A New Class of Algorithms for Finding Short Vectors in Lattices Lifted from Co-dimension k Codes

Robert Lin (Harvard)
Joint work with Peter Shor, arXiv:2401.12383

ANTS XVI lightning talk, July 16th, 2024

Results

- **Problem:** $\vec{w} \cdot \vec{v} = \sum_{i=1}^{d} w_i v_i = 0 \pmod{P}$ defines a lattice in $\mathbb{Z}^d$. Want to find a short lattice vector.
- Our new approach focuses on finding short vectors in this lattice by starting from short vectors not in the lattice
- **Basic idea**: 1. Sort vectors by $\vec{w} \cdot \vec{v}_i$. 2. Apply one step of Euclidean algorithm to neighboring projections, extend linearly to vectors. (This step is *parallelizable*.) 3. Repeat.
- **Input set of d unit vectors** yields $o(d^2)$-time, $O(d^2)$-space algorithm with length scaling as $z^{\#\text{iter}}$ for a random lattice
- When d is large, $z = \sqrt{2}$, $\#\text{iter} \sim \log P / \log d$.
- For fixed P, as d increases, the length *decreases*.

Generalizations

- **Two axes of generalization**: (Lengths are for random lattices)

Large input list of d^* vectors	Multi-partite reduction
$O(d^*)$-time for $\ln(d^*) \sim d$	$o(d^{\max(2,k-1)})$-time ($k \geq 2$)
$O(d^* d)$ space	$O(d^2)$ space
Length $L \sim L_0 \sqrt{2}^{\log P/\log(d^*)}$	Length $L \sim \sqrt{k}^{\log P/(k-1)\log(d)}$

- Solves open problem (Stephens-Davidowitz, '20): Solve approx-SVP(shortest vector problem) without invoking subroutine to solve SVP.
- Numerical results show advantage vs. LLL(L^2) even without the two axes of generalization: two to three orders of magnitude speed-up on test cases, with better or comparable lengths.
- First axis of generalization applied successfully to Darmstadt SVP Challenge for $d = 40, 42$.