April 25, 2022

ITOシステムにおけるコード結合フレーム CoToCoAを用いた宇宙プラズマ連成計算シミュレーションの計算・電力性能評価

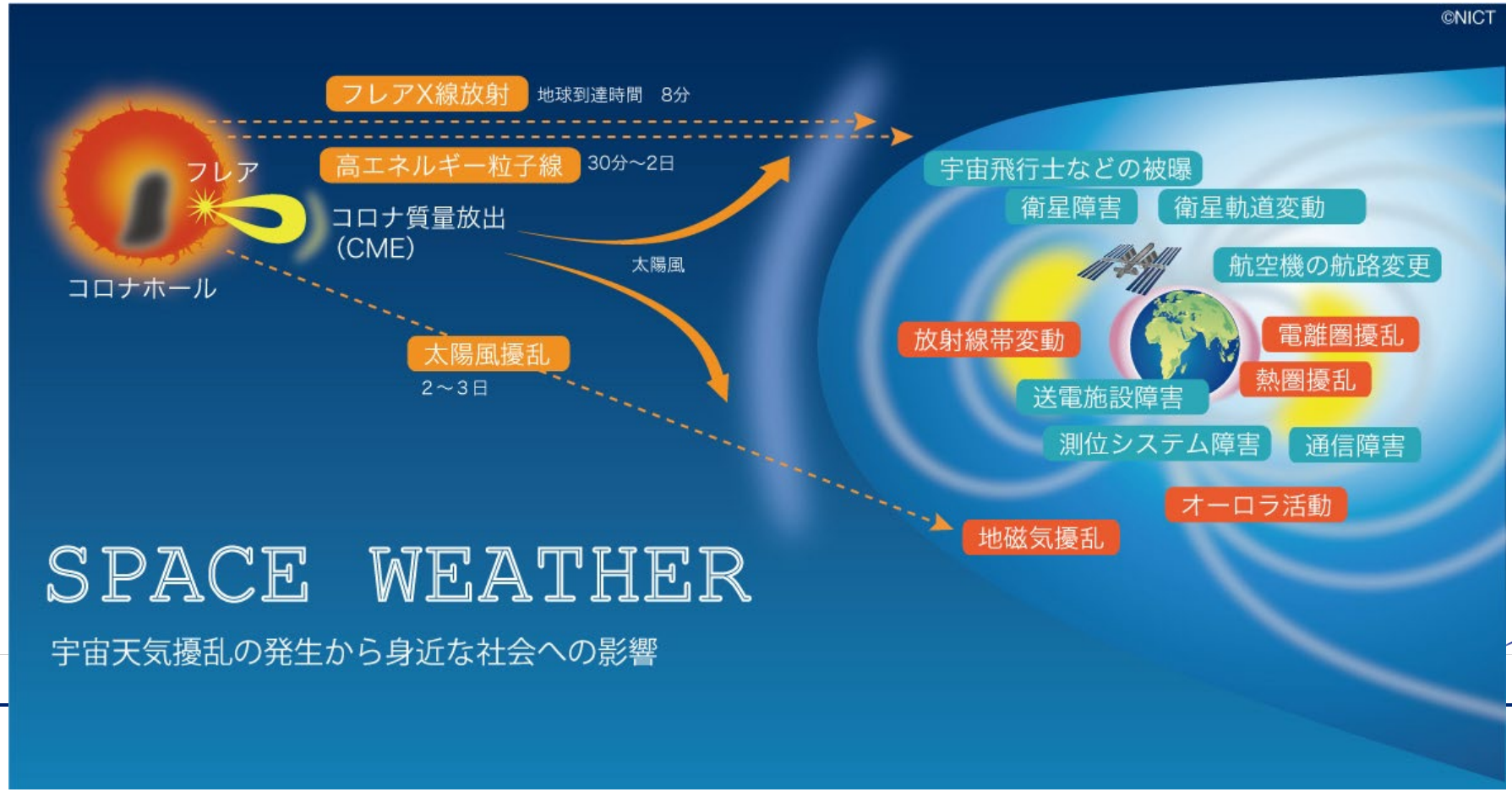
深沢 圭一郎[1]、加藤 雄人[2]、三宅 洋平[3]、南里 豪志[4]、中澤 和也[5]

1. 京都大学学術情報メディアセンター
2. 東北大学大学院理学研究科
3. 神戸大学計算科学教育センター
4. 九州大学情報基盤研究開発センター
5. 神戸大学大学院システム情報学研究科

Background 1

宇宙プラズマ

- 太陽の様々な活動によって宇宙プラズマが動き、宇宙環境変動、地上への電磁的影響が引き起こされる。



Background 2

宇宙プラズマシミュレーション

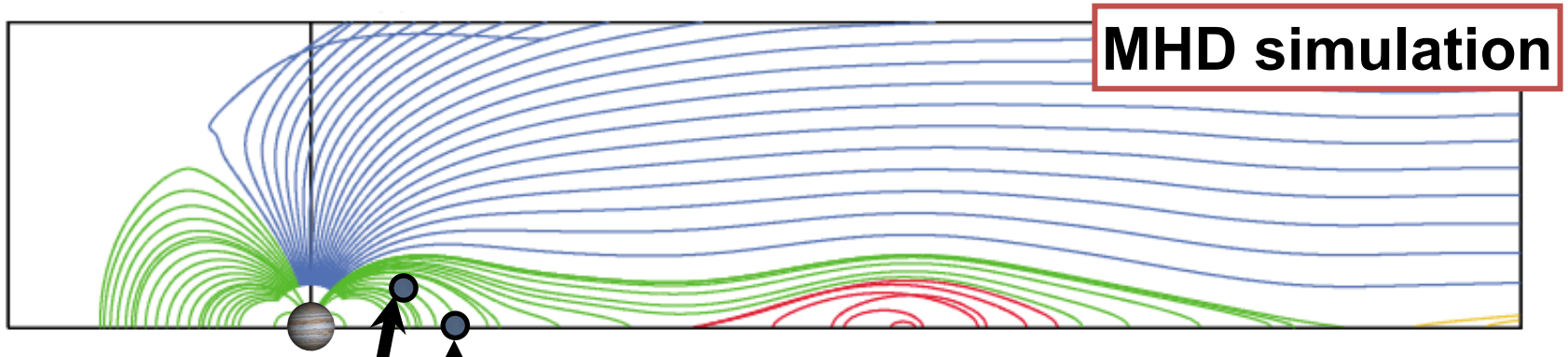
- 宇宙プラズマを流体近似した磁気流体力学（MHD）コードは磁気圏と呼ばれる巨大な惑星磁場の勢力範囲をシミュレーションすることができ、流体近似と粒子を合わせたハイブリッドコードや、粒子コードは磁気圏内やその周辺領域の局所的な運動論を含む物理過程をシミュレーションすることができる。
- 広範囲の宇宙天気を予報する場合、MHDシミュレーションが用いられるが、プラズマの振る舞いを理解するために重要な波動粒子作用などMHDシミュレーションで扱えない現象は、ハイブリッドコードや粒子コードを用いて研究開発が進んでいる。

Motivation

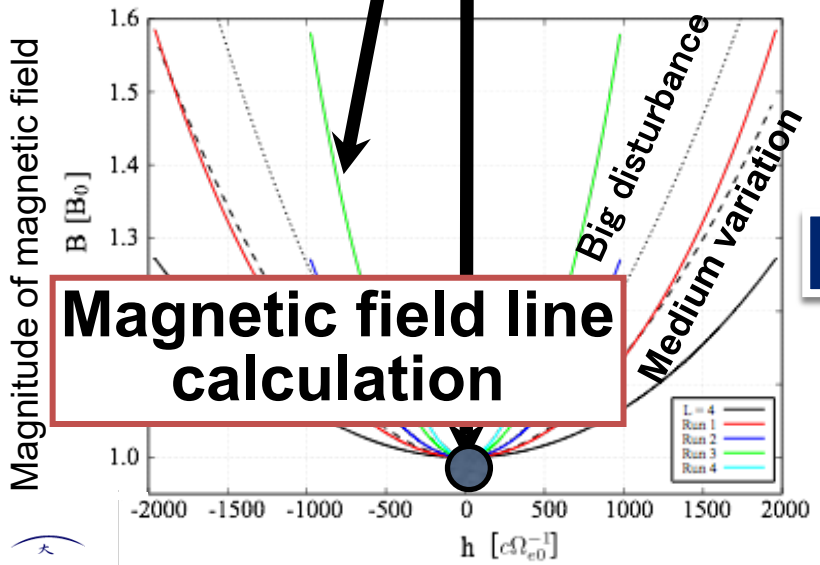
連成計算動作・性能評価と消費電力削減研究

- 本研究ではITOシステムにおいて、この宇宙プラズマの振る舞いを解く粒子コード、ハイブリッドコード、MHDコードにCoToCoAを用いて連成計算させた新しいシミュレーションコードの動作・性能評価を行うことを目的としている。
- CoToCoAは本研究グループが開発しているコード連結フレームワークである。
- また、このような宇宙プラズマコードなどをITO上で走らせる際に消費電力を削減させる手法の評価も行った。

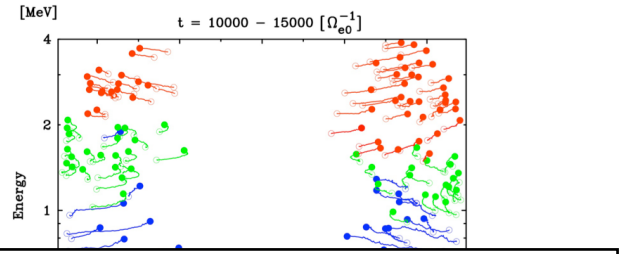
連成計算の例



MHD simulation

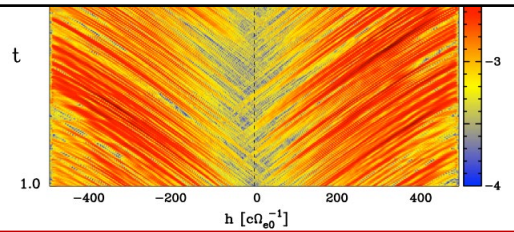


Magnetic field line calculation



Relativistic electron acceleration

Generation of nonlinear wave



Electron Hybrid simulation

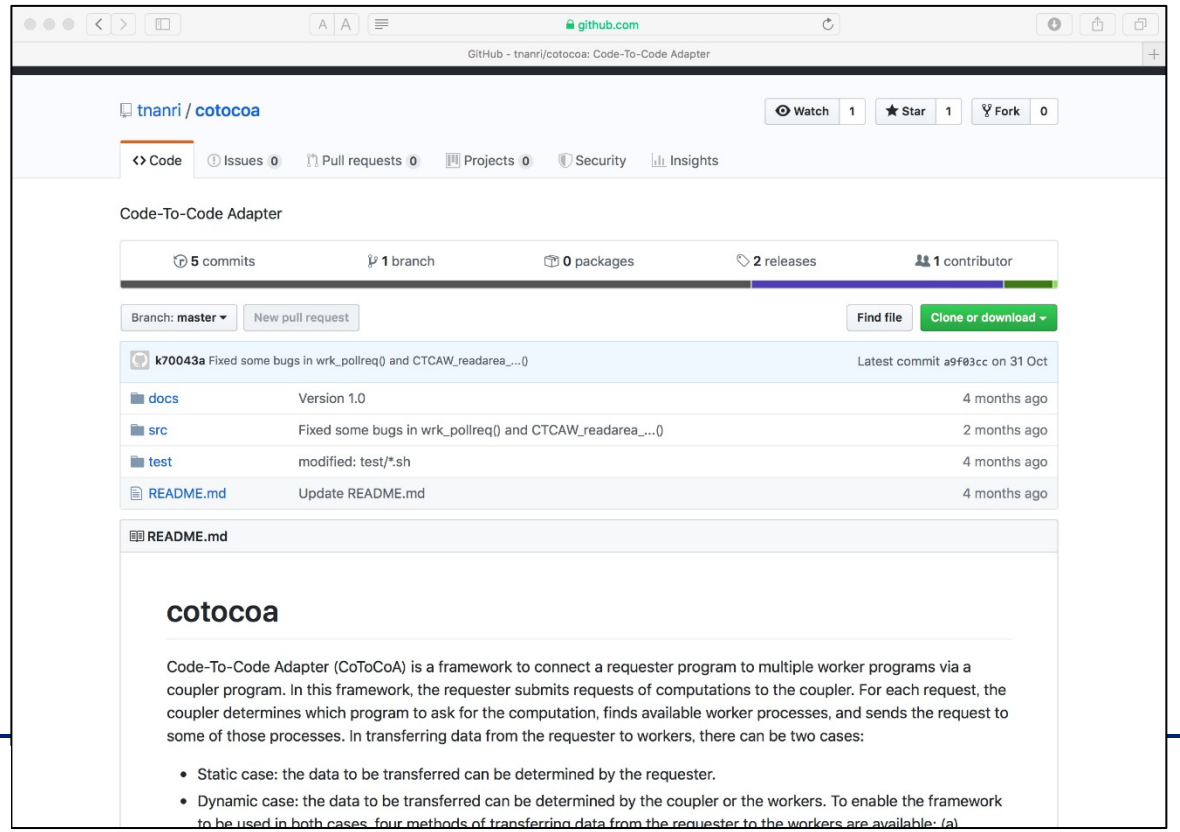
京都大学
KYOTO UNIVERSITY

Code-To-Code Adapter (CoToCoA) Library

連成計算のためのフレームワーク

開発の方針

- ✓ 超並列計算で実用に耐える連成計算コードを開発する
- ✓ 連成計算のために必要なコード変更の手間を最小化する



挑戦的研究（萌芽）2017–2019年度
「スケーラブル通信ライブラリを用いた惑星圏
次世代連成計算技術の創出」により開発。

科研費 KAKENHI

京都大学 KYOTO UNIVERSITY

https://github.com/tnanri/cotocoa

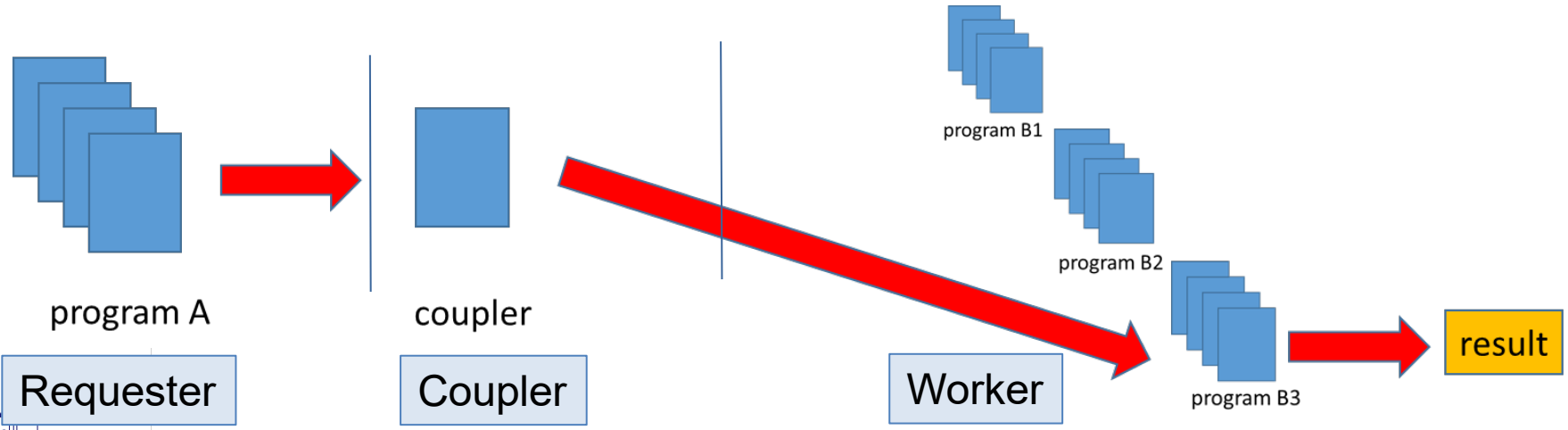
CoToCoAの概略

連成計算のための3種類のプログラム向けインタフェース

Requester : Couplerに計算依頼（request）を送信

Coupler : Requesterからの requestに応じてプログラムを選択
空いている Worker担当プロセス群に requestを転送

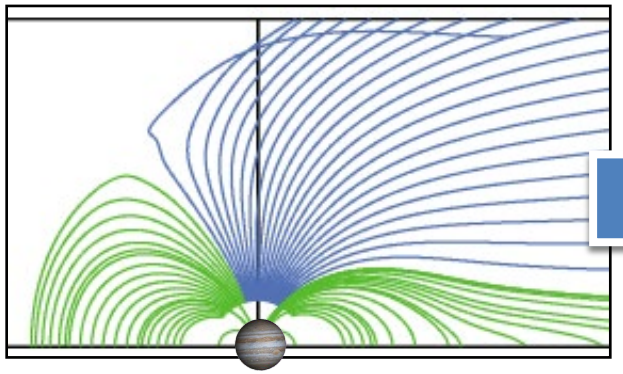
Worker : Couplerからの requestに応じてプログラムを実行



CoToCoAの使われ方

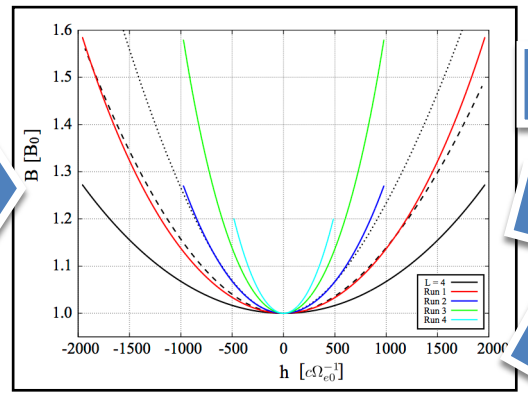
Requester

MHD

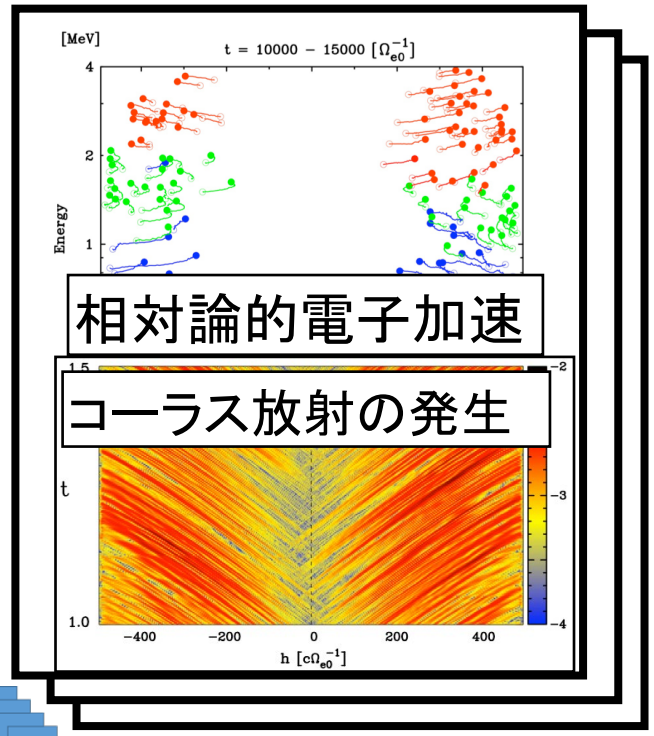


Coupler + α

磁力線抽出



Worker



Requester Coupler Worker

京都大学 KYOTO UNIVERSITY

実装の概略

Coupler reads the data of Requester and transfer it to Worker

Requester

```
call CTCAR_init()

! Initialization part

call CTCAR_regarea_real8( &
     data, size, areaid)

do step = 1, nsteps
!  main calculation part

  call CTCAR_sendreq(dataint, &
       dataintnum)
end do

call CTCAR_finalize()
```

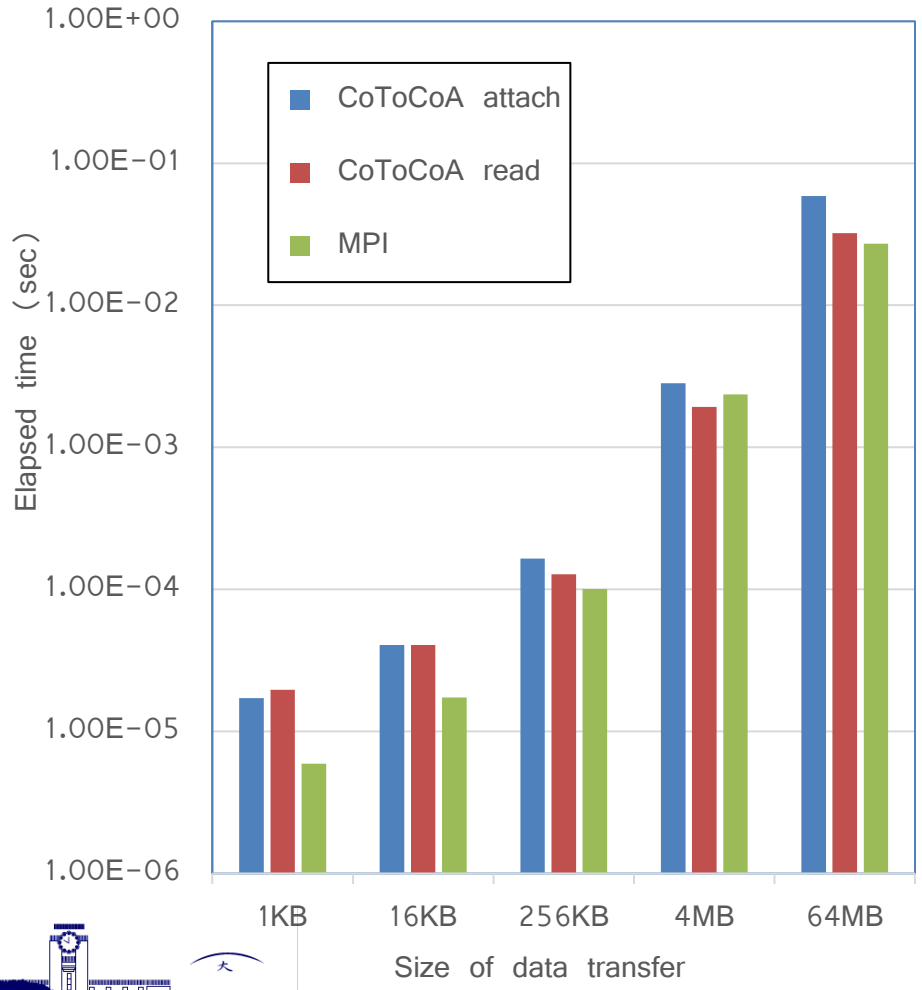
Coupler

```
call CTCAC_init()

! Initialization part

call CTCAC_regarea_real8(areaid)

do while
  call CTCAC_pollreq(reqinfo, fromrank, &
       dataint, dataintnum)

  if (CTCAC_isfin()) then
    exit
  end if

! select the program of Worker and rank, offset
! calculation part

  call CTCAC_readarea_real8(areaid, rank, &
       offset, size, data)

  call CTCAC_enqreq_withreal8(reqinfo, progid, &
       dataint, dataintnum, data, size)
end do
call CTCAC_finalize()
```

Worker

```
call CTCAW_init(progid, procsperreq)

call CTCAW_regarea_real8(areaid)

do while
  call CTCAW_pollreq_withreal8( &
       fromrank, dataint, dataintnum, &
       data, size)

  if (CTCAW_isfin()) then
    exit
  end if

! Initialization part

! main calculation

  call CTCAW_complete()

end do
call CTCAW_finalize()
```

Performance evaluation



Compare elapsed time of a request

From "sendreq" to "complete"

Difference among three data-transfer styles:
- CoToCoA attach: Attach to a request
- CoToCoA read: Remote read
- MPI: Use MPI_Send and _Recv

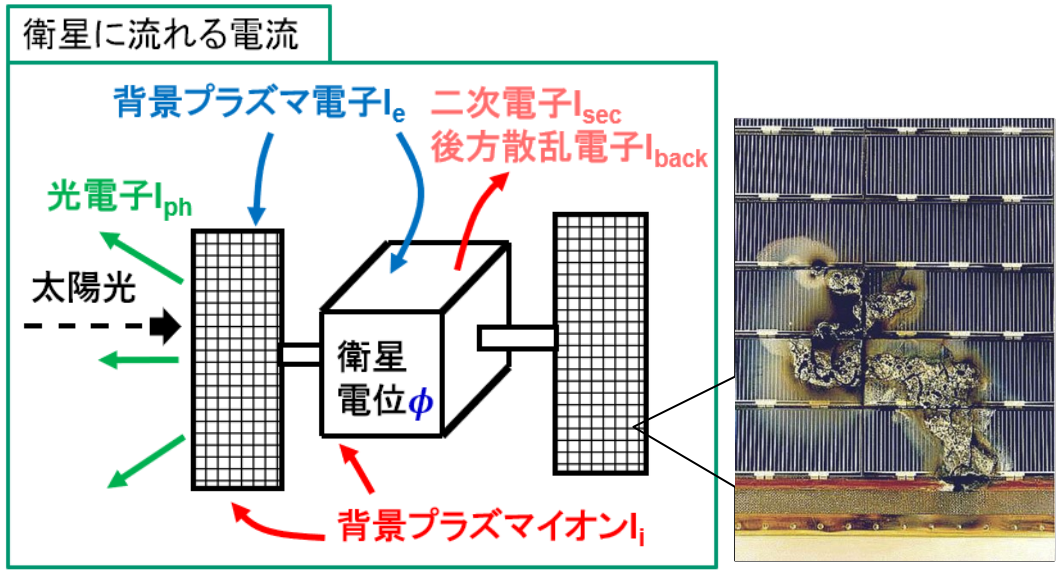
Environment

- ITO subsystem A of Kyushu Univ., Japan
  - Intel Xeon, Mellanox InfiniBand FDR
  - RedHat Enterprise 7, Intel MPI
- 4 nodes
  - 4 procs / node
  - 4 procs for requester, one proc for coupler and 6 procs for worker.

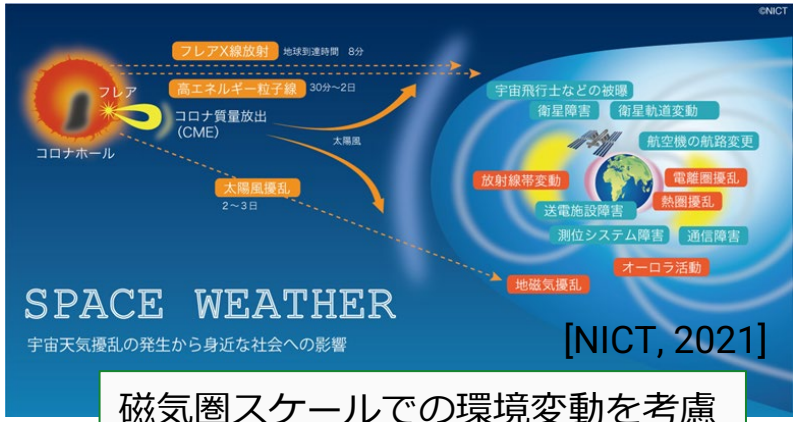
研究の背景

衛星帯電

人工衛星に宇宙プラズマ粒子が衝突し、帯電する
→放電による**衛星の故障原因**
宇宙天気じょう乱時には異常帯電が発生



[ESA, 2020]



[NICT, 2021]

磁気圏スケールでの環境変動を考慮

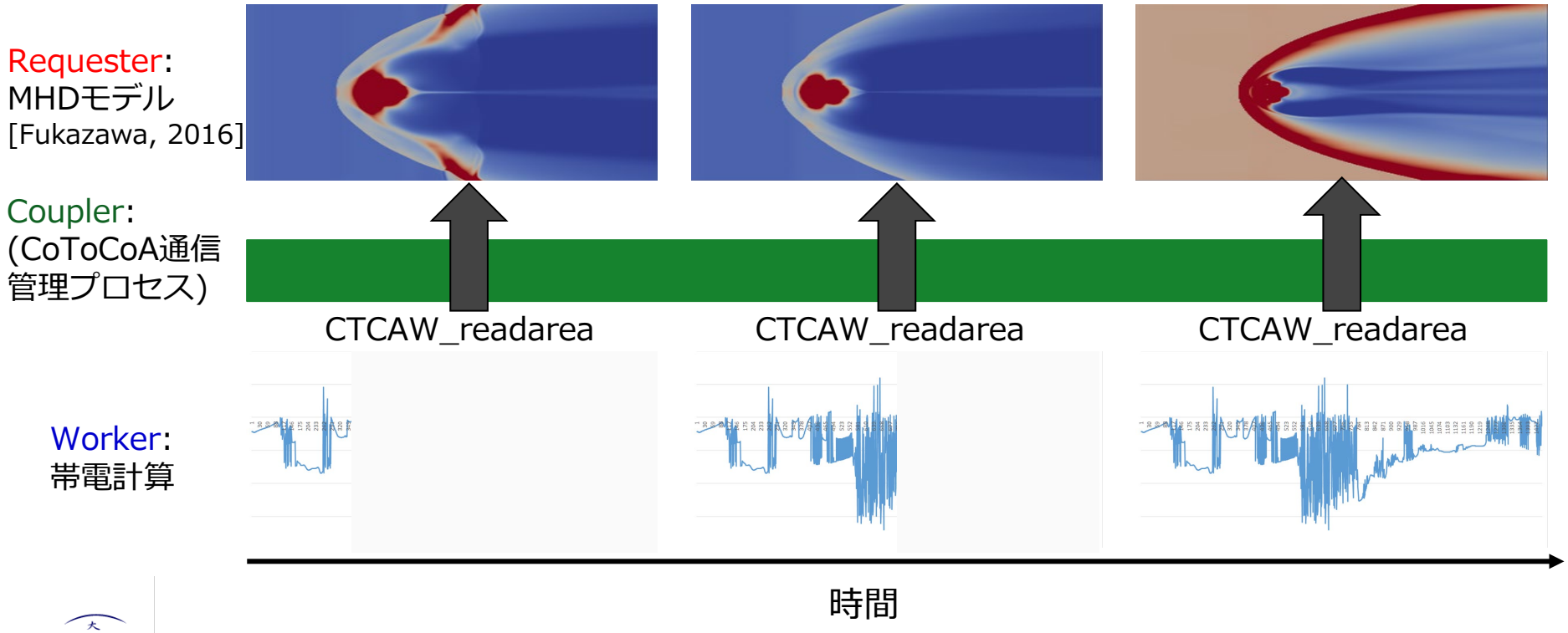
特定の環境変動に対応した帯電解析

**磁気圏グローバル
MHDシミュレーションと
粒子モデル衛星環境シミュレーションの
連成による帯電解析**

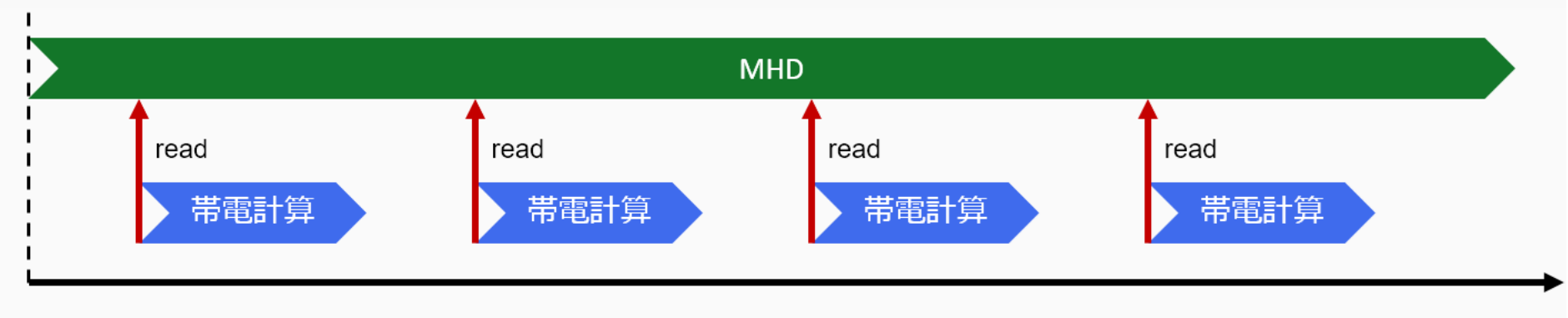
開発実装内容：MHD—衛星帯電解析連成プログラム

コード間結合フレームワークCoToCoA(Code To Code Adapter)を利用

帯電計算に必要な環境データ(プラズマ密度・温度)をMHDから随時取得



開発実装内容：連成計算にあたっての非同期性



帯電計算での遅延がMHDに影響しない実装が必要

- MHD：確保した配列に宇宙環境データを決められた間隔で書き込む
- 帯電計算：配列を読み、更新されていれば計算を進める

帯電計算側のみがCoToCoAのサブルーチンをcallする

- MHDは帯電計算と待ち合わせをせずに計算を進めることができる

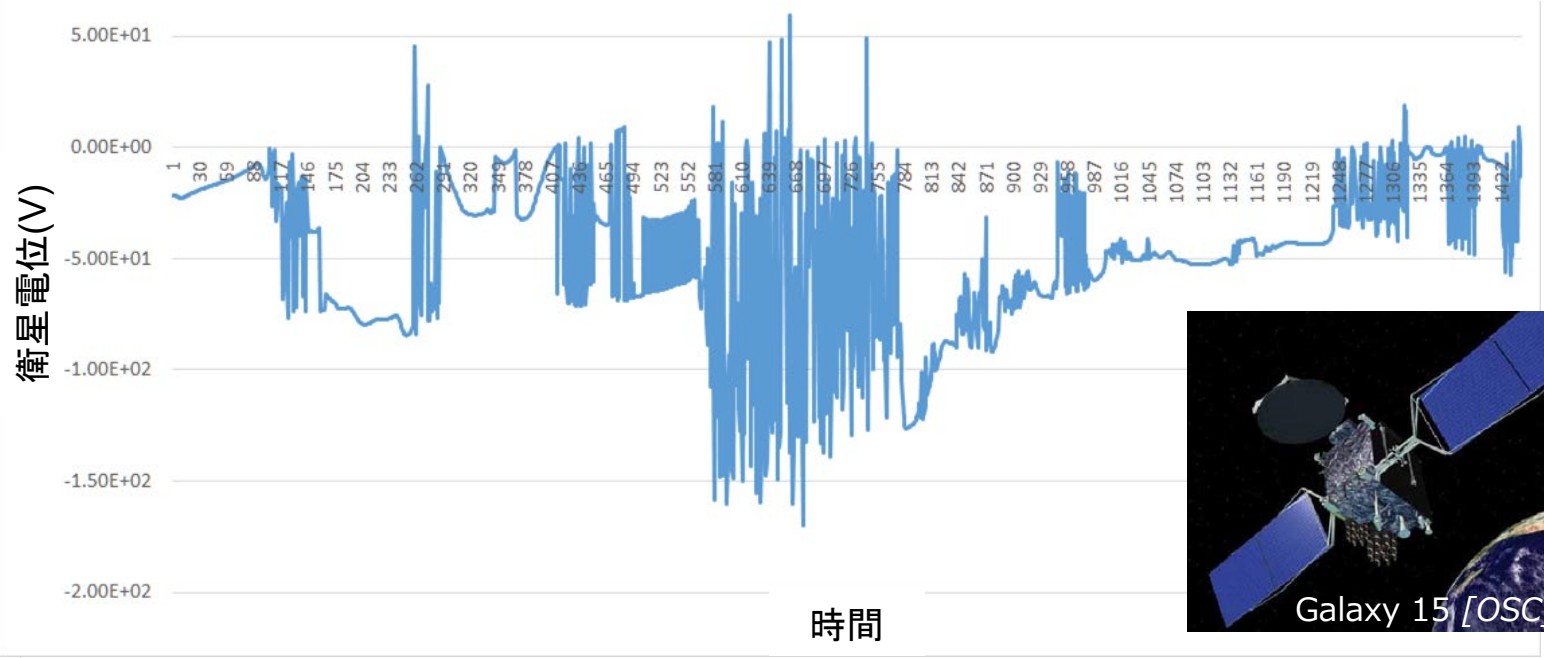
評価：宇宙環境変動に対応した帯電解析

Galaxy15衛星故障時を再現したシミュレーションで評価

CoToCoAによる連成計算を実現

衛星電位計算および連成計算通信コストはおおむねMHD計算時間内に隠ぺいされている

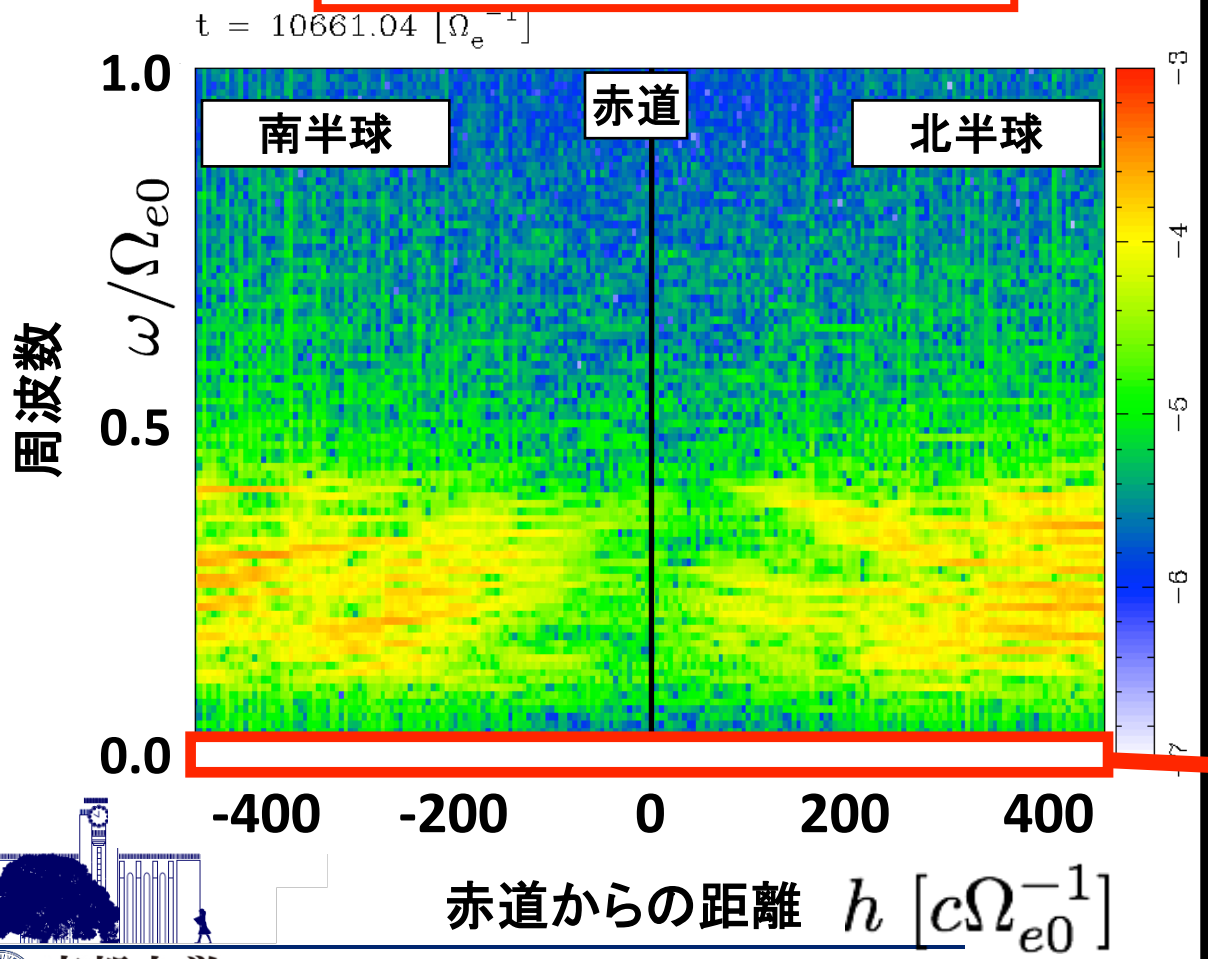
今後の課題：一部の時間帯で衛星電位の解が不安定になっている点の原因調査



地球磁気圏でのプラズマ波動の発生(粒子計算)と伝搬(流体計算)過程の連成計算

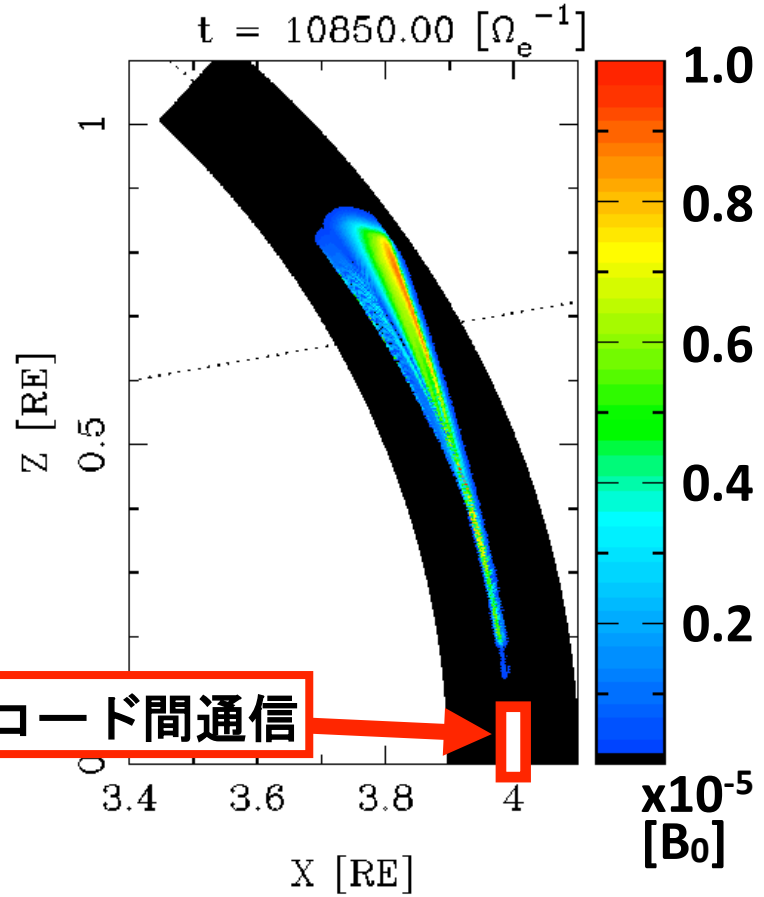
磁力線に沿った空間1次元計算

プラズマ粒子コード



磁気圏子午面の空間2次元計算

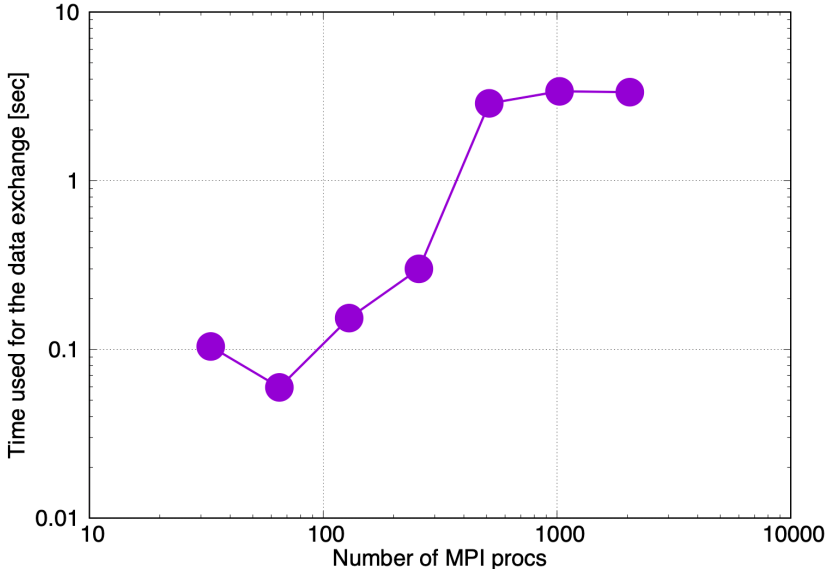
プラズマ流体コード

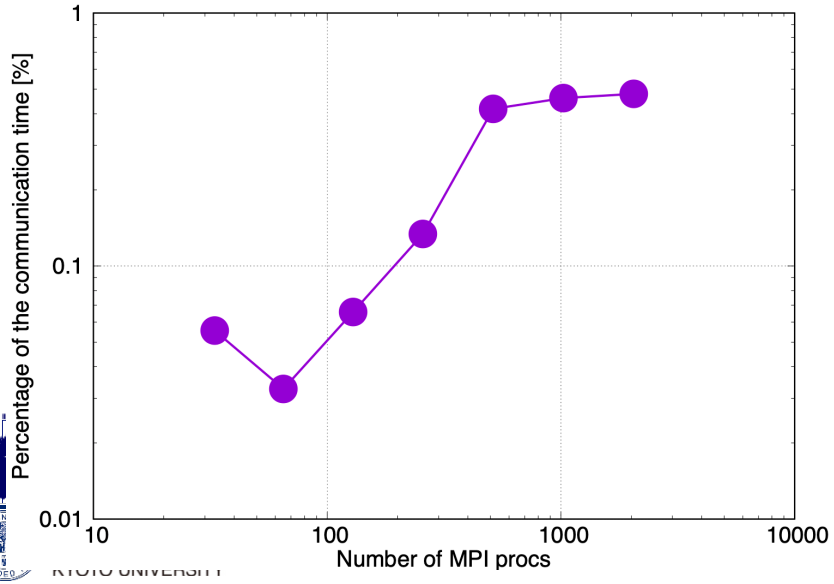


コード間通信

京都大学
KYOTO UNIVERSITY

Performance





The elapse time required for the data exchange between the codes includes the time used in;

(1) the data transfer from the Requester to the Coupler program through CTCAR_sendreq and CTCAC_pollreq APIs

(2) the data transfer from the Coupler to the Worker program through CTCAC_sendreq and CTCAW_pollreq APIs

(3) the broadcast of the received data to all processes of the Worker program.

- The cost corresponding to (3) increases as the number of MPI processes increases
- The increase in the case of 33 processes would be related to the structure of the subsystem A (36 cores/node)
- Despite the increase of the elapse time for the data exchange, its cost is less than 1% of the total elapse time

The present study clarifies that we can realize the 'strong' cross-reference simulation without additional computational resources

Power Evaluation Environment

シミュレーションセッティング

- CPU消費電力制限の計算性能最適化手法への影響を調べるために、配列構造最適化に注目する。
- MHDシミュレーションでは、空間3次元(x, y, z)とMHD変数($m = 8$)をまとめた4次元配列計算に利用している。
- 通常はベクトル向け(x, y, z, m)というSoA形式を用いているが、キャッシュ最適化向けAoS形式(m, x, y, z)を用いると性能が向上するCPUもある。
- 近年はベクトルとキャッシュ最適化を両立させる場合に計算性能が上がることもあり、(x, y, m, z)や(x, m, y, z)という配列構造も利用している。
- そこで今回はこれら4つの配列構造最適化(以降の配列構造をそれぞれ，**xyzm**，**xymz**，**xmyz**，**mxyz**とする)を利用し評価を行う。

Evaluation Environment 3

消費電力制限セッティング

- 消費電力制限手法としては、「**RAPLによるCPU消費電力制限**」と「**利用CPUコア数変化による消費電力制限**」を考える。
- **CPU消費電力制限**
  PKG（CPU）利用可能電力を60W～160Wの範囲で制限する。
  実行はRICの利用制限により、1プロセス/1ノードとし、8（2×2×2の3次元領域分割）プロセス×36スレッドの並列実行としている（並列数は一定）。
- **利用コア数制限**
  1ノードの利用コアを2～36コアと変化させる。
  動作の関係上Flat MPIを用いて実行し、利用コア数によって計算サイズは変わらない（300×300×400の一定、strong scaling）。

Evaluation Results | CPU power capping 1

配列構造xyzmのMHDシミュレーションコード測定結果1

Capping [W]	B/F	Calc time [s]	PKG [W]	DRAM [W]	FREQ [GHz]
60	0.155	32.74	59.79	35.69	1.77
70	0.126	29.65	69.79	35.93	2.18
80	0.110	29.44	79.76	36.10	2.50
90	0.098	27.18	89.66	36.28	2.81
100	0.091	30.89	99.59	36.11	3.01
110	0.085	29.51	109.67	36.21	3.21
120	0.082	29.16	119.61	36.11	3.33
130	0.079	28.37	129.57	36.23	3.47
140	0.075	28.78	139.49	36.22	3.63
150	0.074	28.25	148.51	36.26	3.69
160	0.074	27.44	150.83	36.27	3.71

減少 増加 変化なし

Evaluation Results | CPU power capping 2

配列構造xyzmの測定結果2

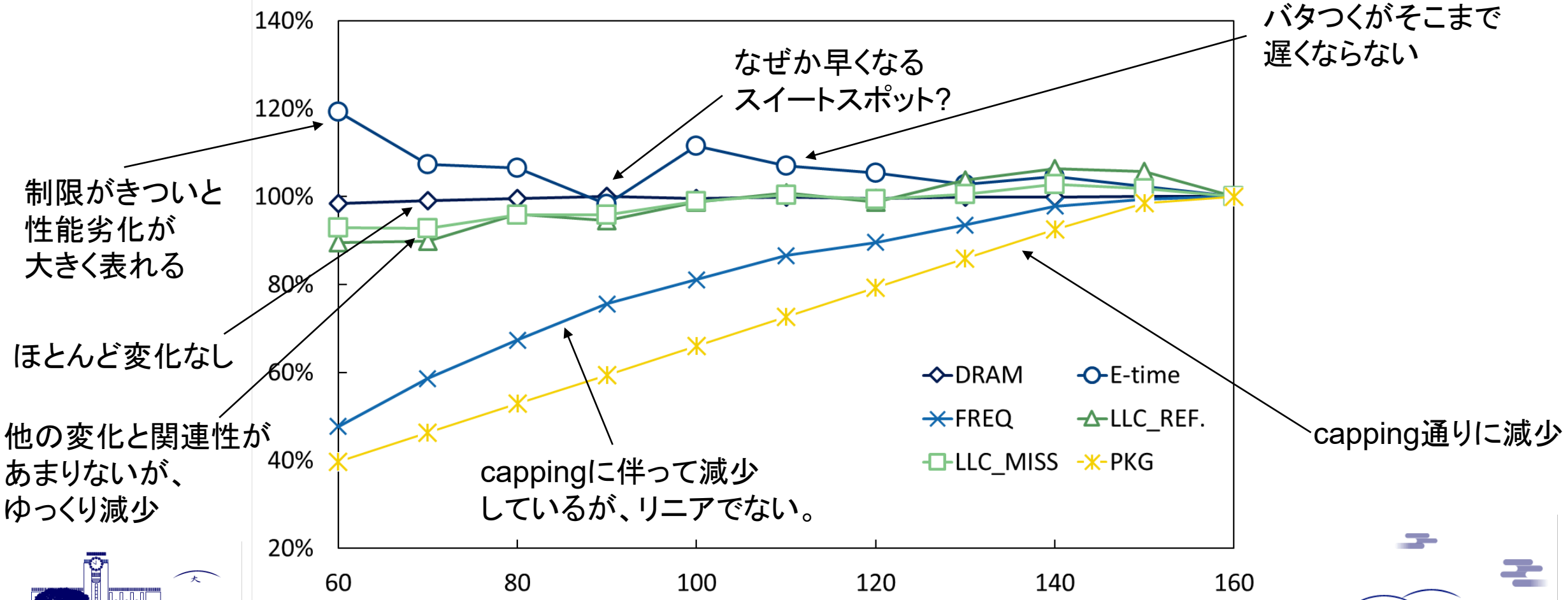


Fig. 5. Variation of xyzm results under CPU power capping.

Evaluation Results | CPU power capping 3

配列構造xymzの測定結果1

Capping [W]	B/F	Calc time [s]	PKG [W]	DRAM [W]	FREQ [GHz]
60	0.164	32.44	59.83	34.48	1.67
70	0.132	31.28	69.80	34.57	2.07
減少 80	増加 0.114	30.32	79.76	変化なし 34.71	2.40
90	0.101	28.51	89.71	34.94	2.72
100	0.093	28.06	99.67	34.79	2.94
110	0.087	28.32	109.59	34.76	3.15
120	0.082	29.82	119.60	34.76	3.36
130	0.079	28.23	129.56	34.76	3.46
140	0.077	26.77	139.53	34.86	3.57
150	0.074	29.30	148.98	34.61	3.69
160	0.075	26.93	151.64	35.00	3.68

Evaluation Results | CPU power capping 4

配列構造xymzの測定結果2

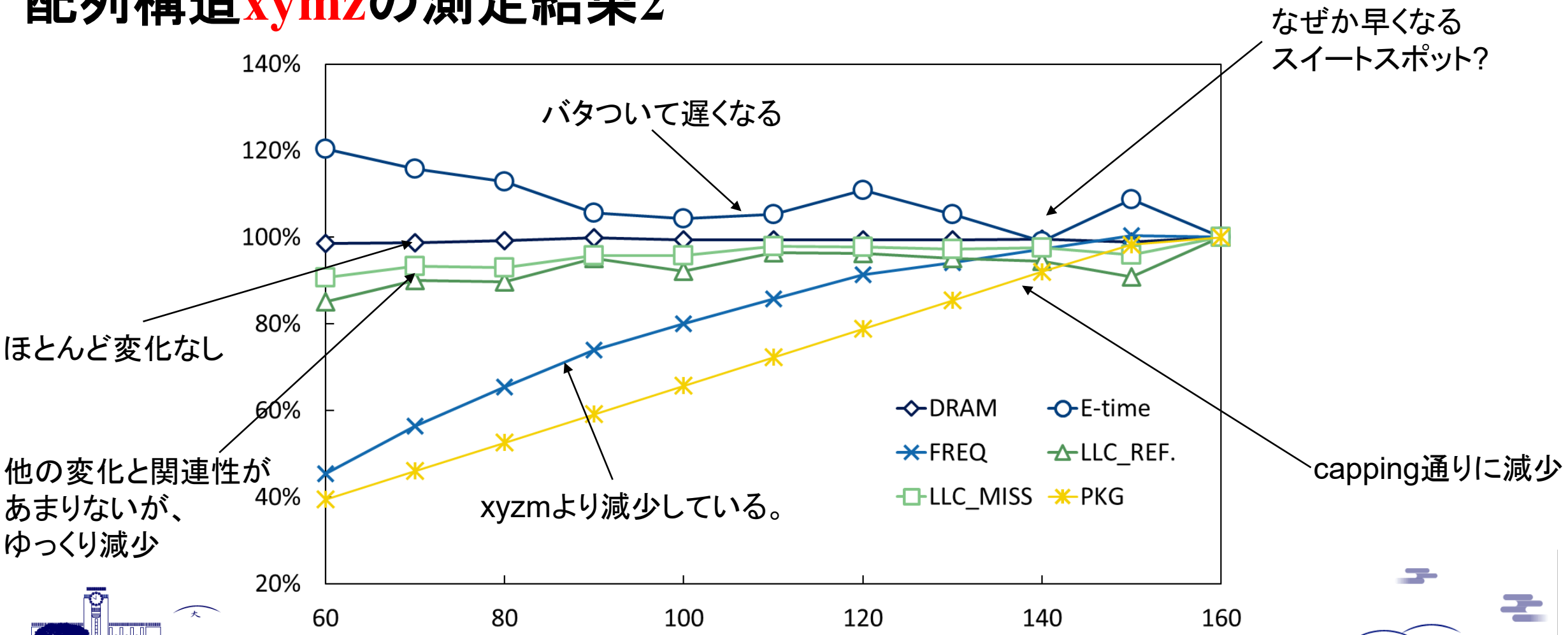


Fig. 6. Variation of xymz results under CPU power capping.

Evaluation Results | CPU power capping 5

配列構造xmyzの測定結果1

Capping [W]	B/F	Calc time [s]	PKG [W]	DRAM [W]	FREQ [GHz]
60	0.169	31.51	59.81	34.07	1.62
70	0.133	30.39	69.77	34.17	2.06
80	0.115	27.28	79.76	34.28	2.39
90	0.100	28.63	89.71	34.23	2.73
100	0.093	29.64	99.68	34.07	2.93
110	0.087	28.32	109.63	34.10	3.16
120	0.082	28.46	119.60	34.10	3.34
130	0.080	28.63	129.54	34.04	3.44
140	0.077	27.01	139.44	34.19	3.58
150	0.075	28.44	148.07	34.07	3.66
160	0.075	27.32	149.48	34.52	3.64

減少 増加 変化なし?

Evaluation Results | CPU power capping 6

配列構造xmyzの測定結果2

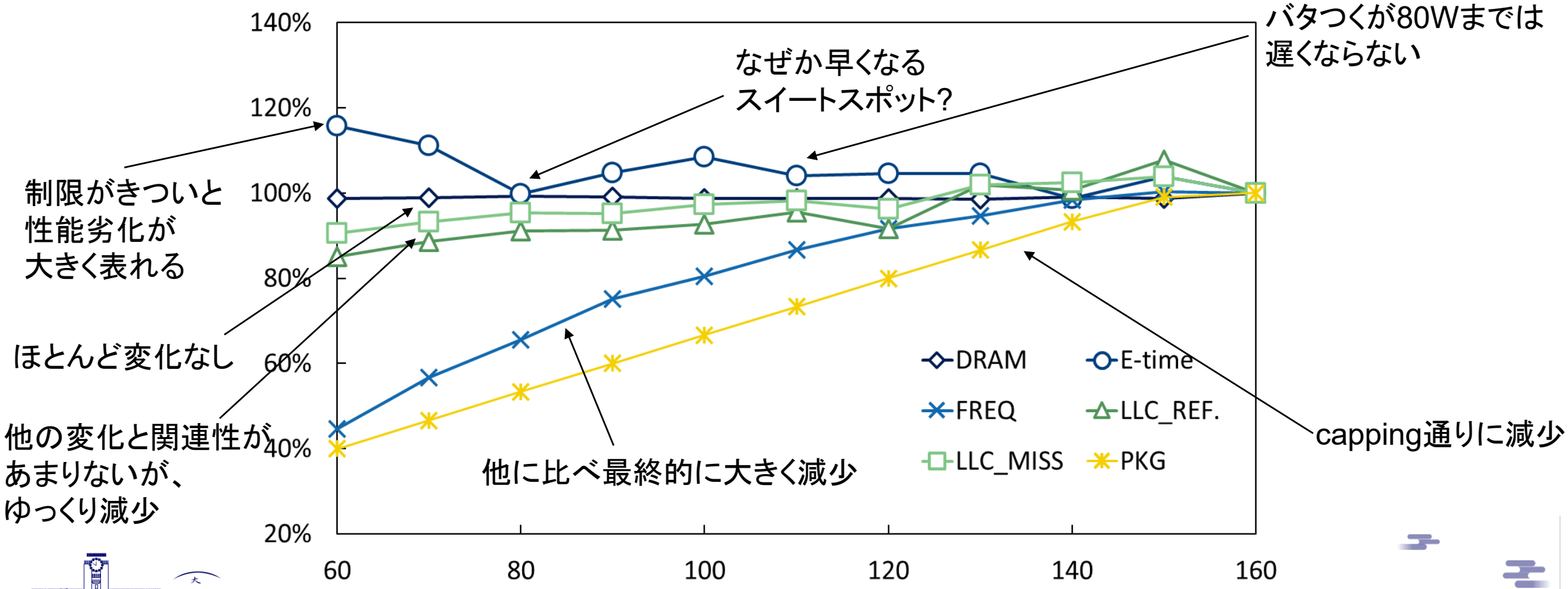


Fig. 7. Variation of xmyz results under CPU power capping.

Evaluation Results | CPU power capping 7

配列構造mxyzの測定結果1

Capping [W]	B/F	Calc time [s]	PKG [W]	DRAM [W]	FREQ [GHz]
60	0.147	26.18	59.83	34.58	1.87
70	0.123	27.14	69.79	34.66	2.23
減少 80	増加 0.106	変化なし 27.54	79.77	34.19	2.58
90	0.095	25.21	89.74	34.28	2.88
100	0.088	25.35	99.69	34.37	3.12
110	0.083	27.17	109.64	34.29	3.31
120	0.083	27.73	119.51	34.58	3.31
130	0.077	23.52	129.41	34.50	3.55
140	0.075	25.62	138.63	34.54	3.66
150	0.075	26.43	143.33	34.56	3.64
160	0.075	24.94	142.39	34.68	3.64

Evaluation Results | CPU power capping 8

配列構造mxyzの測定結果2

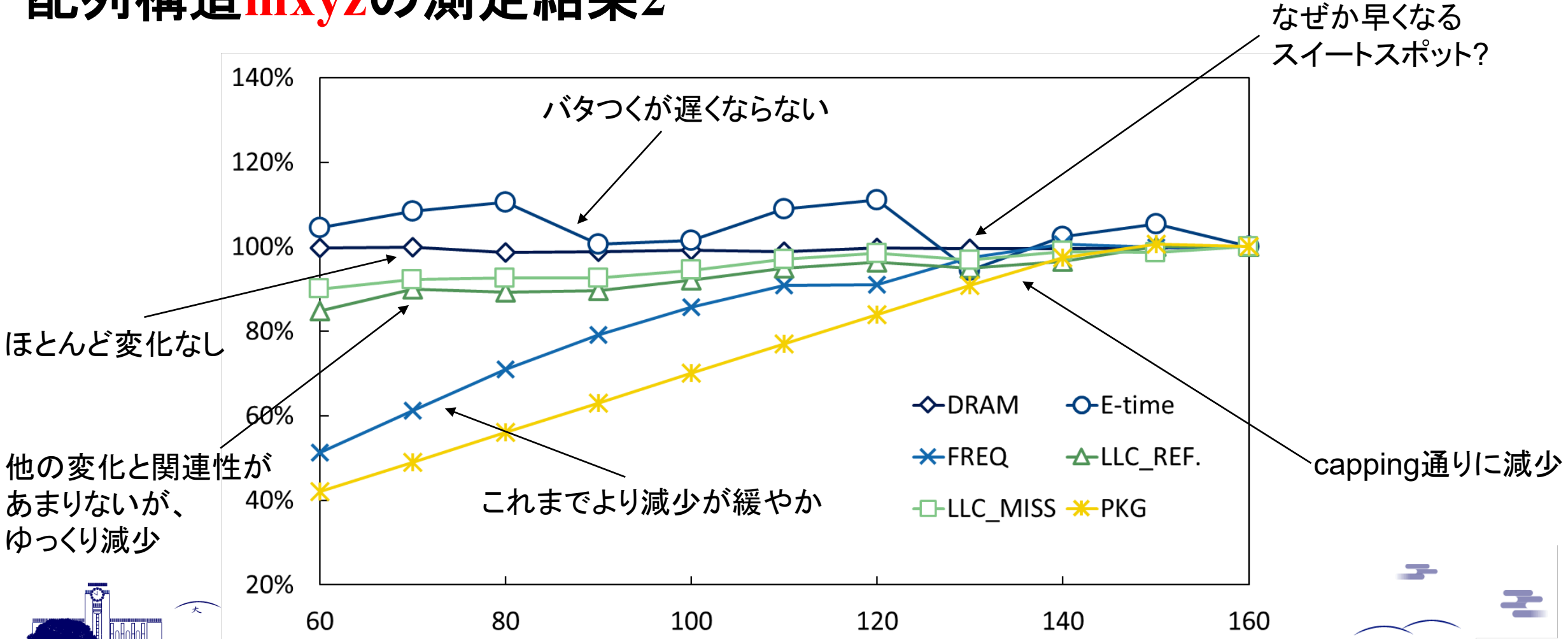


Fig. 8. Variation of mxyz results under CPU power capping.

Evaluation Results | CPU power capping 9

CPU消費電力制限有り無しにおける各配列構造の測定結果比較

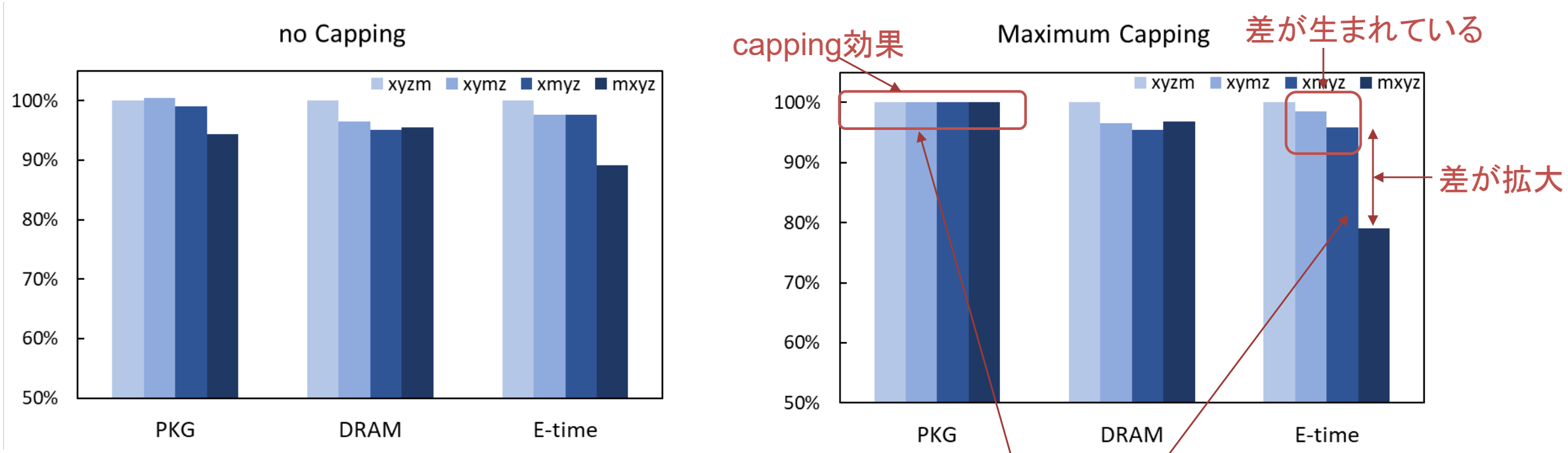


Fig. 9. Comparison of array configuration optimization results with/without CPU power capping.

cappingで消費電力が揃った分、もともとあった消費電力差が、性能に振り替えられている。

Evaluation Results | CPU core capping 1

利用コア数削減における各配列構造の計算時間

Cores	Rpeak [GFlops]	B/F	E-time [s]			
			xyzm	xymz	xmyz	mxyz
2	192	0.666	106.31	124.39	116.71	139.78
4	384	0.333	58.41	67.81	61.69	74.72
8	768	0.167	38.46	39.09	35.07	41.10
12	1152	0.111	29.64	32.95	29.97	31.36
16	1536	0.083	27.36	27.88	25.88	26.02
18	1728	0.074	26.35	26.54	25.33	25.11
36	3456	0.074	13.29	13.61	12.80	12.74

減少

増加

RAPLでの60W制限相当

変化なし

本研究で利用している環境では、コア毎の電力が測定できないことや、CPUコアのCステートを変更できないため、利用コア削減によるCPU消費電力の削減を測定できなかった。ここでは、第1近似的にコアが減る分消費電力も減ると想定する。

Evaluation Results | CPU core capping 2

利用コア数削減における各配列構造の計算時間と性能変化

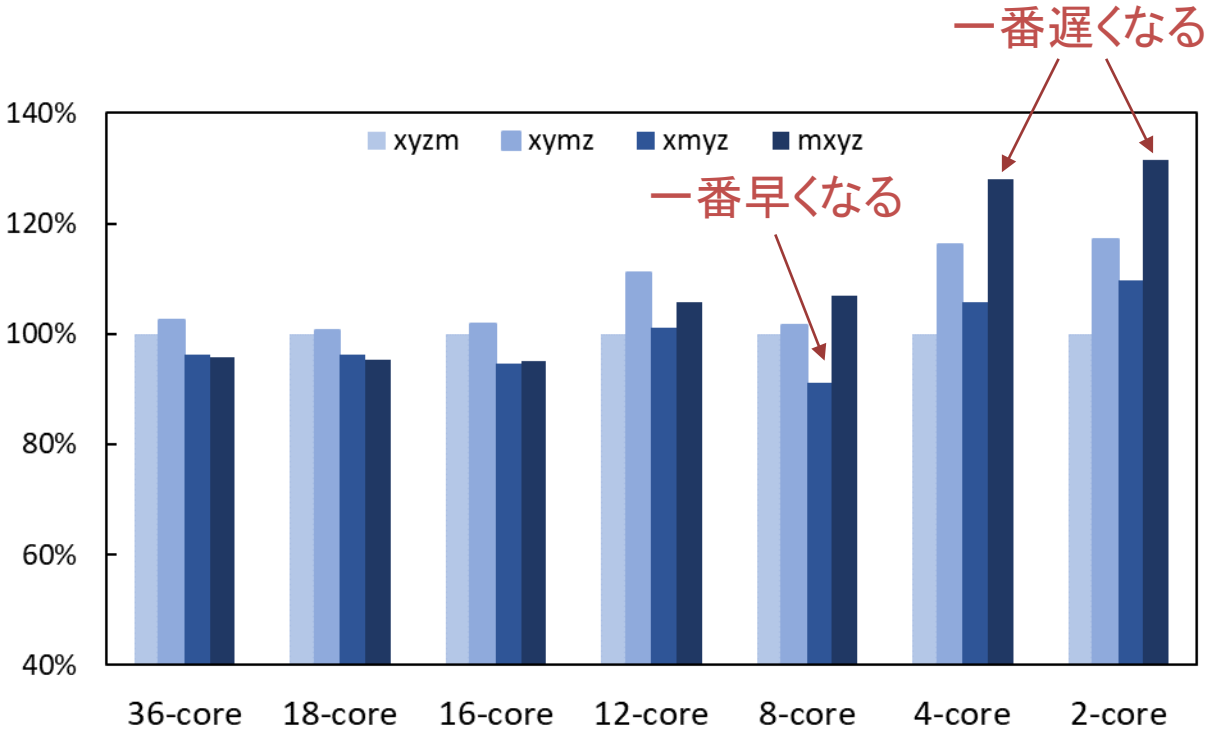


Fig. 10. Comparison of calculation time with decreasing CPU core usage

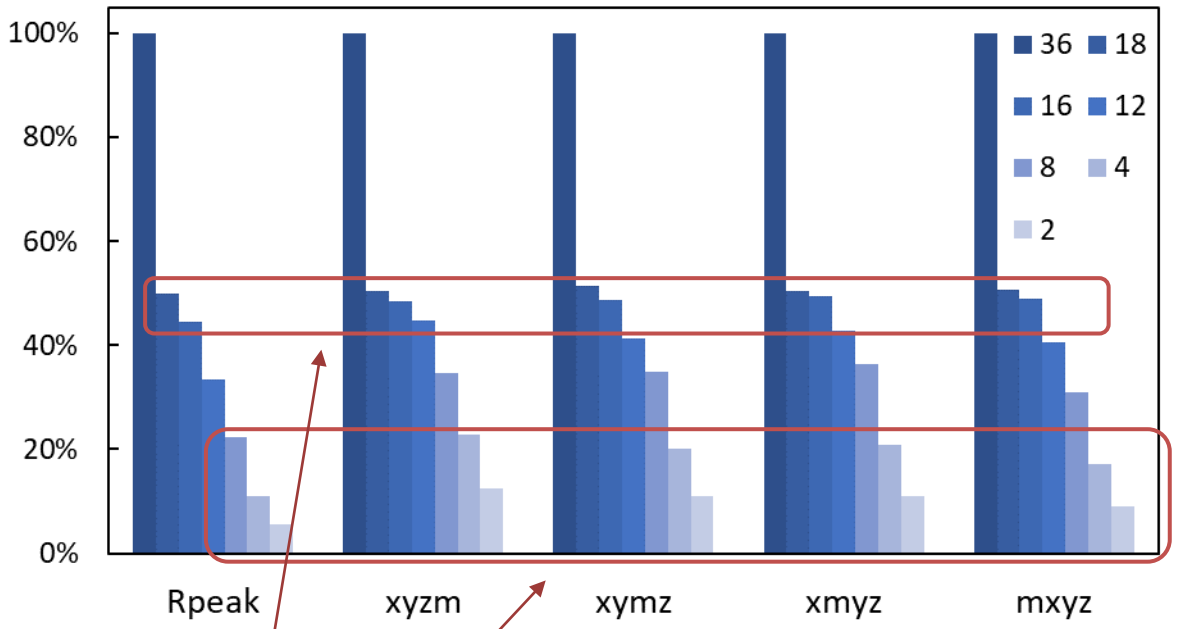


Fig. 11. Performance variation of array configuration decreasing CPU core usage.

Summary

ITO全ノードを使い、連成計算の実行確認を行った。

- ✓ 連成計算ライブラリCoToCoAの基本性能測定を行い、連成計算によくある大きなデータを移動させる際にはかなりオーバーヘッドが見えなくなることが確認された。
- ✓ MHD-衛星帯電解析連成コードでは、衛星帯電計算がMHD計算時間に隠蔽されることが確認できた。
- ✓ プラズマ粒子ー流体連成計算では、連成計算にかかる通信時間が全計算時間の1%程度と負荷が低いことが実証された。

ITO全ノードを使い、MHDコードの配列形状による電力性能を評価した。

- ✓ CPU消費電力制限の場合では、配列構造によってCPU周波数の低下の振る舞いに違いが見られ、計算性能の変化に影響を与えていた。
- ✓ 消費電力制限が無い場合においてCPU消費電力が少ない配列構造がCPU消費電力制限時では、高い計算性能を示していた。