Final Project
Approximating shortest lattice vectors is not harder than approximating closest lattice vectors

March 12, 2012

Katrina Glaeser

Background

Given a basis of linearly independent vectors $B = b_1, \ldots, b_n$ in $\mathbb{R}^m$, the lattice $\mathcal{L}$ is the set of integer linear combinations of the basis vectors.

$$\mathcal{L} = \{\sum_{i=1}^{n} a_i b_i | a_i \in \mathbb{Z}\}$$

When your basis is far from orthogonal, it can be difficult to imagine what this set of points in space will look like, especially when the dimension spanned by the subspace is high.

The **Shortest Vector Problem (SVP)** is the problem of finding a lattice point $\mathbf{u}$ which minimizes $\|\mathbf{u}\|$ i.e. the shortest nonzero lattice point. A similar problem is when you are given a vector $\mathbf{w}$ not in the lattice, to find the closest lattice point $\mathbf{u} \in \mathcal{L}$ which minimizes $\|\mathbf{u} - \mathbf{w}\|$, this is called the **Closest Vector Problem (CVP)** . As the dimension n of the lattice grows these problems become computationally expensive, in fact they are NP hard.

There is also the problem of finding approximate solutions to the SVP and CVP, ie to within some multiple of the minimal solution. This can be written as SVP_f, and will return a vector $\mathbf{v} \in \mathcal{L}$ such that for some function $f : \mathbb{N} \to \mathbb{R}_{\geq 1}$

$$\|\mathbf{v}\| \leq f(n)\|\mathbf{u}\| \text{ for all } \mathbf{u} \in \mathcal{L}$$

since the shortest vector is among the lattice points we can consider this to be no more that a multiple of $f(n)$ off from the actual shortest vector. CVP_f is defined similarly.

There is also a decisional version of this problem called GapSVP_f where the input is $(\mathcal{L}, d)$ a lattice and a guess at the minimal distance d. A GapSVP_f solver will distinguish between the cases where there exists a vector with $\|\mathbf{v}\| \leq d$ and of the cases where no vector $\|\mathbf{v}\| \leq f(n) \cdot d$ exists. GapCVP_f is defined to run similarly on the input $(\mathcal{L}, \mathbf{w}, d)$.

Reduction of SVP to CVP

In the paper *Approximating shortest lattice vectors is not harder than approximating closest lattice vectors* by Goldreich, Micciancio, Safra and Seifert, there is a very nice reduction of the SVP to the CVP.

Since the zero vector is a lattice point, it won't work to supply a CVP oracle with $\mathbf{w} = \mathbf{0}$, because the closest lattice point is $\mathbf{0}$ itself. The paper uses a cute trick which is based on the idea that if we knew shortest vector $\mathbf{u} = \sum_{i=1}^{n} a_i b_i$ then surely the $a_i \in \mathbb{Z}$ cannot all be even. If a_j is odd then the by taking a sub-lattice which replaces b_i with $2b_i$ to create a new basis $B^{(j)}$. By finding inputting $\mathbf{w} = b_i$ as the vector which is not in the sub-lattice, we can find a closest lattice point $\mathbf{v}$ which will have minimal $\|\mathbf{v} - \mathbf{b_i}\|$. If we tried doing this for each of the n basis vectors we could take the solution from whichever gave us the shortest distance. Clever tricks like this make me love computer science and math.

Approximate version

The authors of the paper also have a randomized reduction of GapSVP_f to GapCVP_f. This reduction is not as nice as the reduction from SVP to CVP, but it uses an argument that reminded me of the reductions from this

class.

The proof relies on the fact that if GapSVP_f was a yes case there is a shortest vector that satisfied $\|\mathbf{v}\| \leq d$, from this you could construct a vector with the use of randomly sampled 0's and 1's to create a vector which has probability at least $\frac{1}{2}$ of being a lattice point within distance d of a target vector. It seems backwards at first, to show that the construction of this point required a solution to the SVP, but it is ok, because it is only used to show the existence of a solution to GapSVP_f.

The paper did not provide a version of the proof that would extend this to a reduction from SVP_f to CVP_f. It might seem strange that we could be able to reduce SVP_f to CVP_f without explicitly constructing an instance of CVP_f which solves another instance SVP_f. The reduction of GapSVP_f to GapCVP_f used a family of instances of GapCVP_f, and showed that each one had a bounded probability. Perhaps the logic behind the reduction from SVP_f to CVP_f is similar to the hybrid arguments we used in class, where without explicitly constructing an adversary with a certain advantage we can assume that one exists, which is enough.

Some other ideas

The GapCVP_f might seem to provide very little information about what the actual solution is. Something that GapCVP_f does allow you to do, is cover a ball of radius d around the target vector provided and say conclusively that no lattice point exists in that ball. Depending on the norm that you are using the ball will have a different shape. In the paper *Covering Cubes and the Closest Vector Problem* by Friedrich Eisenbrand, Nicolai Hahnle, Martin Niemeier the authors use turn these GapCVP solvers as the idea for some cool covering problems.