jh200005

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

研究代表者：柏原 賢二　産総研/東大

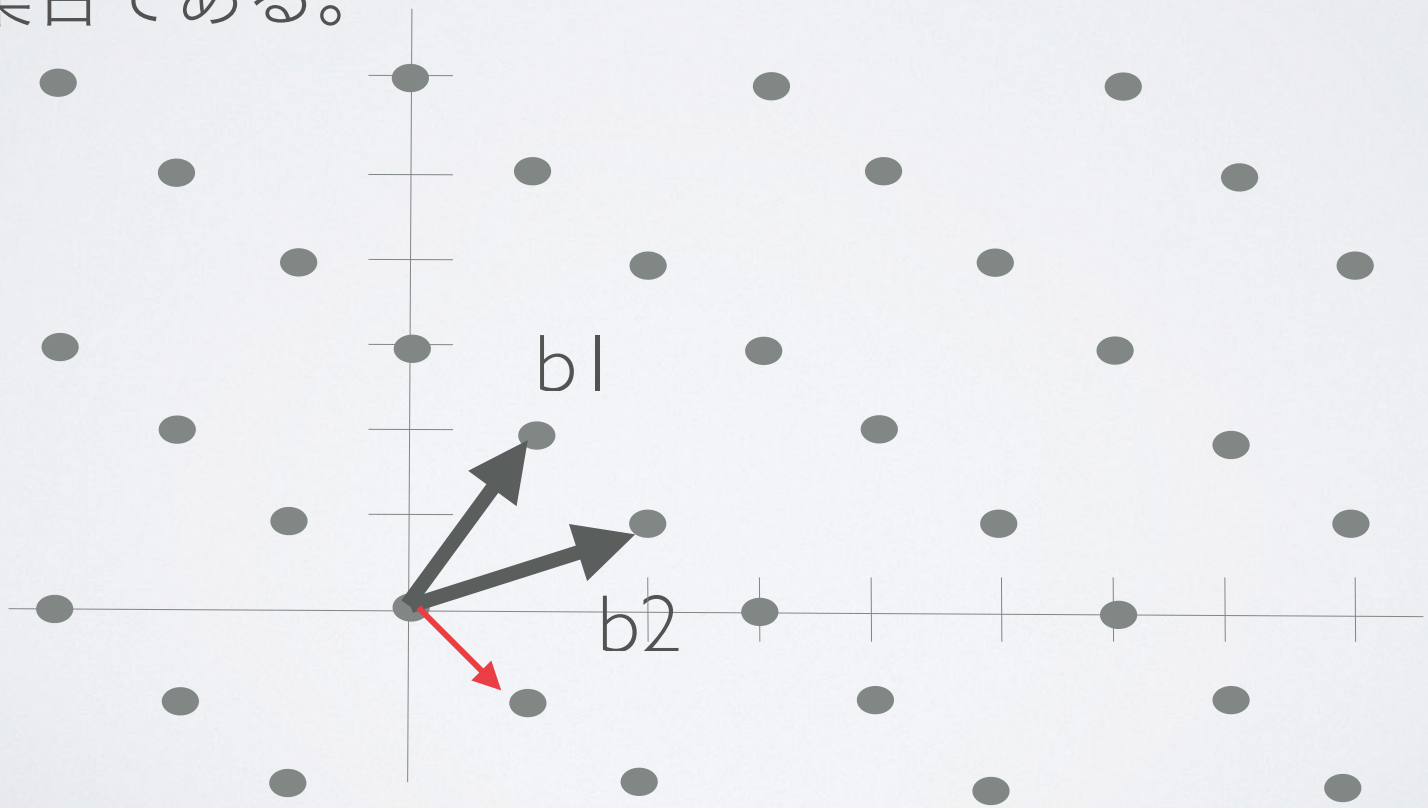
最短ベクトル問題とは

- 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。

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

本研究の手法と目的

- われわれのグループは、これまでsamplingでベクトル生成しながら、プロセス間で更新ベクトル情報を共有することにより、基底簡約を行ってきた。
- 本研究では、BKZアルゴリズムをプロセスごとに行い、更新ベクトル情報をプロセス間で共有することにより、基底簡約計算を高速化する。

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。
  - 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

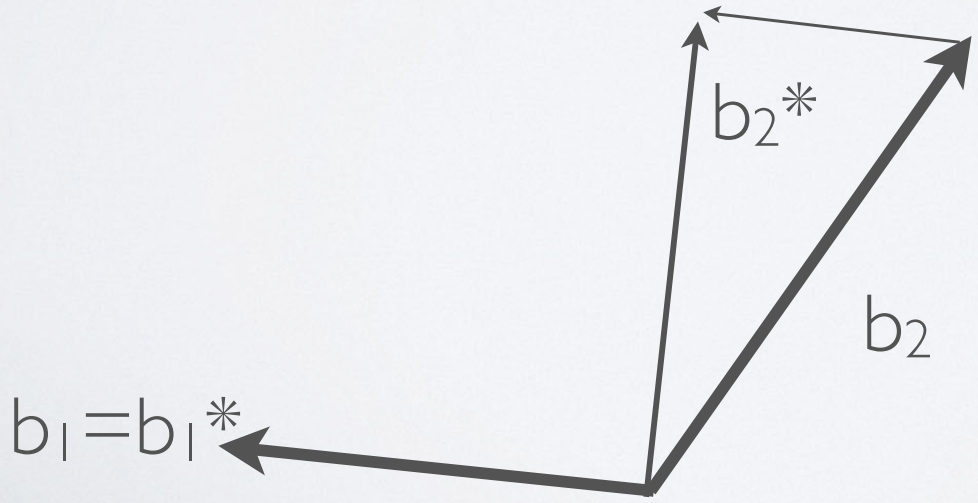
全511エントリー

基底変換

- 同じ格子を生成する基底行列は、無限個ある。
- 本発表では、格子を固定して考えるので、基底行列と呼んだ時には、同じ格子に対応するものを指す。
- 基底変換とは、基底行列を、(同じ格子を生成する)別の基底行列に変換すること。B=B’Uで、Uはユニモジュラー。
- （このあと出てくる）基底簡約となる基底変換が重要。

グラムシュミット直交化法

- 線形独立なベクトル系から、直交系を作り出す方法。
- 前のベクトルから順番に直交化していく。
- 基底ベクトルの順序に依存する。



グラムシュミット直交化法

- bi ：基底ベクトル　　b*i　：直交基底ベクトル
- i > j に対して、

$$\mu_{i,j} = \frac{< b_i, b_j^* >}{< b_j^*, b_j^* >}$$

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*$$

- Bを基底行列として、B=(B*)Mと表現できる。B*は直交行列。
- Mは、グラムシュミット係数でつくられる上三角行列(対角成分は、すべて1)

直交基底ベクトル

- bi ：基底ベクトル
- b*i：直交基底ベクトル
- b1　b2　b3　b4　b5　b6　b7　b8　b9
- b*1 b*2 b*3 b*4 b*5 b*6 b*7 b*8 b*9
  - b*i が、互いに直交している。
  - それぞれ先頭 k 個のベクトルで張る線形空間は等しい。
  - bi を b*i 方向に射影すると b*i と一致する。

直交基底ベクトルの長さの列

- 基底行列Bから直交行列B*を計算すると、各indexごとに直交基底ベクトルの長さが計算出来る。
- 直交基底ベクトル自体は、格子ベクトルではない。基底行列の簡約の度合いを測るのに長さの列を利用する。
- 実用上は、各直交基底ベクトルの長さを自乗した列を考える。$\{|b^*i|^2\}$

基底簡約

- 直交基底ベクトルの長さの列の辞書式順序を考えることにより、基底行列の長さ列全体は、 全順序になる。
- 基底簡約とは、直交基底ベクトルの長さの列の辞書式順序で、より小さい基底行列に基底変換すること。

基底簡約アルゴリズム

- 基底を入力として、ある程度、簡約された基底を出力してくれるアルゴリズムがいくつか知られている。
  - LLLアルゴリズム(Lenstra, Lenstra, Lovász, 1982)
  - BKZアルゴリズム(Schnorr, Euchner 1994)
  - RSRアルゴリズム(Schnorr 2003)

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。

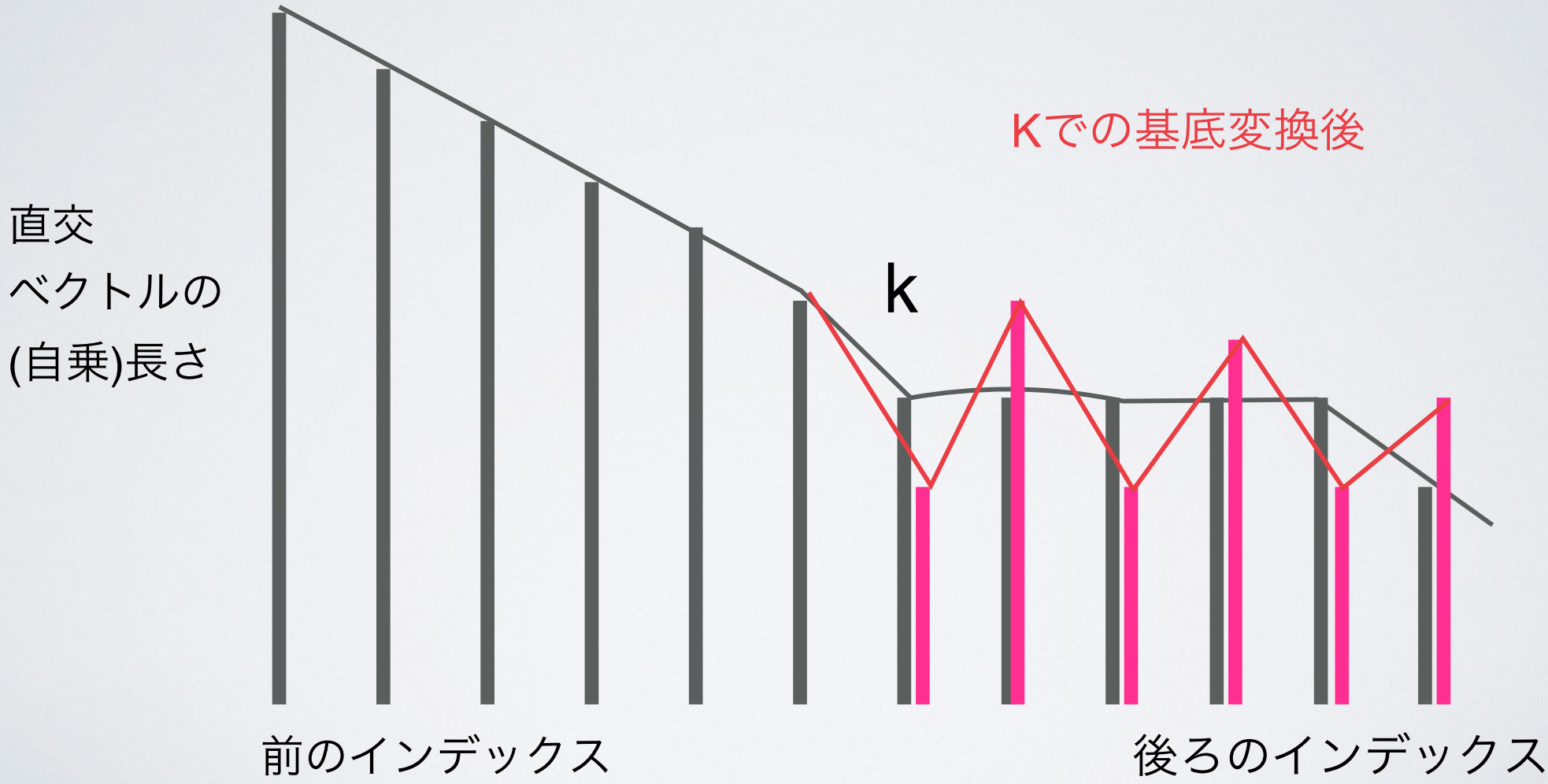
$$\pi_{k-1}(x) = \sum_{i=k}^{n} \nu_i b_i^*$$

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

基底簡約と長さ列

- k番目の直交ベクトルの長さを置き換える基底簡約では、直交基底ベクトルの長さの列 $\{|b^*_i|^2\}$ は、
  - k番目未満の部分の長さの列は、変化しない。
  - k番目の長さの値は、減少する。
  - k番目より上のindexについては、増えたり減ったり。
- すなわち、長さ列は辞書式に 改善していく。

ベクトル適用による変化



BKZとENUMアルゴリズム

- BKZアルゴリズムは、その部分アルゴリズムとしてENUMアルゴリズムを利用する。
- そもそもEnumerationは、与えられた長さ以下のすべてのベクトルを求める厳密解法アルゴリズム。
- BKZアルゴリズムで用いられるENUMアルゴリズムは、それを部分格子空間に適用したもの。
- 窓(部分空間として考えるindexの範囲)をいろいろ変えながら、全体的に直交基底ベクトルを短くしていく。

ENUMアルゴリズム

- 特定の連続するindexに注目し、その部分を窓とする。
- ********WWWWW*********
- そのindexに対応する直交基底ベクトルで貼られる直交部分空間を考える。
- なるべく短いベクトルを生成したい。
- 窓より、後ろの係数は0とする。窓に対応する部分の係数は、なるべく短くなるようにいろいろ考える。
- 窓より前のindexの係数は、Babai liftで作る。

BABAI LIFT

- 射影空間で短いベクトルに対して、全体長さが短くなるように、射影空間以外の次元の係数を変えて作ったベクトル
- 射影空間の部分を除いて、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。

並列計算

- 並列計算の方法には、大きくわけて、スレッド並列とプロセス並列がある。
- スレッド並列は、同一CPU内のコアなど、メモリが共有できるような状況で行われる。
- プロセス並列は、スーパーコンピュータのノード間など、メモリが共有できない状況で行われる。メモリアクセスより比較的時間の掛かるストレージは、共有できる。
- この研究では、プロセス並列により、並列化を行う。

BKZとプロセス並列

- BKZアルゴリズムの並列化は、ENUMの部分の仕事を分割して、並列化するものが研究されていた。
- 今回は、プロセス並列で、プロセスそれぞれが異なる基底を持つものを考える。ただし、前のほうの基底ベクトルは共通である。
- どのプロセスも同じ基底が入力される。プロセス番号により、微妙に動作を変える。

プロセス並列とリンクベクトル

- 前のほうの基底ベクトルを共通化するために、リンクベクトルというものを考える。ストレージに保存される。
- リンクベクトルにより、それぞれのプロセスが協調する。

ENUMとSIEVING

- BKZアルゴリズムでは、ベクトル生成の方法として、ENUMという手法が用いられてきた。
- 本研究では、必要に応じて、Sievingと呼ばれる方法も併用する。

SIEVINGとは

- 既知の短いベクトルを足し引きすることで、短いベクトルを探す。
- もともとのベクトルが短いほど、短いベクトルが見つかりやすい。
- 短いベクトルの集合を短いベクトルに置き換えていく。
- 次元が高いと、短いベクトルを探すのに大量の既知のベクトルが必要。

本研究の目指すもののまとめ

- 以下の手法で基底簡約計算を高速化する。
  - BKZアルゴリズムをプロセスごとに行う。
  - 更新ベクトル情報を並列プロセス間で共有する。
  - 場合によっては、sievingも利用する。

おわり

- ご清聴ありがとうございました。