jh190038-NAJ

大規模並列計算による格子の最短ベクトル探索の効率化に関する研究

代表：照屋 唯紀 (産業技術総合研究所)

副代表：柏原 賢二 (東京大学)

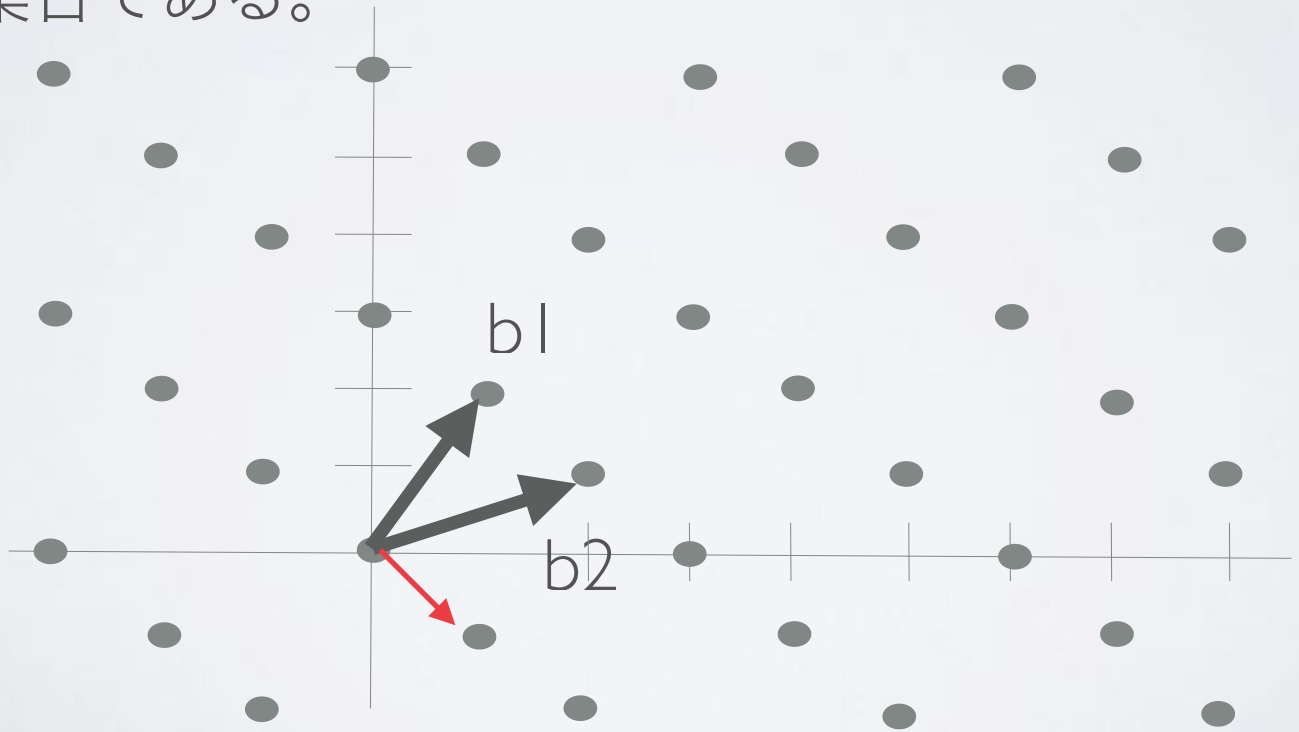
最短ベクトル問題とは

- 行列式が0でない、(整数成分の) N次正方行列に対して
- 格子：行列の列ベクトルの整数結合で表される点の全体。
- 最短ベクトル：格子の点のうち、原点以外で原点にもっとも近い点。
- 最短ベクトル問題(Shortest Vector Problem, SVP)とは、格子を生成する行列から最短ベクトルを求める問題。
- NP困難問題として知られている。

2次元の格子の例

B によって張られる格子：　$\{xb_1 + yb_2 : x, y \in \mathbb{Z}\}$

- 格子とは、線形独立な整数ベクトルの整数係数線形結合の全ての集合である。



基底簡約問題

- 格子の最短ベクトル問題を解くためには、いくつかのアプローチがある。
- 基底簡約問題に帰着させる方法は有力な手法の一つ。
- 基底簡約とは基底を簡単なものに基底変換すること。
  - 例えば、基底ベクトルが互いに直交に近い。

基底簡約問題の応用

- 格子の最短ベクトル問題、基底簡約問題は数論や計算機科学の多くの問題との関連で研究されている。
  - 整数論
  - 多項式の方程式の求解
  - 暗号システム、暗号解読

SIEVING (格子篩)

- Sievingも最短ベクトル問題を解く有力な方法。
- 既知の格子ベクトル集合の和と差を考えることにより、短いベクトルを見つける。
- 適宜、長いベクトルを短いベクトルで置き換えていき、だんだんベクトル集合を短くしていく。
- 短いベクトル同士の和や差は短いものである確率が上がるという原理を利用している。
- 次元が高いと、大量のベクトルが必要になる。しかし、効率は高い。

SVP CHALLENGE

- 格子の最短ベクトル問題で、如何に難しい問題を解けるかを競っているサイト(ダルムシュタット工科大学)がある。
  - https://www.latticechallenge.org/svp-challenge/
  - 次元が高い格子ほど、難しい。
  - 同じ次元ならば、短いベクトルがよい。
- 各次元で、設定された基準(最短推測値の1.05倍)より短くて、今までより最短のベクトルがエントリーできる。

SVP CHALLENGE

HALL OF FAME

Position	Dimension	Euclidean Norm	Seed	Contestant	Solution	Algorithm	Subm. Date	Approx. Factor
1	170	3438	0	L. Ducas, M. Stevens, W. van Woerden	vec	Sieving	2020-05-12	1.04690
2	157	3320	0	L. Ducas, M. Stevens, W. van Woerden	vec	Sieving	2019-05-20	1.04906
3	155	3165	0	M. Albrecht, L. Ducas, G. Herold, E. Kirshanova, E. Postlethwaite, M. Stevens, P. Karpman	vec	Sieving	2018-09-18	1.00803
4	153	3192	0	Martin Albrecht, Leo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn Postlethwaite, Marc Stevens	vec	Sieving	2018-08-30	1.02102
5	152	3217	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other	2018-10-3	1.03313
6	151	3233	0	Martin Albrecht, Leo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn Postlethwaite, Marc Stevens	vec	Sieving	2018-08-30	1.04411
7	150	3212	0	Yuga Miyagi and Eiichiro Fujisaki	vec	Sieving	2020-02-12	1.03914
8	150	3220	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other	2017-01-11	1.04192

全511エントリー (2020年6月20日現在)

ALBRECHTらの手法

- Albrecht et al., "The General Sieve Kernel and New Records in Lattice Reduction," EUROCRYPT 2018
  - Sievingを直交補空間(射影空間)に対して用いている。
  - Sievingにおいて、Locality Sensitive Hashingを利用している。

直交補空間

- k番目の直交補空間とは、1番目からk-1番目の基底ベクトルに垂直な空間。k番目以降の直交基底ベクトルで張られる
- 格子全体を、直交補空間に射影した点の集合を考えることができる。
- 基底簡約とは、各直交補空間において、格子点が射影されたベクトルのなかで、より短いものを探すこと。

$$\pi_{k-1}(x) = \sum_{i=k}^{n} \nu_i b_i^*$$

$$x = \sum_{i=1}^{n} \nu_i b_i^*$$

本研究の目的

- ベクトル生成に、Sievingの手法を利用
- Sievingアルゴリズムに対し、MPIを用いた大規模並列することで基底簡約計算を高速化を目指す。
- 今回の研究は、基底簡約部分ではなく、射影空間で短いベクトルを見つける部分に焦点をあてる。
  - 得られた短いベクトルを基底簡約に役立てたい。

SIEVINGの困難な点

- Sievingの計算を継続させるには、十分な量のベクトルが必要。
- ベクトルの重複に気をつける。
- 計算の重複に気をつける。
- 並列に計算するには、どのようにデータを分割するか考える必要がある。

アルゴリズム

射影空間でのSIEVING

- Sievingは、次元が高くなると大量のベクトルが必要になる。
- 次元の低い直交補空間で、Sievingを行うのが効率的。
- 射影空間でのSievingによって、射影空間で短いベクトルが見つかる。
- 全体空間へはBabai liftで拡張すればよい。

BABAI LIFT

- 射影空間のベクトルを全体空間へと変換する際、1次元ごとに原点に近くなるように係数を求める
- 後ろのほうから順番に最も射影長さが短くなるように係数を決めていく。
  - $(0,0,a_3,a_4) \to (0,a_2,a_3,a_4) \to (a_1,a_2,a_3,a_4)$：原点に近くなるように、$a_2$、$a_1$を順に求める。
- これにより、「よい基底」に対しては、後ろの部分空間で短いベクトルが見つかれば、全体でも短いベクトルが見つかる確率が高くなる。

アルゴリズムの工夫

1. ステップに分けてSievingを考えた。
2. 考える部分空間を大きくしたり小さくしたり動かした。
3. たくさんのベクトル集合を分割して、並列化した。

- 上の3つのそれぞれがループを作り、ネストしている。
- 今回は、並列化に焦点をあてて説明する。

ベクトルの位置関数

- ベクトルの座標からHash値のようなものを考え、保存位置を決める。
- 保存位置により、Sievingの処理を分割する。

ベクトルの重複チェック

- modをとるための大きな数(100万程度)を設定
- ベクトルの重複チェックは、全体長さmodの値と射影空間について、最短のベクトルだけを残すようにする。
- 局所的に判断できるので、並列化に向いている。

並列アルゴリズムの概要

1. 入力：格子ベクトルの集合、プロセス番号
2. for g in 他のベクトルグループ：
3. 自分のプロセス番号に対応するベクトルの集合と、それ以外のベクトルの集合gの組を考えて、和と差を計算。(cuBLAS)
4. その位置でもっとも短い場合に保存。(MPI-IO)

MPI-IOの利用

- Sievingで問題を解くには、大量のベクトルが必要になる。
- ファイルなどを使って、大量のベクトル情報を共有する。
- 大量にファイルを読み書きしないといけないので、高速化が必要。
  - MPI-IOを利用した。

線形計算ライブラリの利用

- Sievingでは、大量のベクトルのペアx, yについて、
  $\|x \pm y\|^2 = \|x\|^2 + \|y\|^2 \pm 2\langle x, y\rangle$を求める。
- 巨大行列積となり、これを高速に計算したい。
- 線形計算を伴う計算の高速化には、BLASと呼ばれる効率のよい線形計算を用いるライブラリが使用される。
- CuBLASと呼ばれるcudaを用いたGPU計算のBLASを一部使用している。

結論

- 154次元の格子の問題に取り組んだ。しかし、最短ベクトルは見つからなかった。
- プログラムは、意図通りに動いたが、期待したほどうまくはいかなかった。
- 大規模Sievingにおいて、短くなるのに時間がかかりすぎた。考えている次元が高すぎると思われる。
- 今後は、方針を変えて最短ベクトル問題に取り組みたい。