# 格子暗号の安全性を検証する最短ベクトル問題に対する解読システムの開発

**立岩 斉明**[1], 品野 勇治[2], 山村圭一郎[1], 吉田 明広[1]
鍛冶 静雄[1], 安田 雅哉[3], **藤澤 克樹**[1]

[1] 九州大学
[2] Zuse Institute Berlin
[3] 立教大学

2021/5/14
先駆的科学計算に関するフォーラム2021
@オンライン

# 目次

1. 研究背景
   ：最適化問題やグラフ解析, 格子暗号, 格子, 最短ベクトル問題(SVP)

2. 並列システム
   ：格子, 格子アルゴリズムの性質, 実装

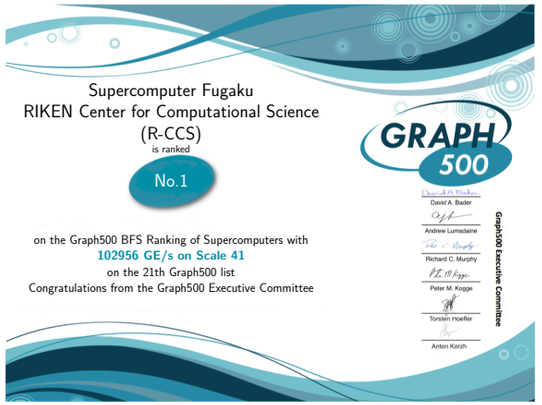
3. 数値実験
   ：SVP Challenge, 求解性能評価, 通信遅延評価

4. 結論

# 研究背景

## 数理最適化及びグラフ解析への HPC 適用

- 大規模グラフ解析及び数理最適化システムの開発と評価
  - **Graph500 ベンチマーク 通算12期：世界第1位 (京及び富岳, 2014年～2020年)**
  - **大規模最適化問題**(半正定計画問題)に対する大規模 GPU 活用 (**SC12, IPDPS14**)
  - 格子暗号：**最短ベクトル問題 (SC20：本発表)**





## Society 5.0 と産業アプリケーション

- **Society5.0 などが目指す超スマート社会**に向けて，道路や電力網などのインフラを含めた都市計画への活用から、ヒト・モノの動きの解析まで**多岐にわたる課題を解決**
- 民間企業との協業で大量のセンサーデータやオープンデータなどを用いて、サイバー空間での最適化やシミュレーションを行う**産業アプリケーションの開発を推進**

# 研究背景

## 量子計算機/アルゴリズムの台頭

- Shorのアルゴリズムに代表される**量子アルゴリズム**
  及び **大規模量子計算機**の開発により
  広く普及している**RSA暗号**や**楕円曲線暗号**の効率的な解読が危惧されている.
  (国家機密であれば, 40~50年ほどの期間, 情報が守られている必要がある)

## 耐量子計算機暗号の標準化

- そこで量子計算機でも効率的に解読できない
  **耐量子計算機暗号**の選定が進められている (Post-Quantum Cryptography Standardization[1])
- **格子問題**の求解困難性を拠り所にした**格子暗号**が有力な候補の一つ
  その上で格子暗号の
  1. **安全性の検証**
  2. **適切な公開鍵サイズの検証**

  が急務になっている

[1]https://csrc.nist.gov/projects/post-quantum-cryptography

# 研究背景

## 最短ベクトル問題(SVP)

- 多くの格子暗号の安全性を支えている問題
- さまざまな求解アルゴリズムが考案されているが, 決定的なものはない
- さらに, **大規模計算機を用いた求解実験**はなく,
  大規模並列化による求解困難性の評価が未検証であった

**研究目標**

**最短ベクトル問題の大規模並列ソフトウェアの開発**
及び, それを用いた最短ベクトル問題の求解困難性の検証

# Contribution

**MAP-SVP**: the ***Ma****ssive* ***P****arallel Solver for* ***S****hortest* ***V****ector* ***P****roblem*
の実装とその評価

- ✓ 分散で非同期的な SVP ソルバ
- ✓ 安定的なMPI並列実行が可能(最大103,680 coresを用いた実行実績)
- ✓ SVP求解ソフトのコンペティションである
  SVP Challenge[1]にていくつかの記録を更新

- ✓ 研究成果をまとめた論文は
  HPC分野のトップカンファレンスである SC20 に採択[2]
  The International Conference for High Performance Computing, Networking, Storage, and Analysis

[1]https://www.latticechallenge.org/svp-challenge/

[2]Tateiwa, Nariaki, et al. "Massive parallelization for finding shortest lattice vectors based on ubiquity generator framework." *2020 SC20: International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*. IEEE Computer Society, 2020.

# 研究内容

## MAP-SVP: the *Massive Parallel Solver for Shortest Vector Problem*

**理論**

**格子アルゴリズム**

1. 既存のアルゴリズムの並列計算用拡張

2. 効率的な求解を実現するためのシステム設計

3. C++ / MPI によるソフトウェア実装

**実装**

**大規模並列化技術**

4. スパコンを用いた大規模実験と評価

立教大学
安田研

ZIB研究所(ドイツ)
品野研究員

九州大学
藤澤研
鍛冶研

# 最短ベクトル問題 (SVP)

## 格子(lattice)

**$n$次元格子**は, 一次独立な$n$本の
ベクトル $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$の整数係数による
足し合わせで生成されるベクトルから
構成される集合(網目)

$$\mathcal{L}(\mathbf{B}) = \left\{ \sum_{i=1}^{n} a_i \mathbf{b}_i ; \ \ a_i \in \mathbb{Z} \right\}$$

このとき
$\mathbf{B}$ を **格子基底** と呼ぶ

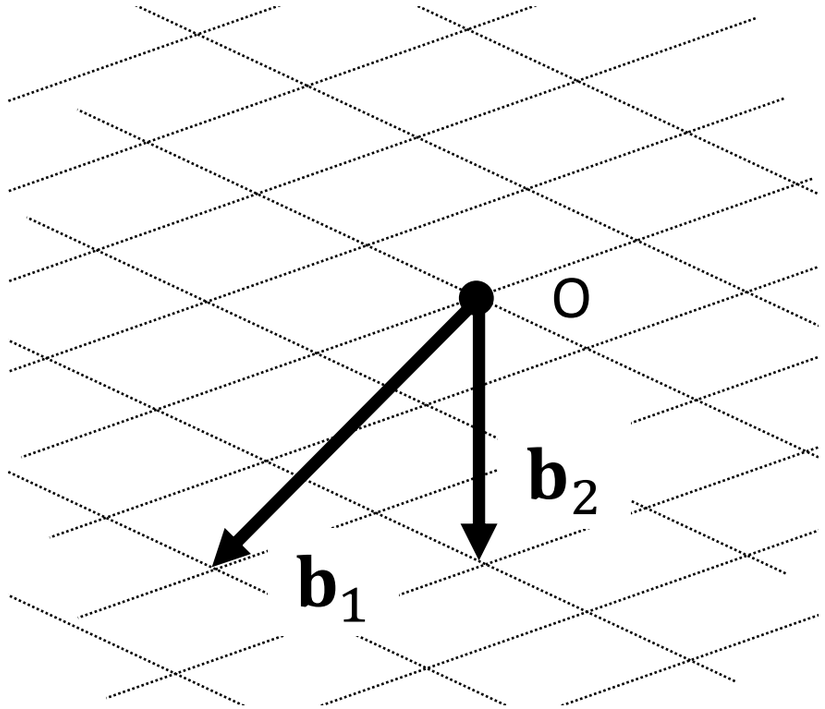



図: 2次元格子

# 最短ベクトル問題 (SVP)

## 最短ベクトル問題

最短ベクトル問題は,
の原点に最も近い格子点を求める問題

性質:
- ある仮定のもとでNP-Hard
- 格子の次元が大きくなるほど, 求解困難性が増大

**SVP**

$$\begin{aligned} &\text{minimize} \quad \|\mathbf{v}\| \\ &\text{subject to} \quad \mathbf{v} \in \mathcal{L}(\mathbf{B}) \setminus \{\mathbf{0}\} \end{aligned}$$

$$\mathcal{L}(\mathbf{B}) = \left\{ \sum_{i=1}^{n} a_i \mathbf{b}_i ; \ \ a_i \in \mathbb{Z} \right\}$$

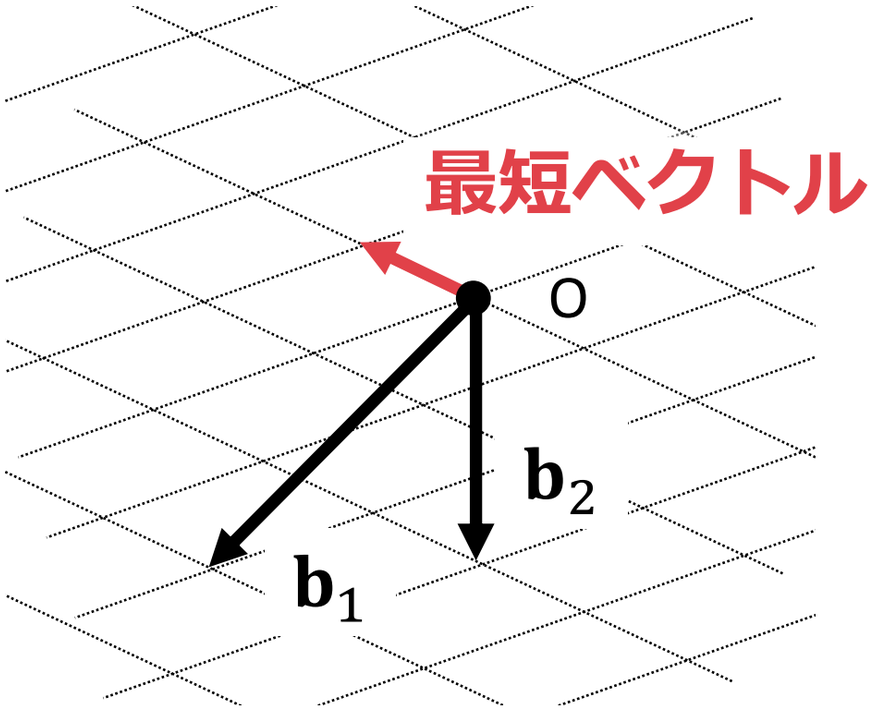



図: 2次元格子と最短ベクトル

# 最短ベクトル問題 (SVP)

## 最短ベクトル問題

最短ベクトル問題は,
の原点に最も近い格子点を求める問題

性質:
- ある仮定のもとでNP-Hard
- 格子の次元が大きくなるほど, 求解困難性が増大

**SVP**

$$\begin{aligned} &\text{minimize} \quad \|\mathbf{v}\| \\ &\text{subject to} \quad \mathbf{v} \in \mathcal{L}(\mathbf{B}) \setminus \{\mathbf{0}\} \end{aligned}$$

$$\mathcal{L}(\mathbf{B}) = \left\{ \sum_{i=1}^{n} a_i \mathbf{b}_i ; \ \ a_i \in \mathbb{Z} \right\}$$

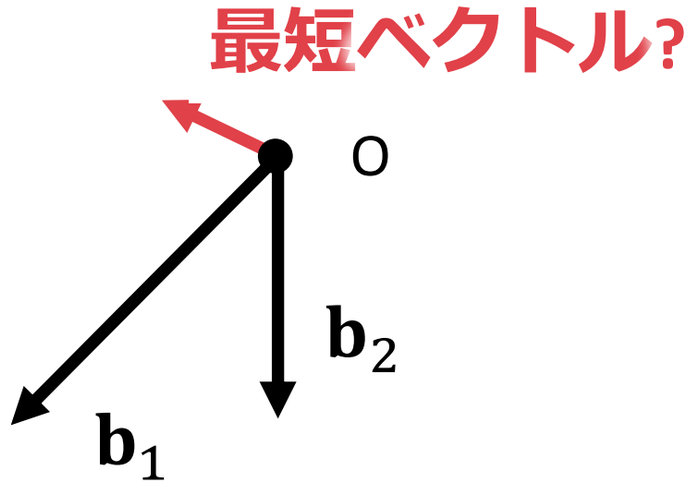
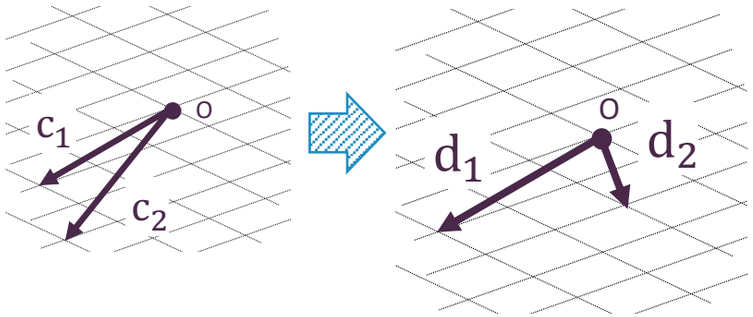



図: 2次元格子と最短ベクトル

# 代表的な格子アルゴリズム

## DeepBKZ (近似的)

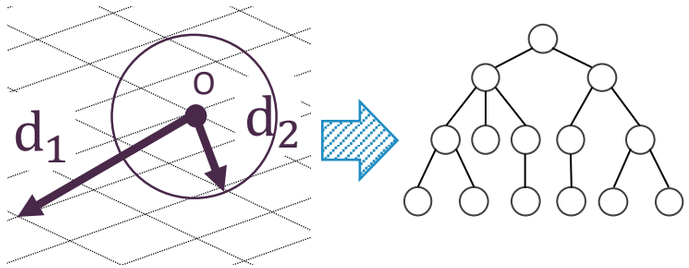



格子基底がなるべく直交に
近くなるように行列の基本変形を行う

アルゴリズム単体での並列化は?

- 高々200次元の行列の逐次的な行列変形であるため並列効果は極小

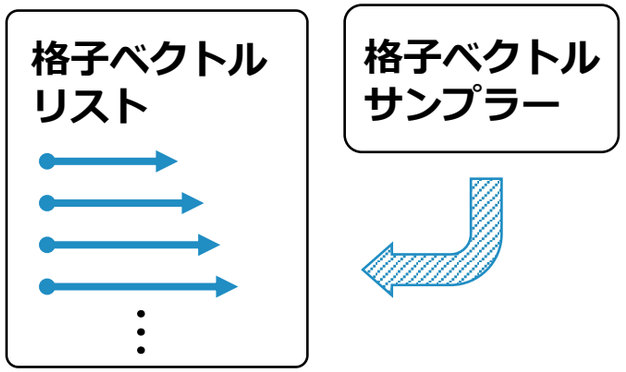
## ENUM (厳密的)




最短ベクトルを含む探索領域を
木で表現し, 深さ優先探索を行う

- 探索木の分割により, 並列数に対して線形の効果の並列化が可能
- だが, もともと探索木が次元に対し**指数的に**極大なため, 探索できる範囲はそれでも極小

## Sieve (確率的)

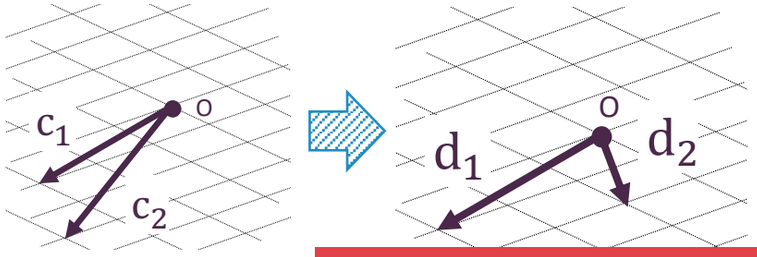



サンプリングした格子ベクトル
同士の足し引きにより
ベクトルを短く変換する

- 共有メモリシステムでの並列手法の提案がある
- が, 次元に対し**指数的な**サイズのベクトルの保持が必要なため, メモリ確保がボトルネックに

# 代表的な格子アルゴリズム

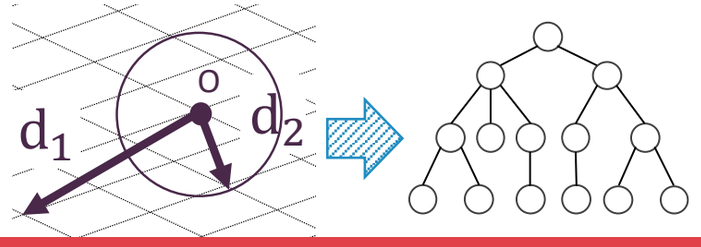
## DeepBKZ (近似的)




格子基底がなる
近くなるように

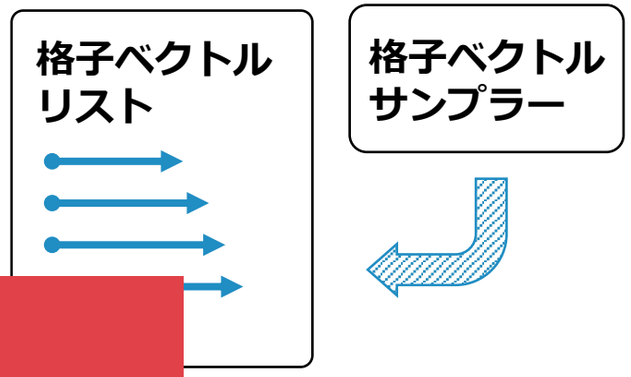
- 高々200次元の行列の逐次的な行列変形であるため並列効果は極小

## ENUM (厳密的)




- 探索木の分割により, 並列数に対して線形の効果の並列化が可能
- だが, もともと探索木が次元に対し**指数的に**極大なため, 探索できる範囲はそれでも極小

## Sieve (確率的)




ングした格子ベクトル
し引きにより
を短く変換する

- 共有メモリシステムでの並列手法の提案がある
- が, 次元に対し**指数的な**サイズのベクトルの保持が必要なため, メモリ確保がボトルネックに

**アルゴリズム単体の並列化だけでは,**
**大規模計算機を十分に活用できない**

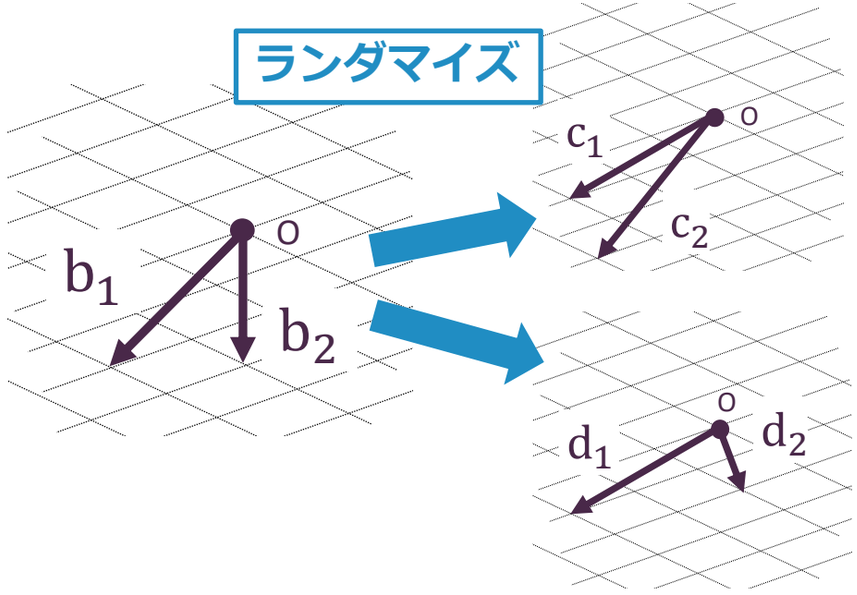
アルゴリズム
単体での
並列化は?

# 格子の性質

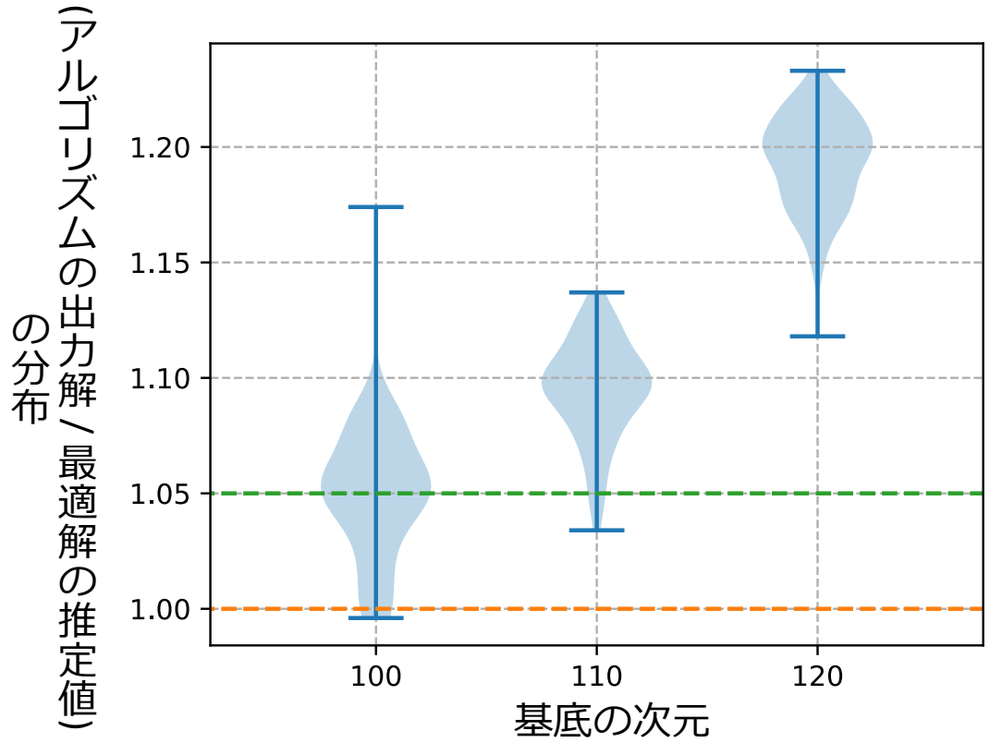
## 格子基底のランダマイズ

$\mathcal{L}(\mathbf{B}) = \mathcal{L}(\mathbf{UB})$ $\forall \mathbf{U}$: ユニモジュラ行列
$(\mathcal{L}(\mathbf{B}) \coloneqq \{\sum_{i=1}^{n} a_i \mathbf{b}_i;\ a_i \in \mathbb{Z}\})$

ユニモジュラ行列を基底に作用させても
生成される格子は不変
→ 無限個の異なる問題表現が可能




- **入力基底が変わることで, アルゴリズムの挙動も変化**
- 例えば, 128個のランダム基底に対してDeepBKZを実行した場合
  アルゴリズムが見つけた最短ベクトルの長さにも
  下図のような, ばらつきが見られた



# 格子の性質

## 格子アルゴリズムの副産物

格子アルゴリズムは, 最短ベクトルの
探索の過程において

- **最短ではないが十分短いベクトル**

などの副産物が得られるが,
それらをDeepBKZやENUMアルゴリズムに対し
効果的に活用したSVP求解ソルバはなかった

→ 活用するとどうなる?

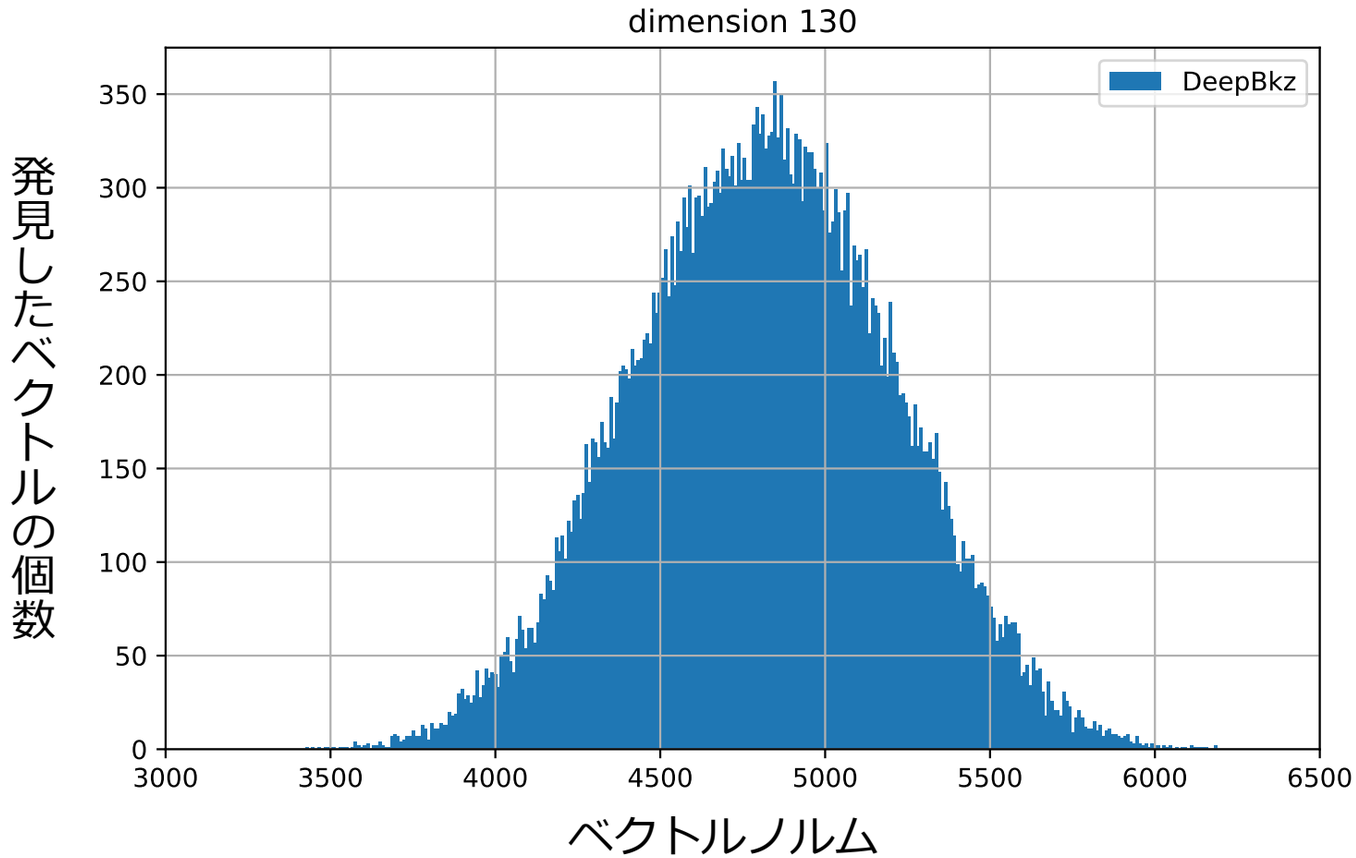
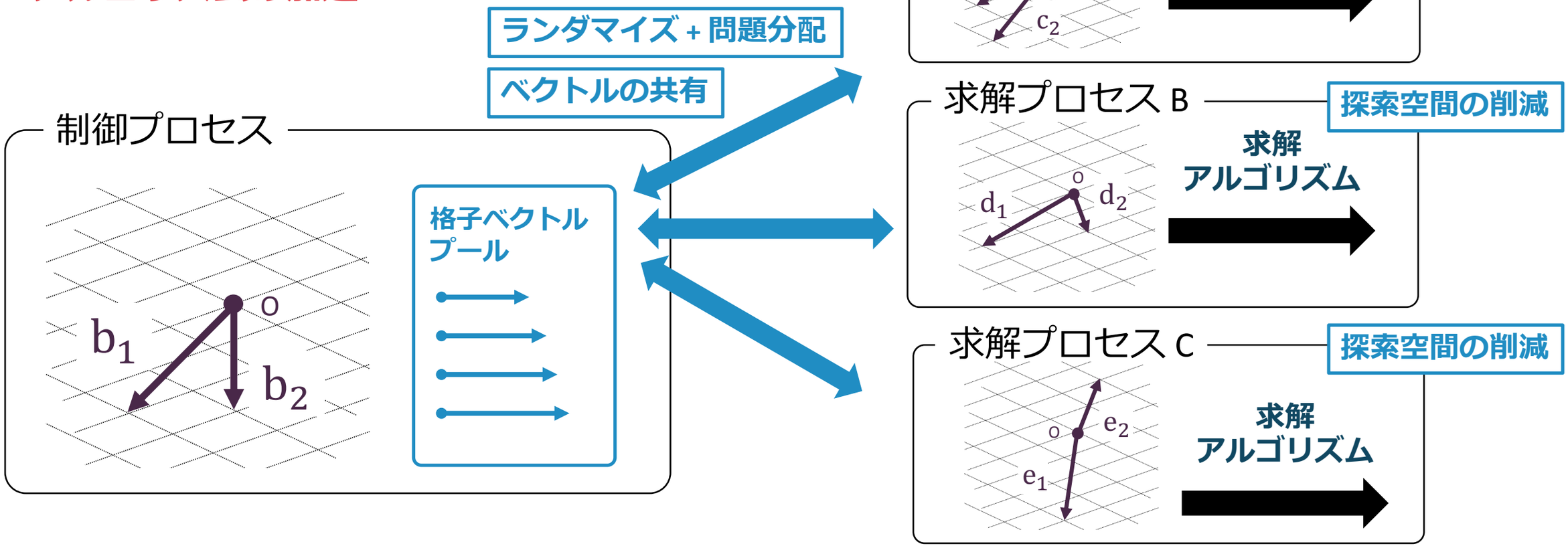



図: DeepBKZアルゴリズムが実行中に
発見する格子ベクトルの分布図

# 並列システム

- ランダマイズによる, 別表現の問題の分配
- 並列で競争的な実行
- 非同期での格子ベクトルの共有による
  アルゴリズムの加速

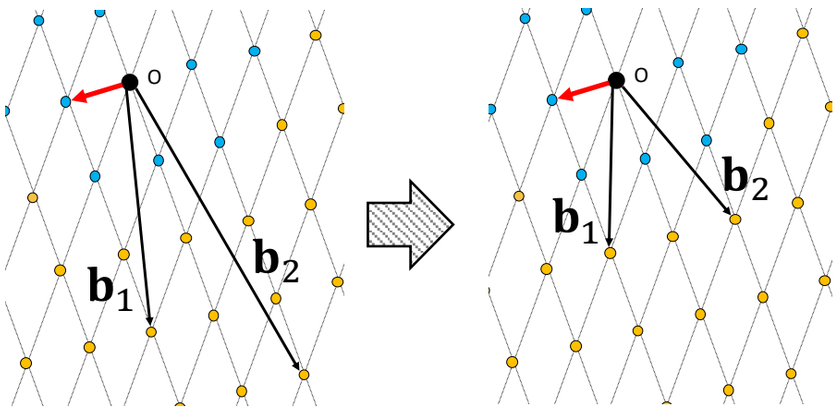


# 並列システム

短いベクトル共有によるアルゴリズムの効率化

## DeepBKZ (近似的)

アルゴリズム内の一部サブルーチンを
“外部から短いベクトルを取得する”ことで
より, 基底ベクトルの直交化を促進できる
→ **アルゴリズムの強力化**




## ENUM (厳密的)

現在見つかっている最短ベクトルノルムを
用いることで探索空間をより小さくできる
→ **探索時間の削減**

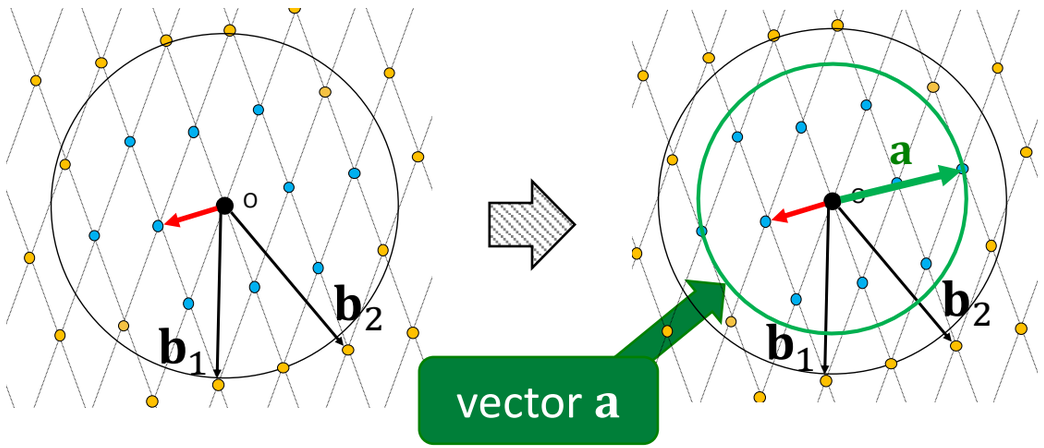



図: ENUMにおける探索領域削減のイメージ
(探索領域を擬似的に円で表現)

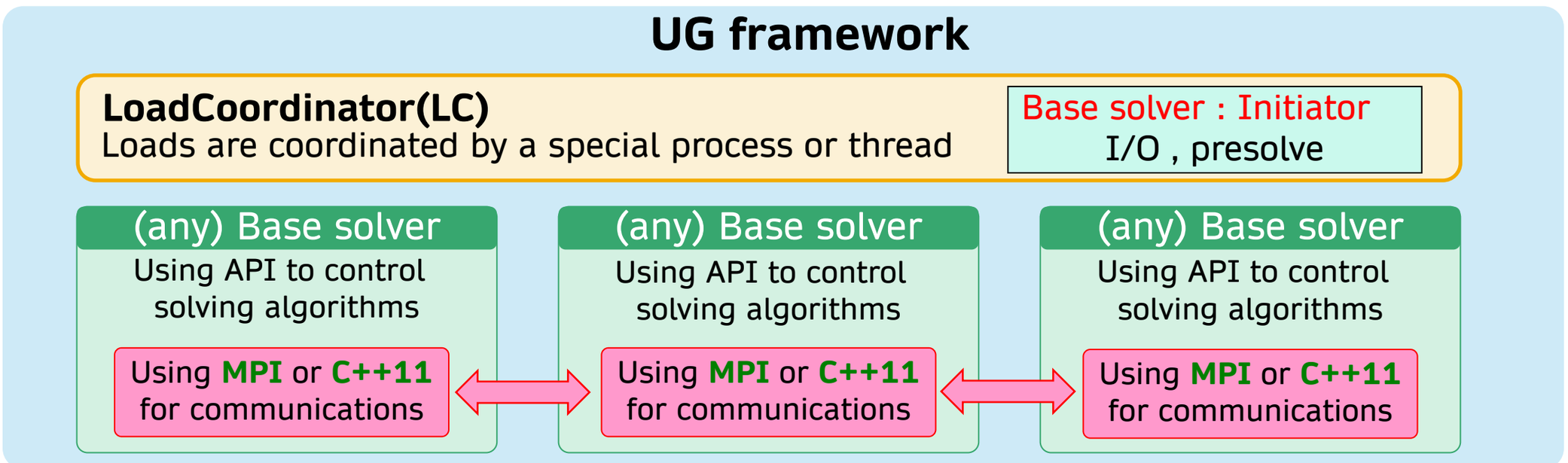
# 実装

## Ubiquity Generator (UG) Framework

- 分枝限定B&Bソルバ(SCIP, CPLEX, Xpress, …)を**外側から**並列化する汎用分散並列ソフトウェア
- 制御プロセス(LC)と複数のSolverから構成される
- LCはSolverに渡す問題の分配と動的管理を行い，**Solverは与えられた問題に対し基本的に独立に動作**

UG
Ubiquity Generator framework

https://ug.zib.de

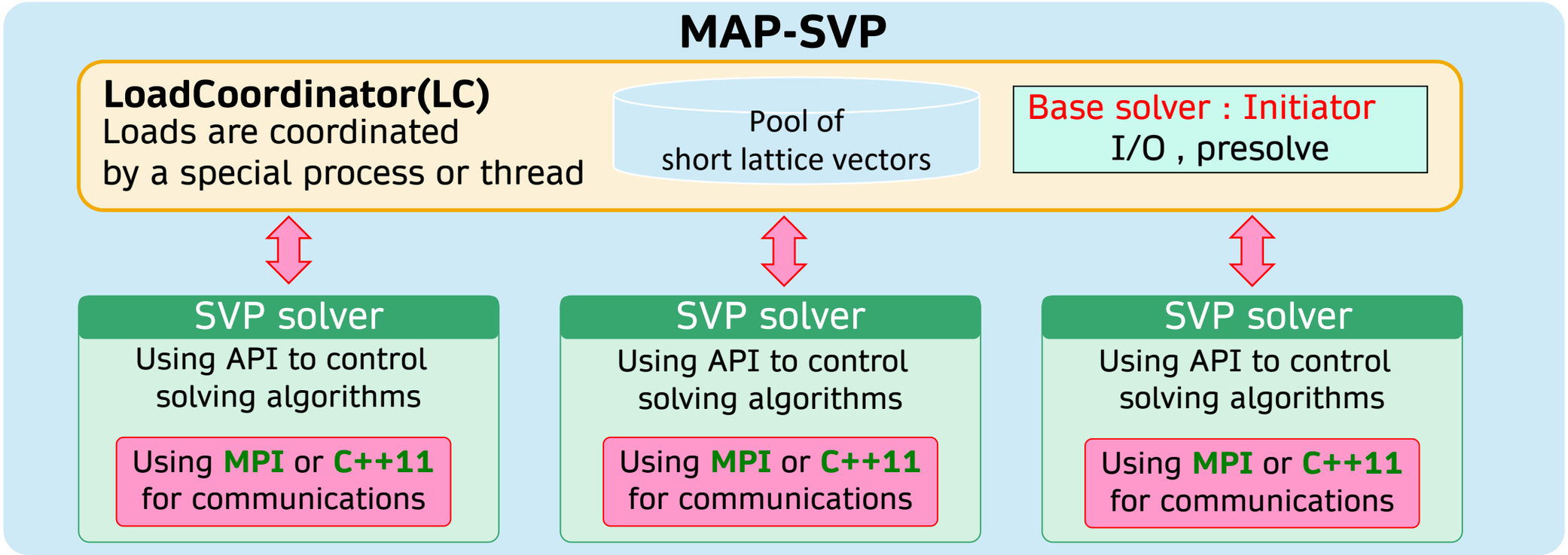


# 実装

UGを継承して SVP用の求解ソフトウェアを開発

- UGの通信機能に加え, ベクトル共有の通信関数を追加

1. 管理プロセス(LC)はSolverにランダマイズ問題を分配(MPI)
2. Solverは問題を受け取りDeepBKZとENUMを実行
3. Solverは定期的にLCに短いベクトルの送受信を行う(MPI)



# The Darmstadt SVP Challenge

## SVP求解ソフトウェアのコンペティション

**$\alpha$-approximate SVP**

finding $\quad \mathbf{v} \in \mathcal{L}(\mathbf{B}) \setminus \{\mathbf{0}\}$

subject to $\|\mathbf{v}\| \leq \alpha \, \lambda_1(\mathcal{L}(\mathbf{B}))$

格子$\mathcal{L}(\mathbf{B})$における最短ベクトルノルム

SVP Challengeでは

- $\alpha$ = 1.05
- $\lambda_1(\mathcal{L}(\mathbf{B})) :=$ **GH**
  (最短ベクトルノルムの推定値)

が使用される

次元ごとに記録が管理され
1. 上記の条件を満たし
2. 既存記録よりも短い

ベクトルであればサイトに登録される

HALL OF FAME

| Position | Dimension | Euclidean Norm | Seed | Contestant | Solution | Algorithm | Subm. Date | Approx. Factor |
|---|---|---|---|---|---|---|---|---|
| 1 | 180 | 3509 | 0 | L. Ducas, M. Stevens, W. van Woerden | vec | Sieving | 2021-02-8 | 1.04002 |
| 2 | 178 | 3447 | 0 | L. Ducas, M. Stevens, W. van Woerden | vec | Sieving | 2021-02-8 | 1.02725 |
| 3 | 176 | 3487 | 0 | L. Ducas, M. Stevens, W. van Woerden | vec | Sieving | 2020-10-13 | 1.04411 |
| 4 | 170 | 3438 | 0 | L. Ducas, M. Stevens, W. van Woerden | vec | Sieving | 2020-05-12 | 1.04690 |
| 5 | 158 | 3240 | 0 | Sho Hasegawa, Yuntao Wang, Eiichiro Fujisaki | vec | Sieving | 2021-01-22 | 1.02311 |
| 6 | 157 | 3320 | 0 | L. Ducas, M. Stevens, W. van Woerden | vec | Sieving | 2019-05-20 | 1.04906 |
| 7 | 156 | 3219 | 0 | Sho Hasegawa, Yuntao Wang, Eiichiro Fujisaki | vec | Sieving | 2021-01-22 | 1.01986 |
| 8 | 155 | 3165 | 0 | M. Albrecht, L. Ducas, G. Herold, E. Kirshanova, E. Postlethwaite, M. Stevens, P. Karpman | vec | Sieving | 2018-09-18 | 1.00803 |

- 現在の最高記録は
  180次元の記録 (2021年2月8日).
  ただ, $\|\mathbf{v}\|$ / GH は1.04であり,
  最短ベクトルの発見には至っていない

# 計算機

TABLE I
COMPUTING PLATFORMS USED

| Machine | Memory / node | CPU | CPU frequency | # of nodes | # of cores |
|---|---|---|---|---|---|
| HLRN IV | 384 GB | Intel Xeon Platinum 9242 (CLX-AP) | 2.30 GHz | 1,042 | 100,032 (96 × 1,042) |
| ISM | 384 GB | Intel Xeon Gold 6154 | 3.00 GHz | 144 | 5,184 (36 × 144) |
| ITO | 192 GB | Intel Xeon Gold 6154 (Skylake-SP) | 3.00 GHz | 128 | 72,000 (36 × 2000) |
| CAL A | 32 GB | Intel(R) Xeon(R) CPU E3-1284L v3 | 1.80 GHz | 45 | 180 (4 × 45) |
| CAL B | 256 GB | Intel(R) Xeon(R) CPU E5-2640 v3 | 2.60 GHz | 4 | 64 (16 × 4) |
| CAL C | 256 GB | Intel(R) Xeon(R) CPU E5-2650 v3 | 2.30 GHz | 4 | 80 (20 × 4) |

以降の実験は下記の設定で行っている.

- MPIプロセスのみを使用.
- 1コアに1プロセスを割り当てる(ハイパースレッディング機能は使用しない).
- LC と Solver は1つのプロセスに割り当てる

# SVP Challengeでの記録更新

## 複数の次元で記録を更新

表 2　MAP-SVP による SVP チャレンジ の新記録

| 次元 | シード | Norm | 近似係数 | #Process | Total time |
|---|---|---|---|---|---|
| 104 | 35 | 2516 | 0.97173 | 120 | 551 seconds |
| | 85 | 2520 | 0.97010 | 120 | 214 seconds |
| | 82 | 2529 | 0.97719 | 120 | 432 seconds |
| 111 | 29 | 2597 | 0.96979 | 2000 | 792 seconds |
| | 30 | 2635 | 0.98382 | 2000 | 541 seconds |
| | 8 | 2660 | 0.99467 | 2000 | 611 seconds |
| 121 | 4 | 2780 | 0.99706 | 2304 | 682 minutes |
| | 2 | 2809 | 1.00820 | 2304 | 481 minutes |
| 127 | 3* | 2790 | 0.97573 | 91,200 | 147 hours |
| 127 | 1† | 2890 | 1.01429 | 9,980 | 31 hours |
| 130 | 1† | 2935 | 1.01923 | 103,680 | 180 hours |

表 3　127 次元, シード 3 の SVP チャレンジの実行履歴

| try | Norm | 近似係数 | #Process | Wall Time | Machine |
|---|---|---|---|---|---|
| 1 | 3186 | 1.11435 | 4,608 | 6 hours | ITO |
| 2 | 3186 | 1.11435 | 180 | 11 hours | CAL B, C |
| 3 | 3037 | 1.06218 | 4,608 | 6 hours | ITO |
| 4 | 2956 | 1.03397 | 4,608 | 6 hours | ITO |
| 5 | 2956 | 1.03397 | 49,152 | 12 hours | HLRN IV |
| 6 | 2922 | 1.02202 | 5,184 | 100 hours | ISM |
| 7 | 2790 | 0.97573 | 91,200 | 6.3 hours | HLRN IV |
| Total | 2790 | 0.97573 | | ≈ 147 hours | |

* † ベクトル取得のために,複数の計算機で複数回MAP-SVPを実行した. この表には,その中の最大のプロセス数とwall timeの合計値の概算を記載している.
† これらの解は新記録ではないが, 既存記録と同じ,もしくはそれに十分に近いものであるため記載した.

**checkpoint and Restart**

**DeepBKZ**　基底を保存

**ENUM**　探索ノードの保存

# SVP Challengeでの記録更新

## 130次元の求解報告

130次元の現在記録はノルム2870, 近似係数 0.99413

表 4　130 次元, シード 1 の SVP チャレンジの実行履歴

| try | Norm | 近似係数 | #Process | Wall Time | Machine |
|---|---|---|---|---|---|
| 1 | 2994 | 1.03979 | 103,680 | 24 hours | HLRN IV |
| 2 | 2994 | 1.03979 | 4,608 | 30 hours | ITO |
| 3 | 2974 | 1.03275 | 70,200 | 10 hours | ITO |
| 4 | 2974 | 1.03275 | 103,584 | 12 hours | HLRN IV |
| 5 | 2935 | 1.01923 | 103,680 | 14 hours | HLRN IV |
| 6 | 2935 | 1.01923 | 4,608 | 36 hours | ITO |
| 7 | 2935 | 1.01923 | 103,680 | 24 hours | HLRN IV |
| 8 | 2935 | 1.01923 | 4,608 | 30 hours | ITO |
| Total | 2935 | 1.01923 | | ≈ 180 hours | |

# 数値実験

## 並列数の増加による性能向上

- **Root Hermite Factor** はDeepBKZなどの簡約アルゴリズムの質を測る資料

$$\gamma^{1/n} := \left(\frac{\|\mathbf{b}\|}{\mathrm{vol}(L)^{1/n}}\right)^{1/n}$$

ここで $\mathbf{b}$ は簡約アルゴリズムが出力する最短ベクトル

- $\gamma^{1/n}$が小さいほど, より短いベクトルが発見できている

- 100,032プロセスでの大規模であっても性能向上(スケーラビリティ)は保たれている

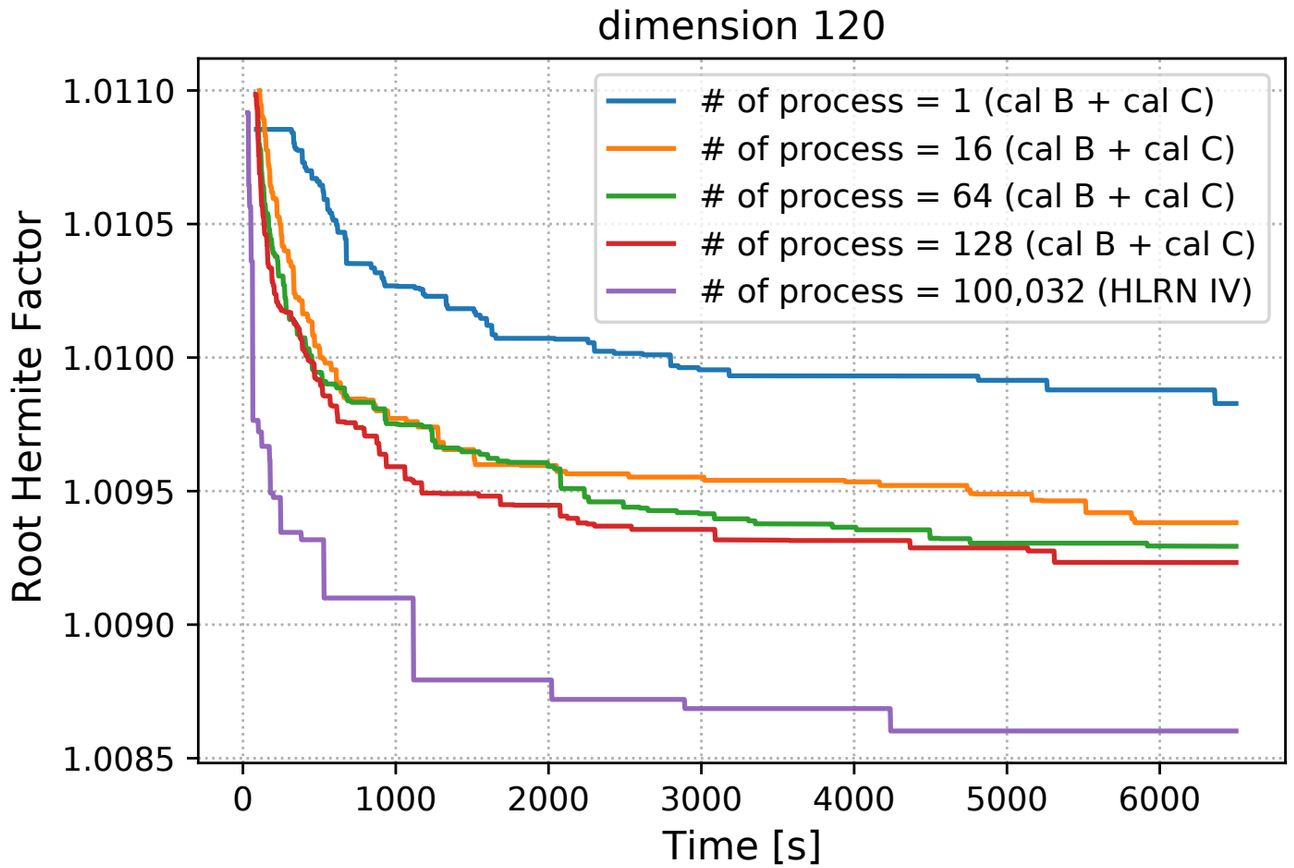



図: 120次元のSVPに関する平均Root Hermite Factorの推移

# 数値実験

## 並列数の増加による推定計算時間の減少

- ENUMアルゴリズムの探索木のサイズの推定式からENUMの探索時間を推定
- 並列数が増加するほど,より強く探索空間を削減される
- 実際に並列数が増加するほど,探索時間が減少していることを確認

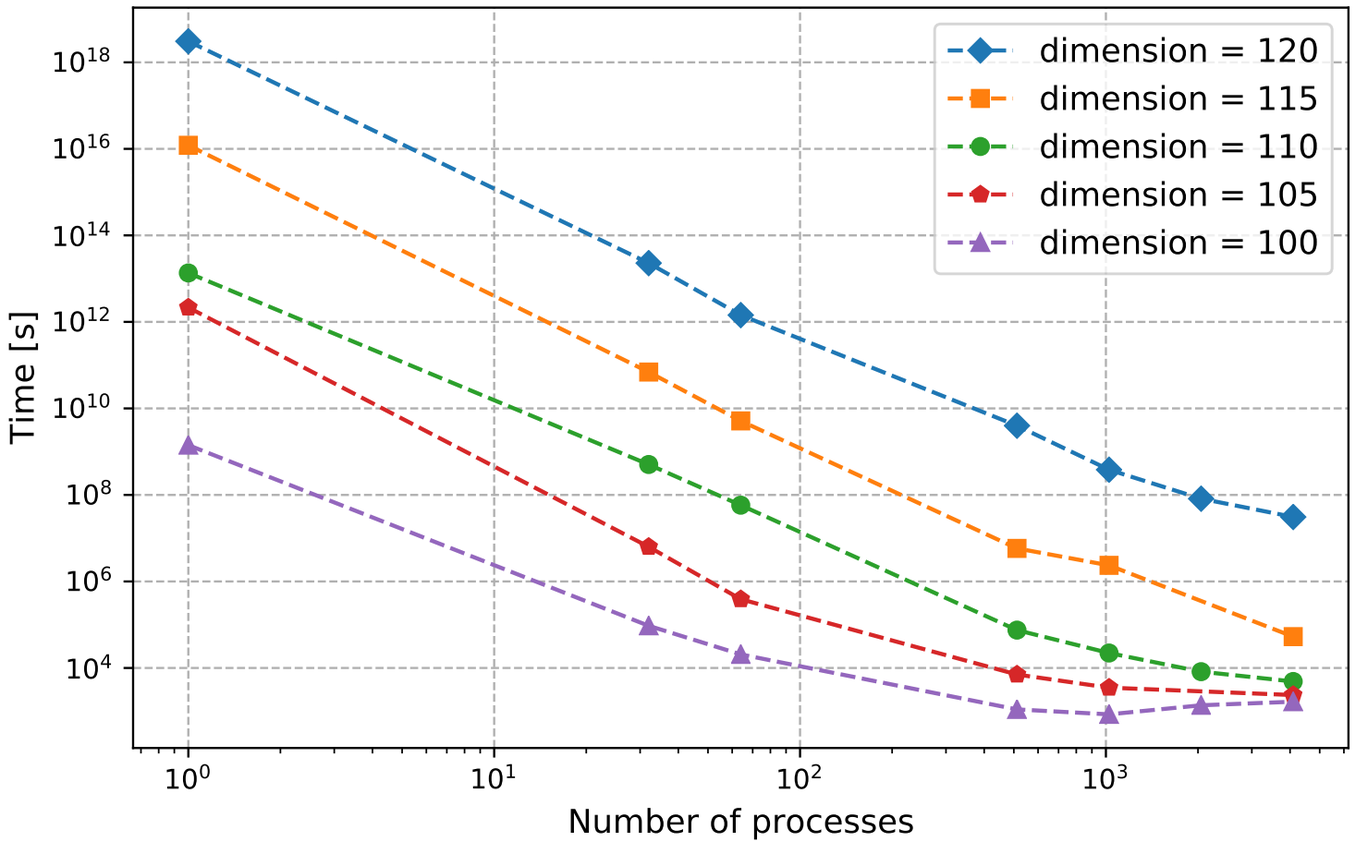
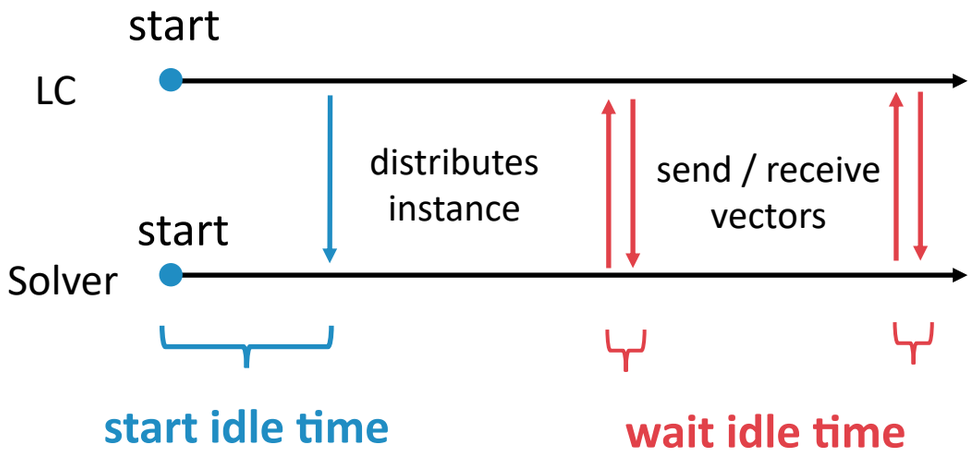



図: 0.99の確率で最短ベクトルが求められる推定時間
(ENUMアルゴリズムの探索時間から推定)

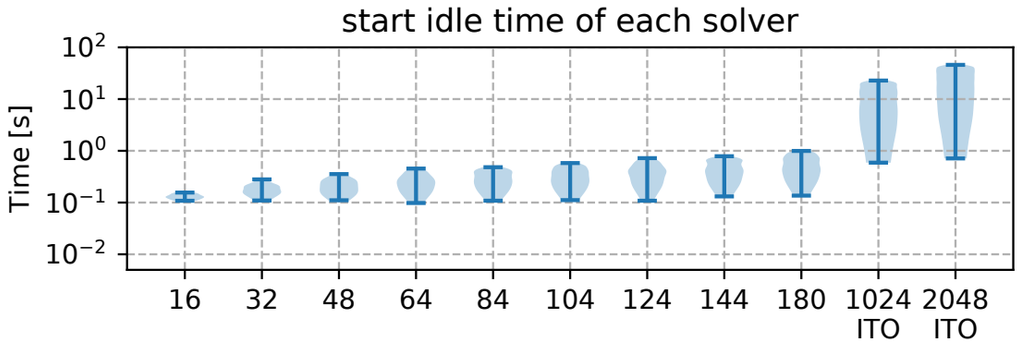
# 数値実験

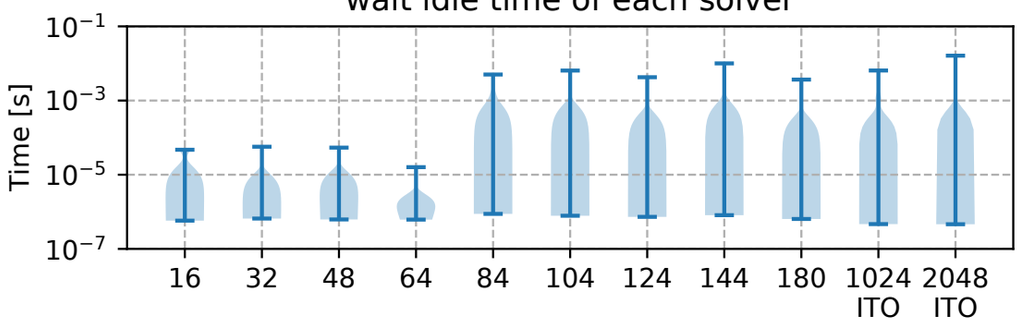
## コミュニケーションの小遅延

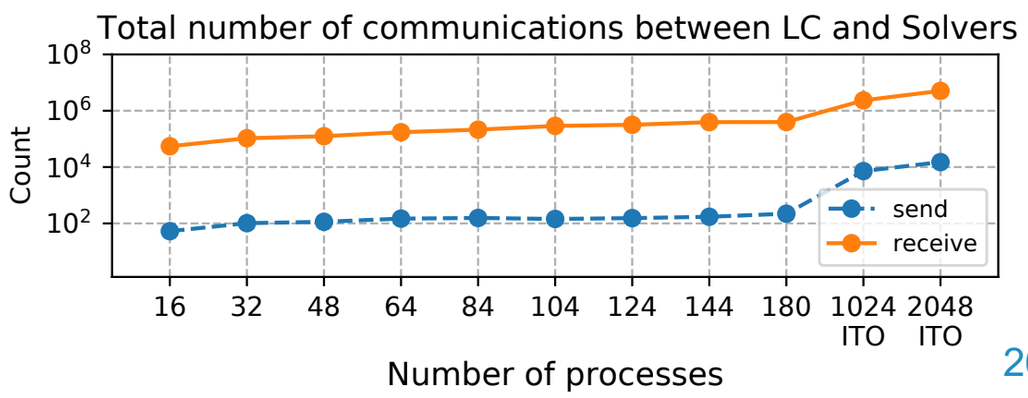



- 100次元のSVPを1時間実行した場合の通信遅延の調査

- プロセス数が増加するほどLCの負荷(メッセージの総送受信)も増加するが,非同期で通信するため, 局所的な集中は少ない
- 実際, 通信の遅延は1時間の実行時間に比べてはるかに少ない



## Conclusion

- UGフレームワークをベースに新しいSVPソルバである
  初の非同期協調分散型システムMAP-SVPを開発した.
- 数値実験により MAP-SVPを用いて SVP Challenge において
  最高 127 次元の記録の更新に成功した.
  これは現状厳密解法アルゴリズムによるソルバで達成された
  最大次元の記録である.
- また, 130次元の求解において, 103,680 processでの並列実行に成功した

## Future Work

- 本稿での研究成果を踏まえて,
  UG自体を分枝限定法以外のソルバも扱えるように汎用化し,
  MAP-SVPをスクラッチから再設計する
- Solverが実行するアルゴリズムを柔軟に変えられるように変更
  → HeterogeneousなSolverへ
- 将来的に本稿のアルゴリズムを含むより強力なSolverのコードを公開する予定

# 学会, 研究会発表

## 国際学会

- Tateiwa, N., Shinano, Y., Nakamura, S., Yoshida, A., Kaji, S., Yasuda, M., & Fujisawa, K. (2020, November). Massive parallelization for finding shortest lattice vectors based on ubiquity generator framework. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis* (pp. 1-15). IEEE.

## 国内研究会

- 立岩斉明, 品野勇治, 吉田明広, 鍛冶静雄, 安田雅哉, & 藤澤克樹. (2020). 最短格子ベクトル問題求解における Ubiquity Generator Framework を用いた大規模 MPI 並列化. 研究報告ハイパフォーマンスコンピューティング *(HPC)*, *2020*(1), 1-10.