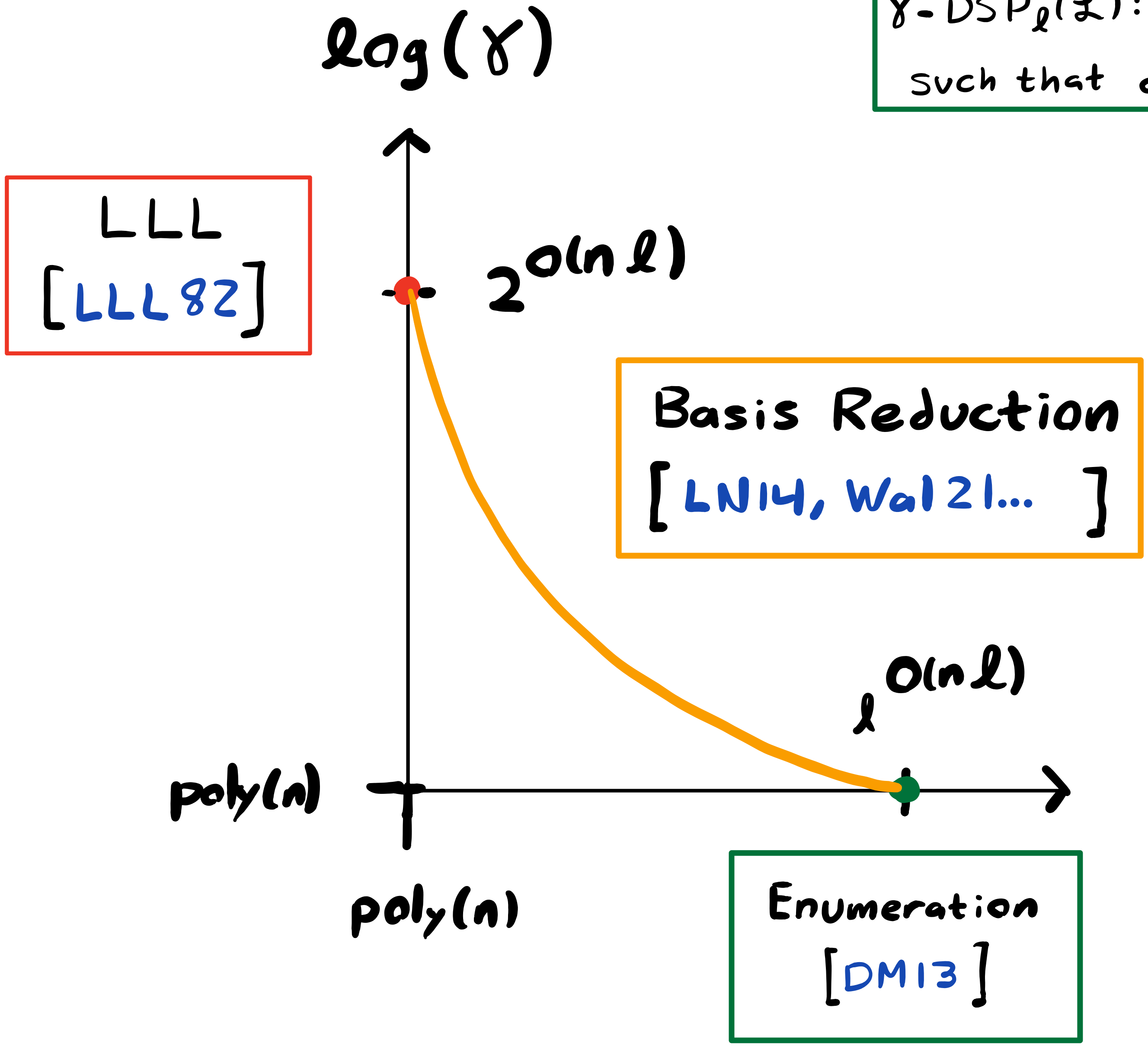
Time / γ Tradeoffs for γ-DSP_ℓ

γ-$\mathrm{DSP}_\ell(\mathcal{L})$: Find $\mathcal{L}' \subset \mathcal{L}$ of rank ℓ such that $\det(\mathcal{L}') \leq \gamma \cdot \det(\mathcal{L})^{\ell/n}$.



Time / γ Tradeoffs for γ-DSP_ℓ

γ-$\mathrm{DSP}_\ell(\mathcal{L})$: Find $\mathcal{L}' \subset \mathcal{L}$ of rank ℓ such that $\det(\mathcal{L}') \le \gamma \cdot \det(\mathcal{L})^{\ell/n}$.



- Works by solving exact DSP in dimension k
- $T = \mathrm{poly}(n)\, \ell^{O(k\ell)}$
- $\gamma = \delta_{k,\ell}^{\frac{n-\ell}{2(k-\ell)}} = k^{\Theta\left(\frac{\ell(n-\ell)}{(k-\ell)}\right)}$.
- Requires $\ell \le k$.

Analysis

- $C = \mathrm{poly}(n)$ oracle calls!
- Recovers basis reduction, even for $\ell > k$!

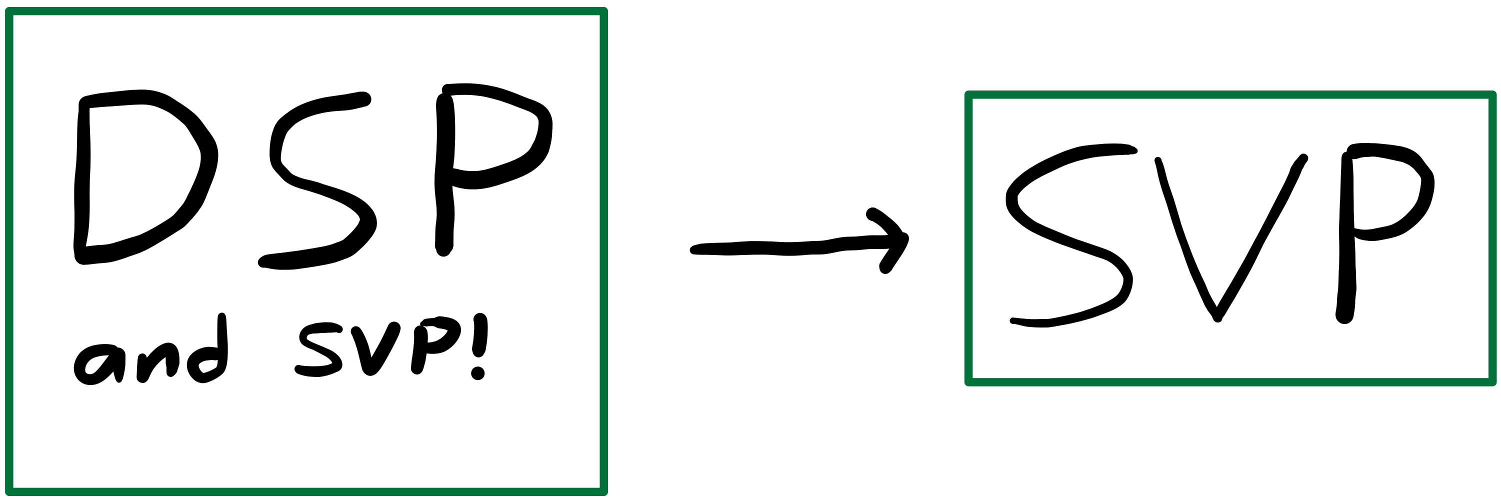
$$\boxed{\gamma = (1 + o(1))\, k^{\frac{\ell(n-\ell)}{2(k-1)}}}$$

Analysis

- $C = \mathrm{poly}(n)$ oracle calls!
- Recovers basis reduction, even for $\ell > k$!

$$\boxed{\gamma = (1 + o(1))\, k^{\frac{\ell(n-\ell)}{2(k-1)}}}$$

- Relies on conjecture $\sqrt{\delta_{k,\ell}} \approx k^{\frac{\ell(k-\ell)}{2(k-1)}}$
- DSP oracle calls are expensive.


DSP
and SVP!
SVP

Challenges

- The big challenge: make sure that

$$\boxed{n = k \Rightarrow \ell = 1 \quad (\text{or } \ell = n-1)}$$

Challenges

- The big challenge: make sure that

$$\boxed{n = k \Rightarrow \ell = 1 \quad (\text{or } \ell = n-1)}$$

- Solution idea: maintain the invariant that

$$\boxed{\hat{\ell} := \min\{\ell, n-\ell\} \leq n-k+1}$$

  - Squeeze ℓ to the extremes!

Challenges

- The big challenge: make sure that

$$\boxed{n = k \Rightarrow \ell = 1 \quad (\text{or } \ell = n-1)}$$

- Solution idea: maintain the invariant that

$$\boxed{\hat{\ell} := \min\{\ell, n-\ell\} \leq n - k + 1}$$

  - Squeeze ℓ to the extremes!

- Problem: Recurrence not favarable when ℓ is large.

Challenges

- The big challenge: make sure that

$$\boxed{n = k \Rightarrow \ell = 1 \quad (\text{or } \ell = n-1)}$$

- Solution idea: maintain the invariant that

$$\boxed{\hat{\ell} := \min\{\ell, n-\ell\} \leq n-k+1}$$

  - Squeeze ℓ to the extremes!

- **Problem:** Recurrence not favarable when ℓ is large.

- Solution idea: When ℓ is large, be patient:
  don't reduce the depth parameter τ.

The Reduction

Start with $\ell \leq n-k+1$.

- Duality: if $\max\{1, \frac{(n-k)}{5}\} < \ell < \frac{n}{2}$

  OR $\ell \geq n - \max\{1, \frac{(n-k)}{10}\}$:

  Output $\mathcal{L} \cap A(\mathcal{L}^*, n-\ell, \tau)^{\perp}$.

- Recursive step:

  $\hat{\mathcal{L}} \leftarrow A(\mathcal{L}^*, \lceil \frac{(n-k)}{20} \rceil, \tau - b)$

  Output $A(\mathcal{L} \cap (\hat{\mathcal{L}})^{\perp}, \ell, \tau)$.

  $$b = \begin{cases} 1, & \ell > \frac{n}{2} \\ 0, & \ell \leq \frac{n}{2} \end{cases}$$

- Base cases:

  if $\tau = 0$, output $LLL(\mathcal{L}, \ell)$.

  if $\text{rank}(\mathcal{L}) = k$ AND $\ell = 1$, return $SVP(\mathcal{L})$.

Analysis: Key Lemmas

All can be verified directly from local checks!

Lemma 1. All recursive calls satisfy

$$\min\{\ell, n-\ell\} \le n-k+1.$$

↳ Algorithm does not get "stuck".

Lemma 2. The potential $\Phi(n,\tau) := \tau + 20\log(n-k+1)$ drops by at least 1 from parent to grandchild.

↳ poly(n) runtime (oracle calls).

Lemma 3. The guess $f(n,\ell,\tau) := k^{\frac{\ell(n-\ell)}{2(k-1)}} \cdot \exp(n^3/2^\tau)$ satisfies the recurrence induced by $\mathcal{A}$.

↳ Achieve basis reduction tradeoff $k^{\frac{\ell(n-\ell)}{2(k-1)}}$.

Computer-Aided Search

- Our DSP → SVP involved tricky parameter choices.

  Why not let the computer do it?

Computer-Aided Search

- Our DSP → SVP involved tricky parameter choices.

  Why not let the computer do it?

Duality: if CONDITION(n, ℓ, τ), output $\mathcal{L} \cap A(\mathcal{L}^*, n-\ell, c)^\perp$

Recursive step: $\hat{\mathcal{L}} \leftarrow A(\mathcal{L}^*, \ell^*, c^*)$

Output $A(\mathcal{L} \cap (\hat{\mathcal{L}})^\perp, \ell, c - c^*)$.

Base cases: $\tau = 0$, output LLL$(\mathcal{L}, \ell)$; if $n = k$ AND $\ell = 1$, output SVP$(\mathcal{L})$.

Computer-Aided Search

- Our DSP → SVP involved tricky parameter choices.

  Why not let the computer do it?

Duality: if CONDITION(n, ℓ, τ), output $\mathcal{L} \cap A(\mathcal{L}^*, n-\ell, c)^\perp$

Recursive step: $\hat{\mathcal{L}} \leftarrow A(\mathcal{L}^*, \ell^*, c^*)$

Output $A(\mathcal{L} \cap (\hat{\mathcal{L}})^\perp, \ell, c-c^*)$.

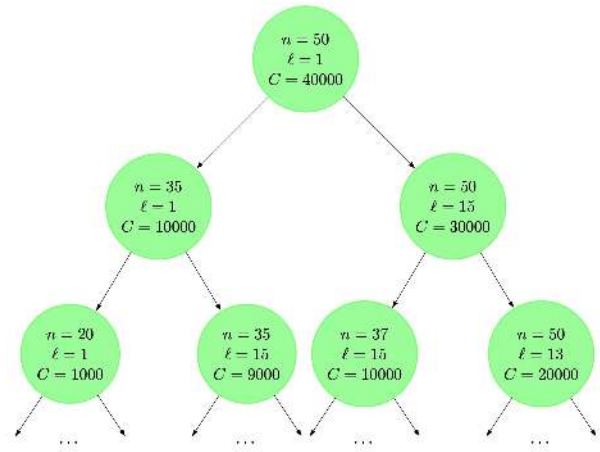
Base cases: $\tau = 0$, output LLL$(\mathcal{L}, \ell)$; if $n = k$ AND $\ell = 1$, output SVP$(\mathcal{L})$.

- Using dynamic programming, solve

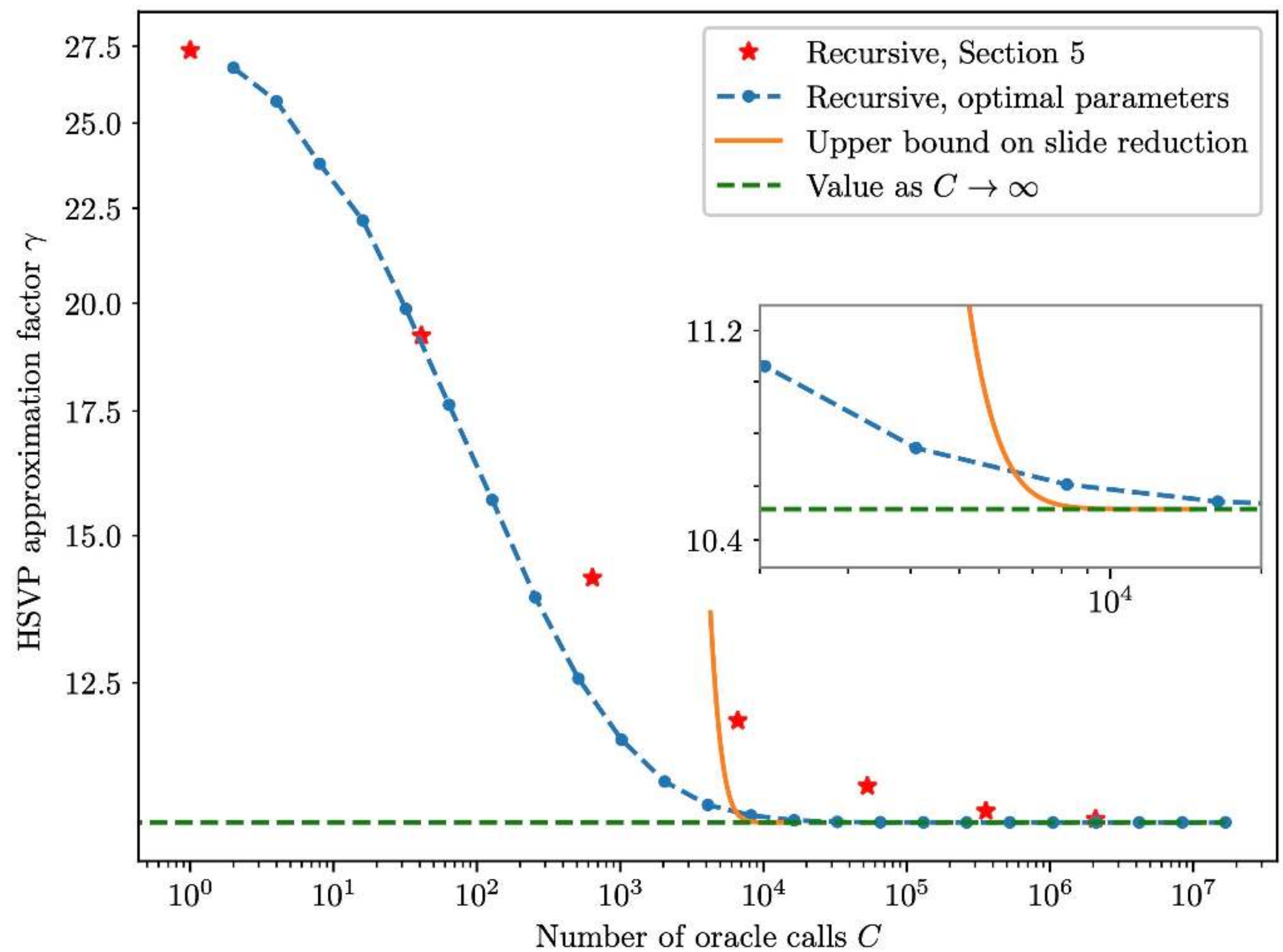
$$\gamma(n, \ell, c) := \min \left\{ \begin{array}{l} \gamma(n, n-\ell, c) \\ \displaystyle\min_{1 \le \ell^* \le n-k} \ \min_{c^* \le c} \ \gamma(n-\ell^*, \ell, c-c^*)\, \gamma(n, \ell^*, c^*)^{\frac{\ell}{n-\ell^*}} \end{array} \right\}$$

Results

- Optimal DP solution bounds rather massively improve on DSP → SVP guarantee.

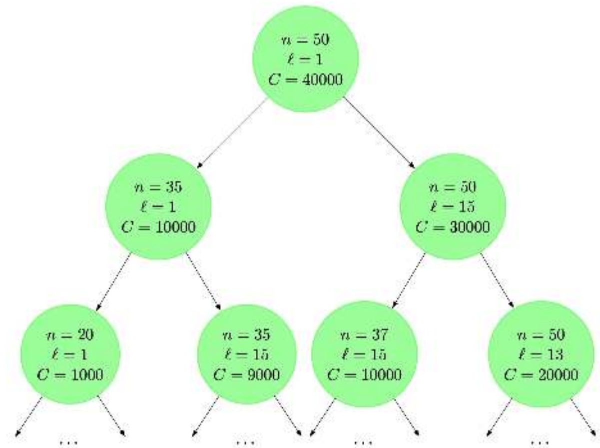


n = 50, K = 10



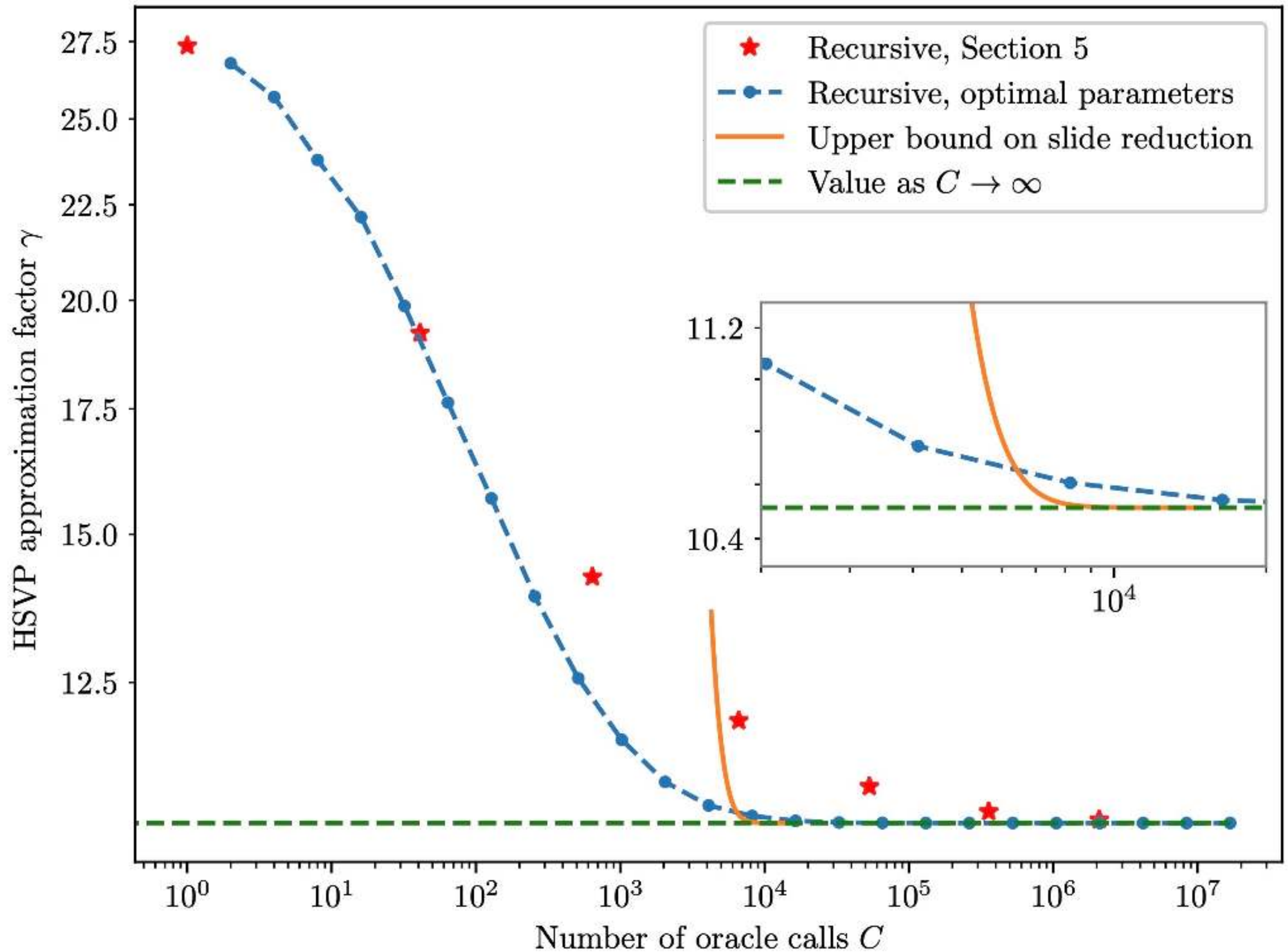
Results

- Optimal DP solution bounds rather massively improve on DSP → SVP guarantee.



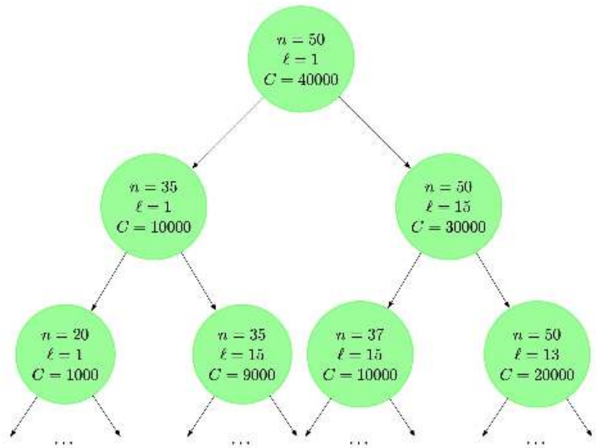
- Comparable to provable bounds on slide reduction.

n = 50, K = 10



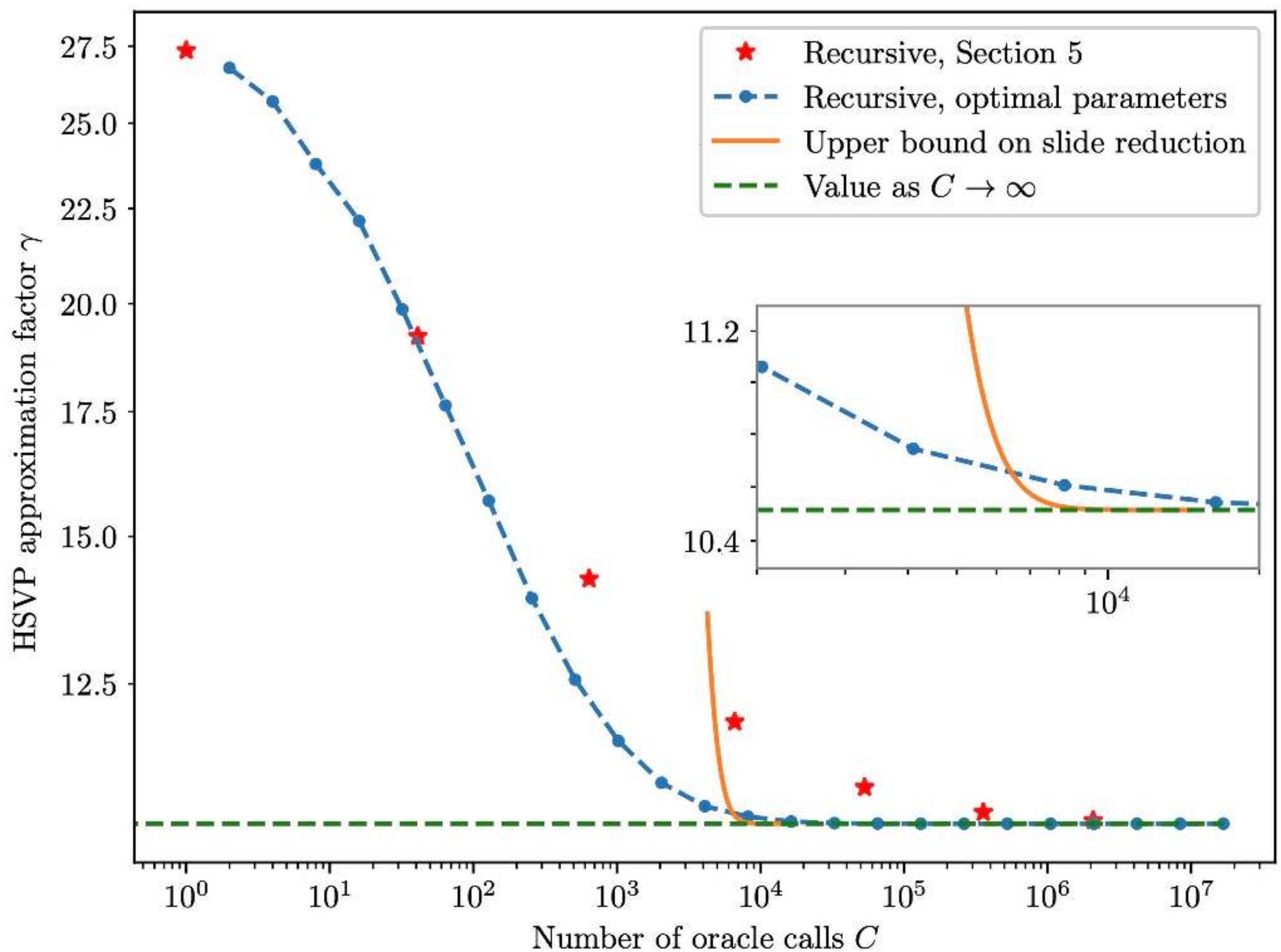
Results

- Optimal DP solution bounds rather massively improve on DSP → SVP guarantee.

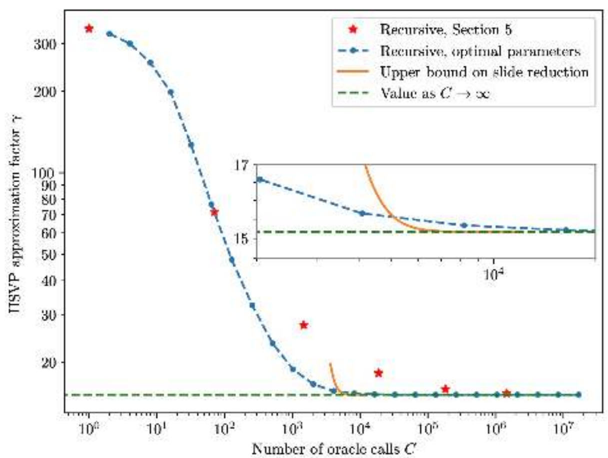


- Comparable to provable bounds on slide reduction.

$n = 50$, $\kappa = 10$



$n = 100$, $\kappa = 30$



Mixing and Matching Oracles

- Rough estimate: solving SVP in dimension n takes time 2^n.

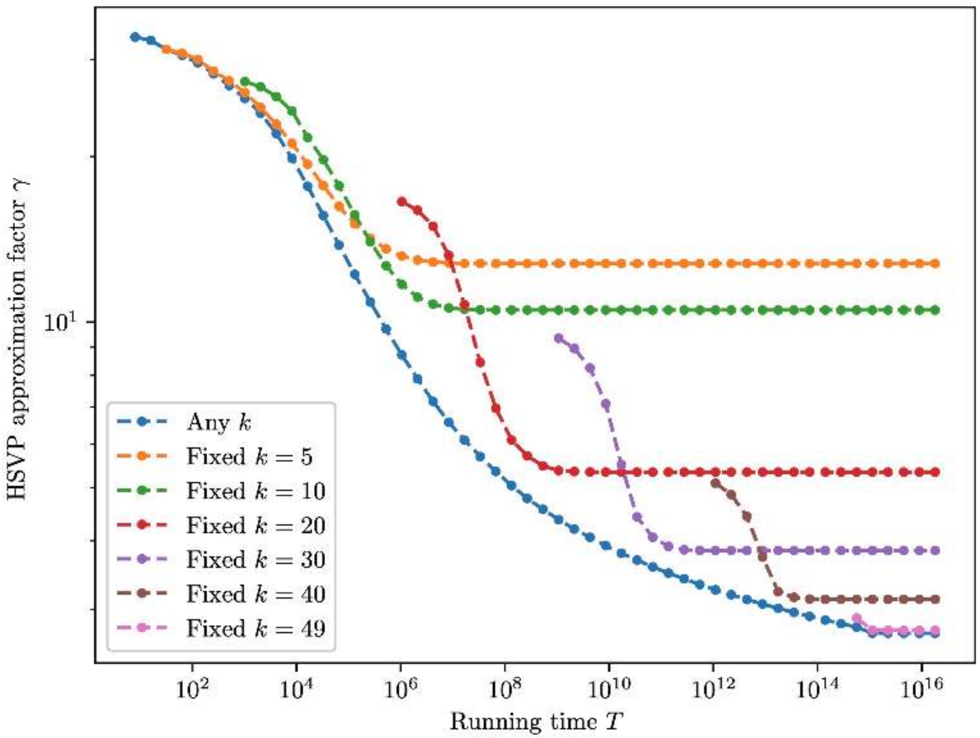
$\boxed{\begin{array}{l}\text{Optional Base Case: For all } T \geq 2^n, \\ \qquad \gamma(n, 1, T) = \sqrt{n}.\end{array}}$

Mixing and Matching Oracles

- Rough estimate: solving SVP in dimension n takes time 2^n.

$$\boxed{\text{Optional Base Case: For all } T \geq 2^n, \quad \gamma(n, 1, T) = \sqrt{n}.}$$

$n = 50, \ k = 10$

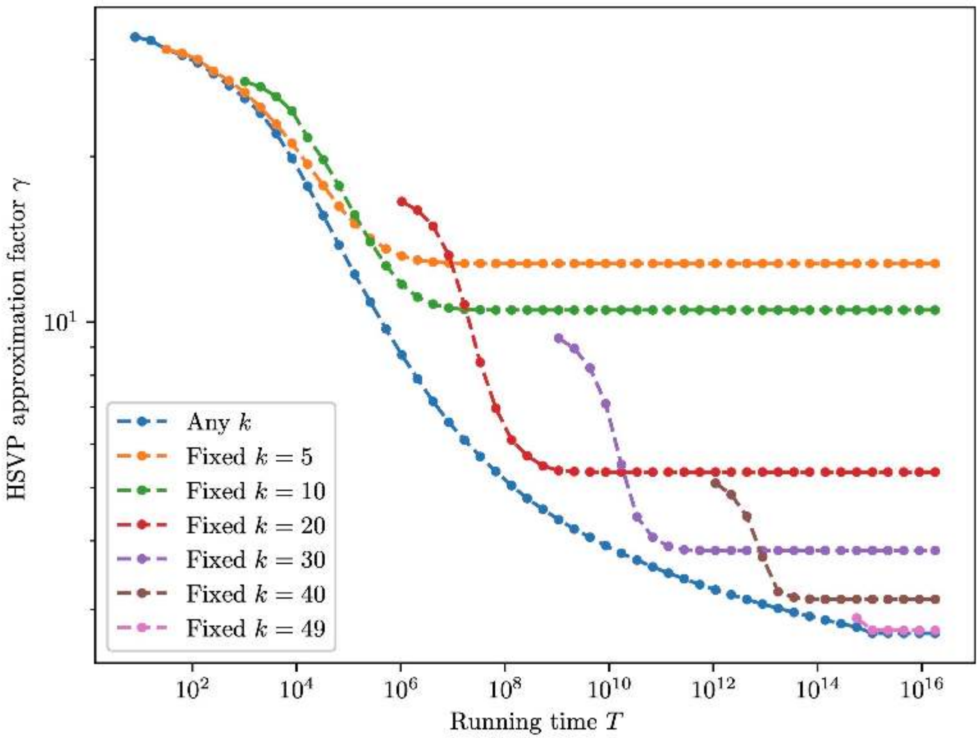


Mixing and Matching Oracles

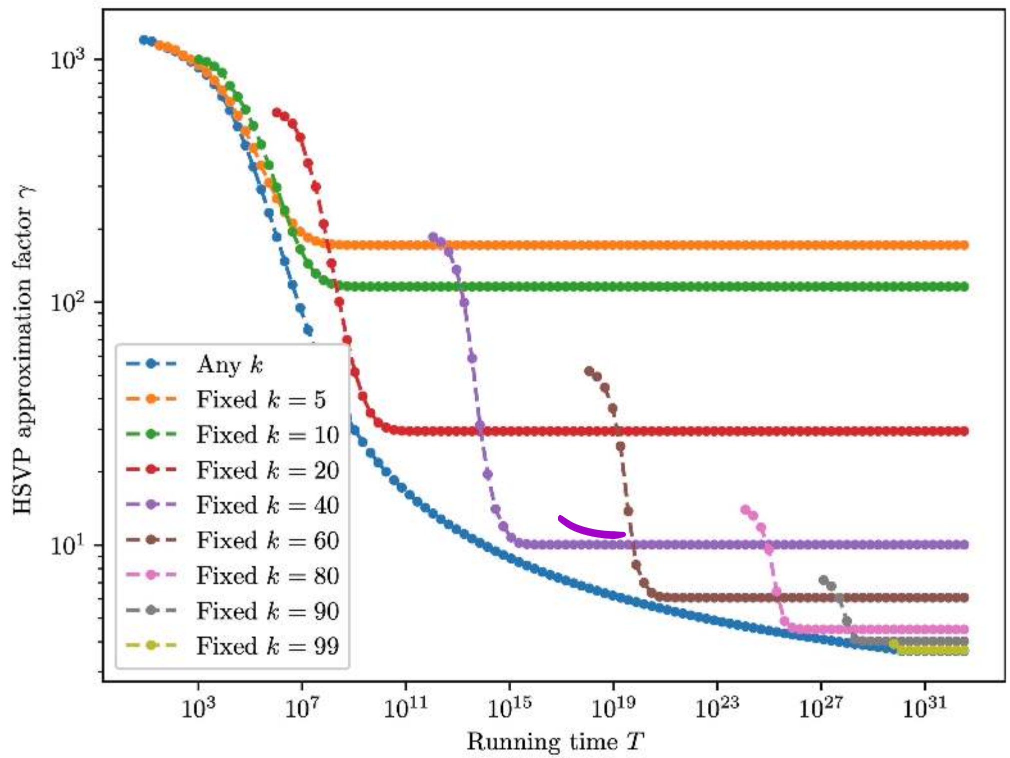
- Rough estimate: solving SVP in dimension n takes time 2^n.

> **Optional Base Case:** For all $T \geq 2^n$,
> $$\gamma(n, 1, T) = \sqrt{n}.$$

$n = 50,\ k = 10$



$n = 100,\ k = 30$



Future Directions

- **Explore the design space**
  - Tuning parameters to exploit structure
  - Trees of higher arity?

Future Directions

- Explore the design space
    - Tuning parameters to exploit structure
    - Trees of higher arity?
- Basis Reduction and Recursive Lattice Reduction
    - Is there a formal relationship?
    - Can recursive lattice reduction capture LLL?
    - Mix and match?

Future Directions

- Explore the design space
  - Tuning parameters to exploit structure
  - Trees of higher arity?
- Basis Reduction and Recursive Lattice Reduction
  - Is there a formal relationship?
  - Can recursive lattice reduction capture LLL?
  - Mix and match?
- Practical behavior: Implementation and heuristic analysis

Future Directions

- Explore the design space
  - Tuning parameters to exploit structure
  - Trees of higher arity?
- Basis Reduction and Recursive Lattice Reduction
  - Is there a formal relationship?
  - Can recursive lattice reduction capture LLL?
  - Mix and match?
- Practical behavior: Implementation and heuristic analysis
- Study reductions that use very few oracle calls.

Thanks for listening!