SS研科学技術計算分科会
「ハイパフォーマンス・コンピューティングの現在と未来」
2010年10月20日（水）、 ANAクラウンプラザホテル神戸

3 次元MHD コードによる FX1 の性能評価

名古屋大学太陽地球環境研究所
荻野竜樹

共同研究者
梅田隆行 （名古屋大学太陽地球環境研究所）
深沢圭一郎 （九州大学大学院理学研究院地球惑星科学部門）

アウトライン

１．背景

２．地球磁気圏の３次元グローバルMHDモデル

３．PRIMEPOWER HPC2500からFX1とHX600への移行

４．FX1とHX600おける３次元MHDコードの計算速度

５．種々のスパコンの計算速度の比較と考察

６．地球磁気圏の３次元MHDシミュレーションの将来

７．まとめ

１．背景

・太陽地球系科学：太陽活動とジオスペース（地球外圏）環境変化の因果関係を解明する研究

・外部境界をできるだけ遠くに設定してかつ空間分解能をできるだけ上げる

・計算方法の改良と最大級のスーパーコンピュータの効率的な利用

・並列計算共通プログラム／手法の開発

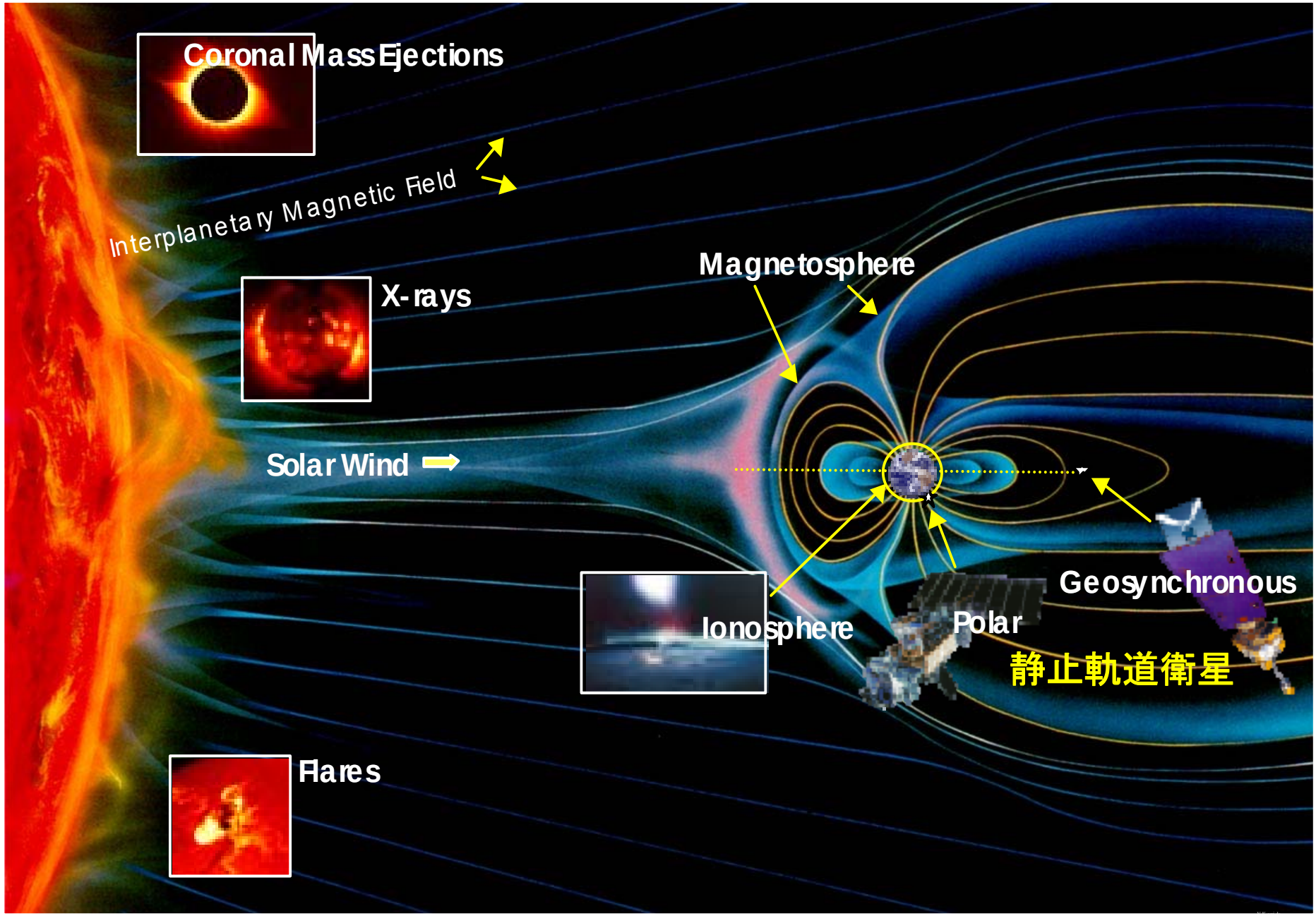
Message Passing Interface (MPI)

・超並列スーパーコンピュータの効率的な利用

富士通株式会社：PRIMEPOWER HPC2500

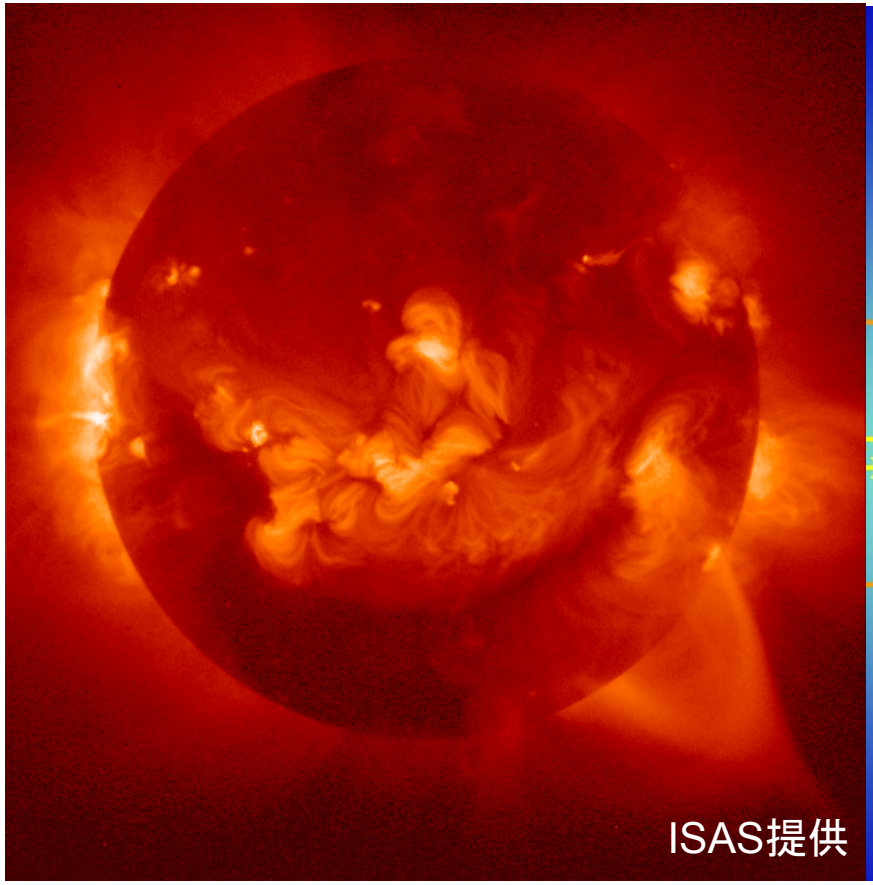
からFX1とHX600へ

Solar-Terrestrial Environment 太陽と地球の関係

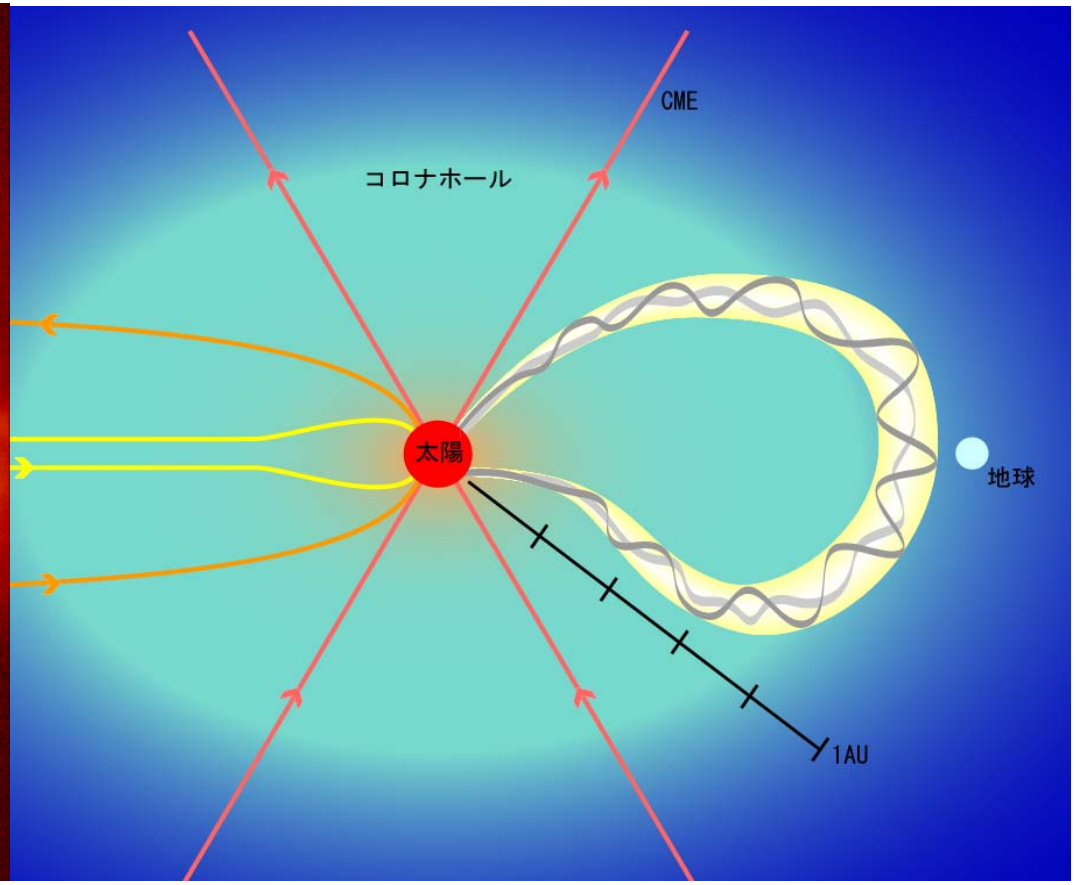


太陽表面爆発現象と惑星間擾乱の伝播

太陽フレアなどの太陽活動はジオスペース環境変化の源



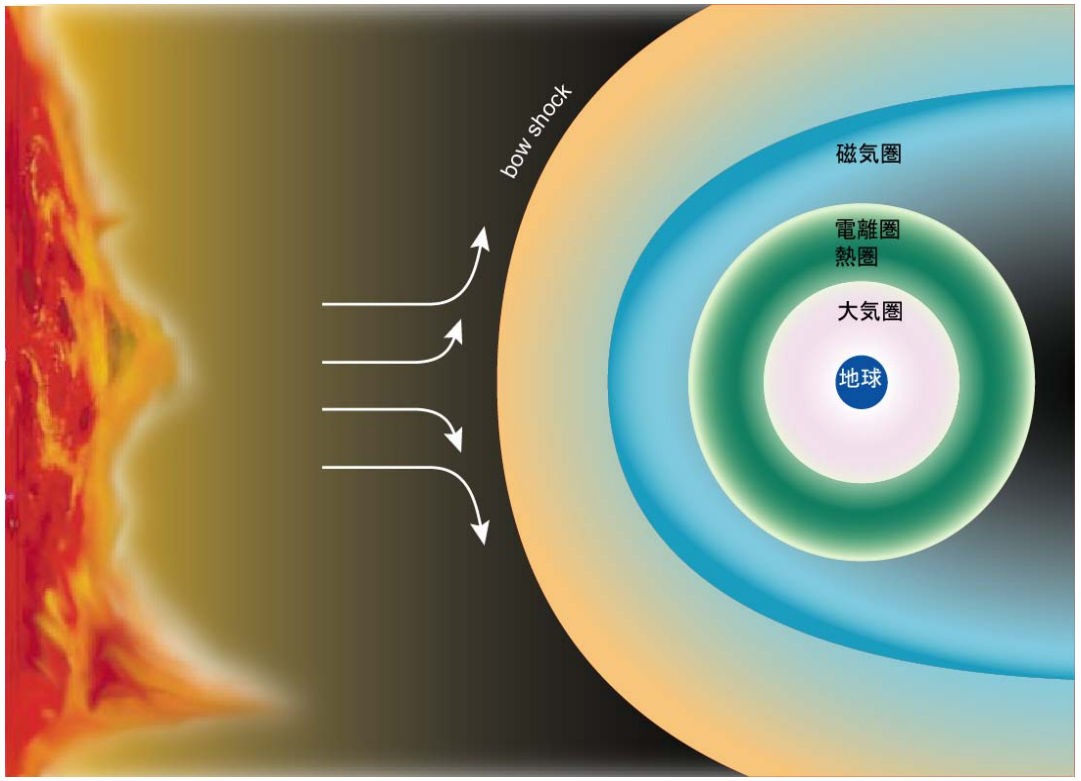
「ようこう」による太陽の軟X線画像



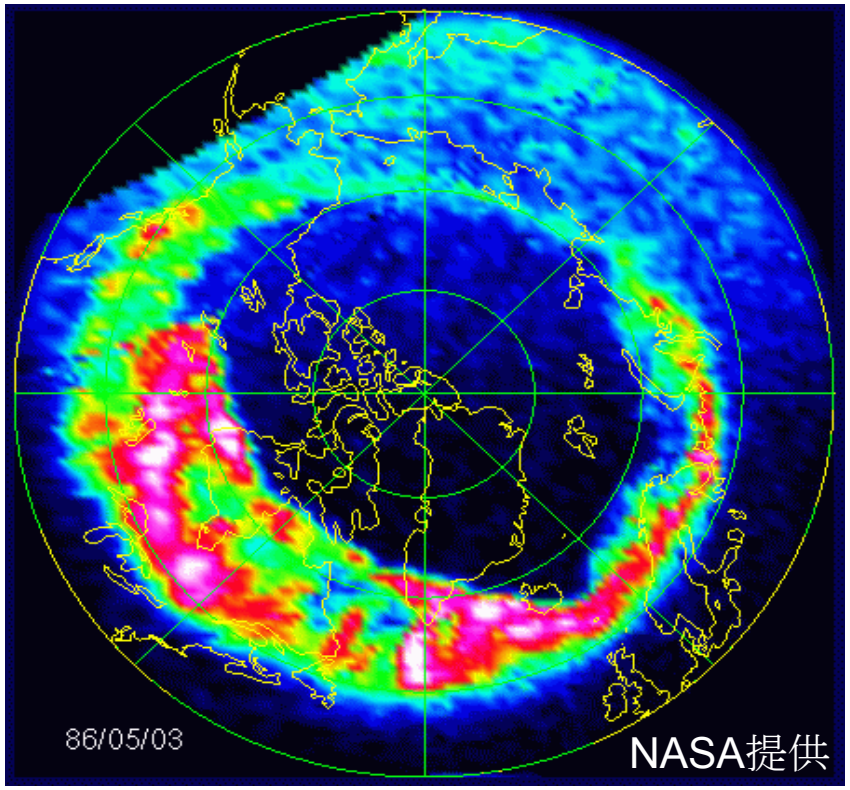
惑星間空間を伝播するCME（コロナ質量放出）

磁気圏・電離圏・熱圏結合：領域間結合

太陽風のエネルギーは磁気圏・電離圏・熱圏の領域間結合を介して地球内部に運ばれ、極域でオーロラを輝かせる。宇宙天気研究はその領域間結合を調べ、予報に結びつける。



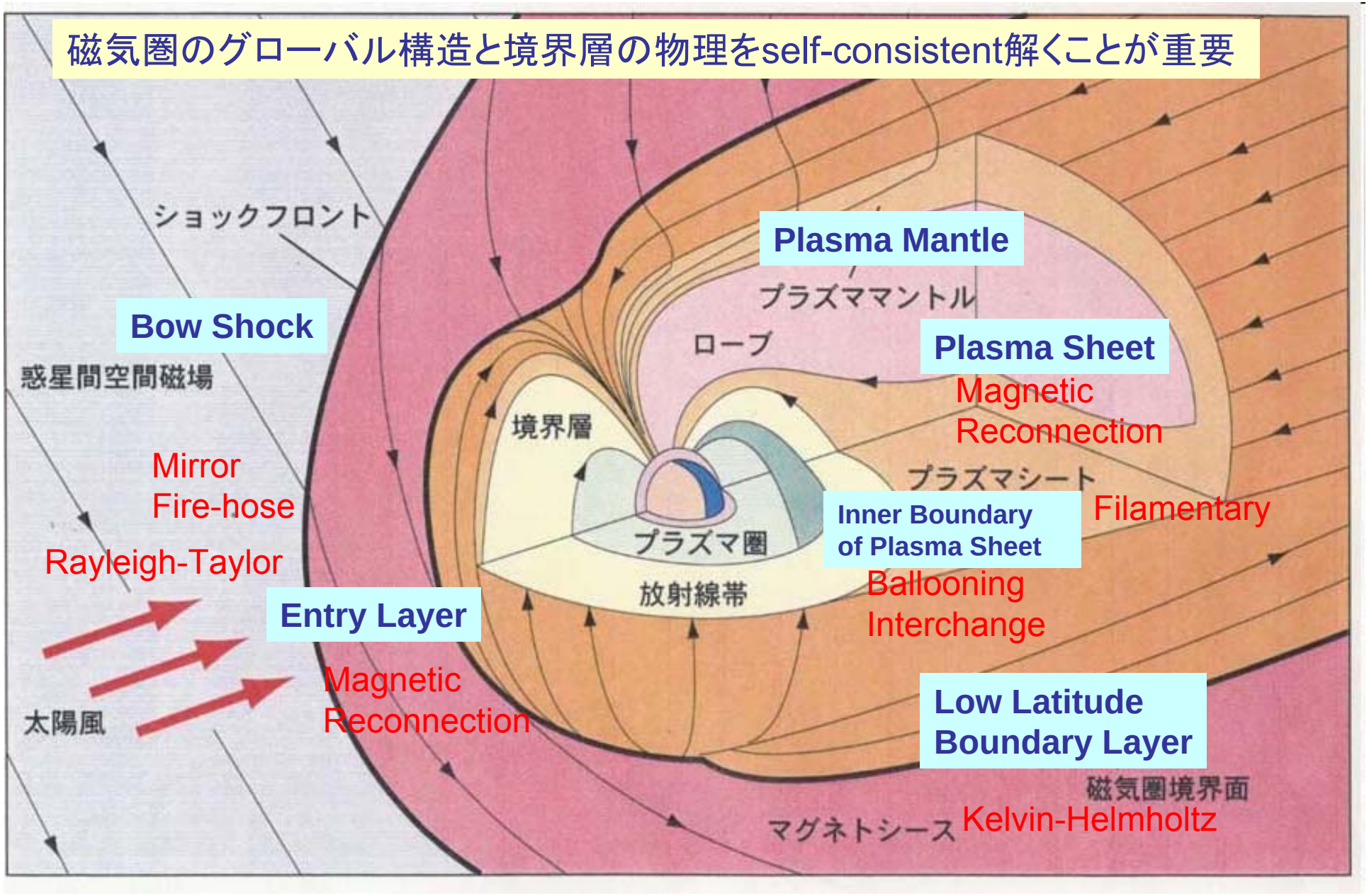
太陽風－磁気圏－電離圏・熱圏－大気圏結合システム

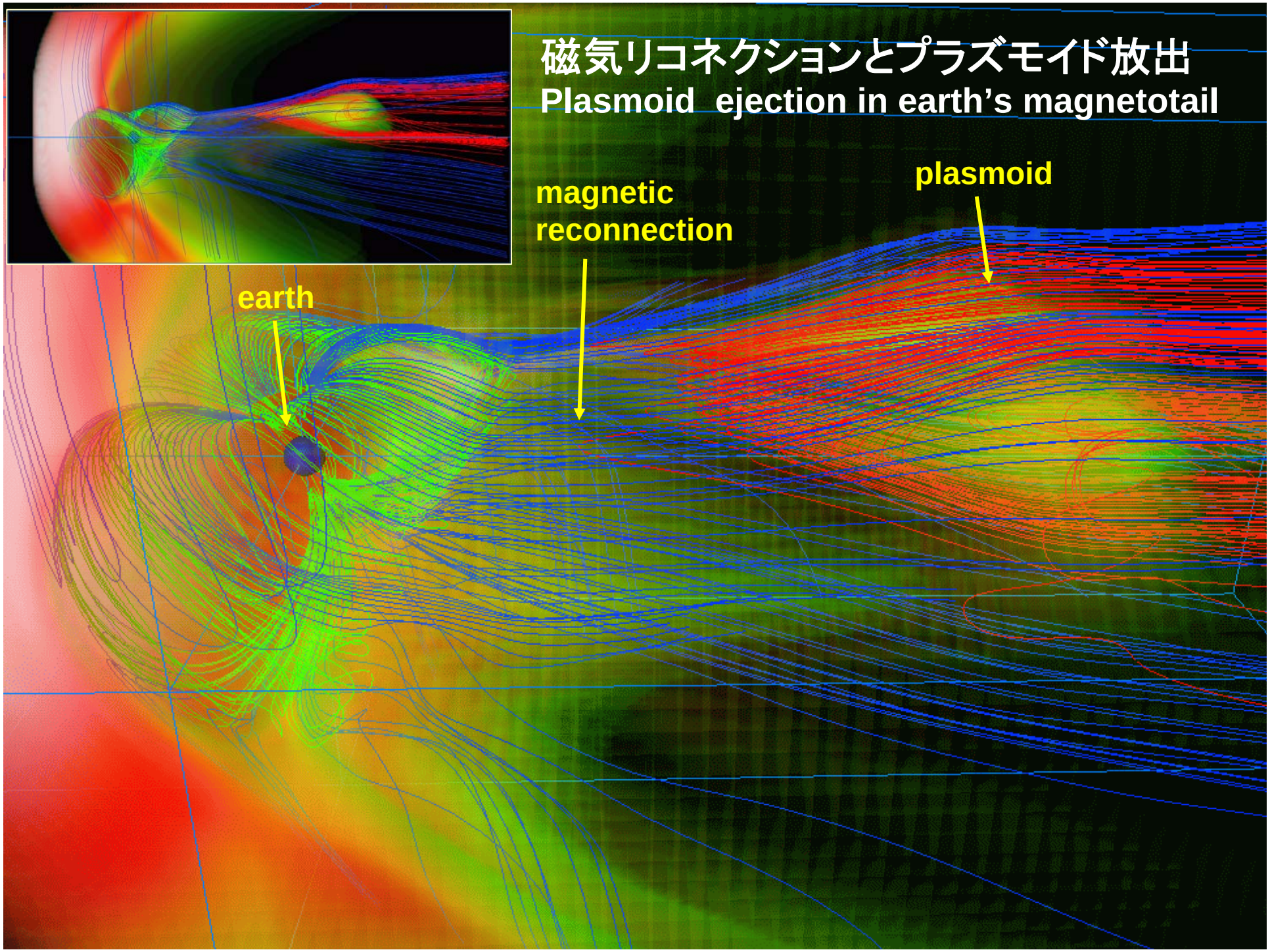


磁気嵐時の衛星オーロラ画像

北極域のオーロラ

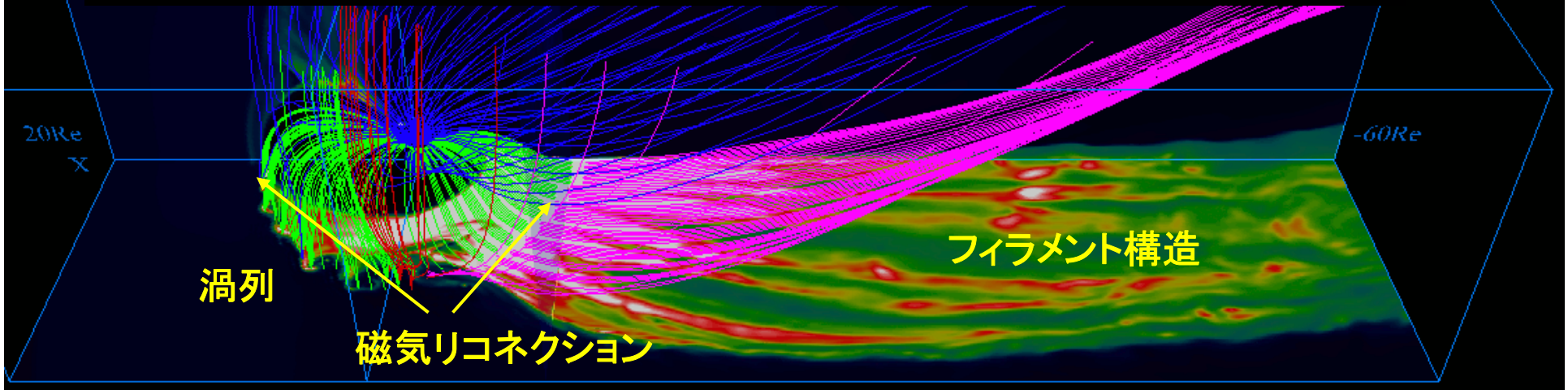
地球磁気圏の境界層とグローバルシミュレーション



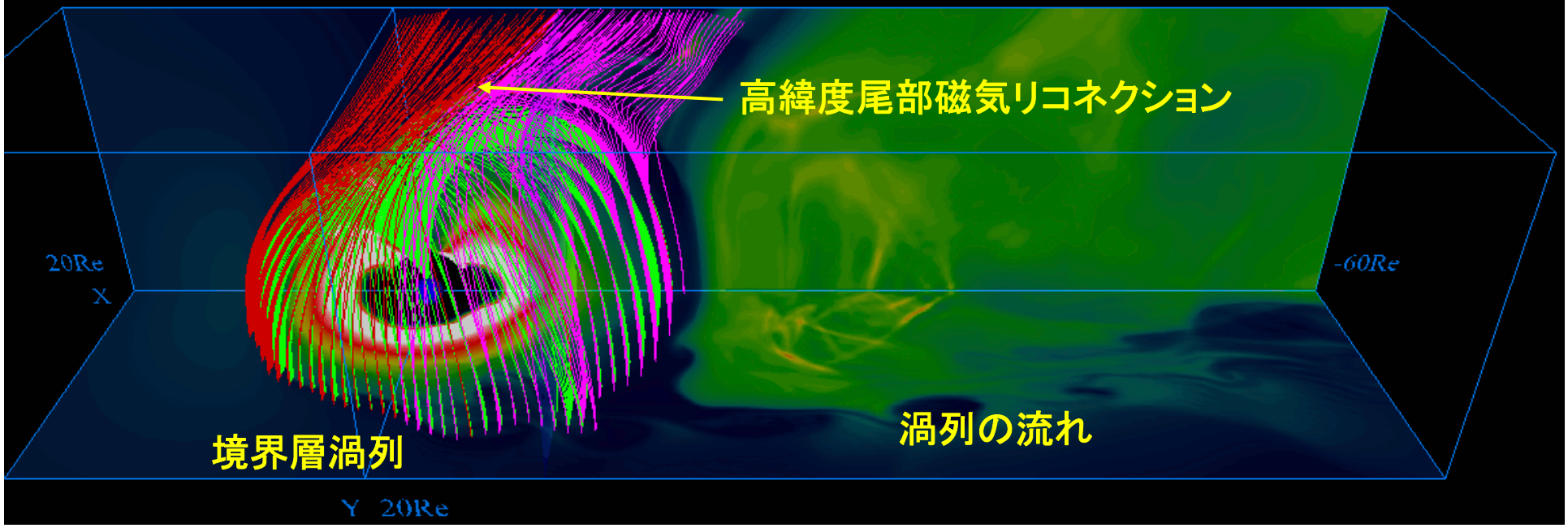

磁気リコネクションとプラズモイド放出
Plasmoid ejection in earth's magnetotail
magnetic
reconnection
plasmoid
earth

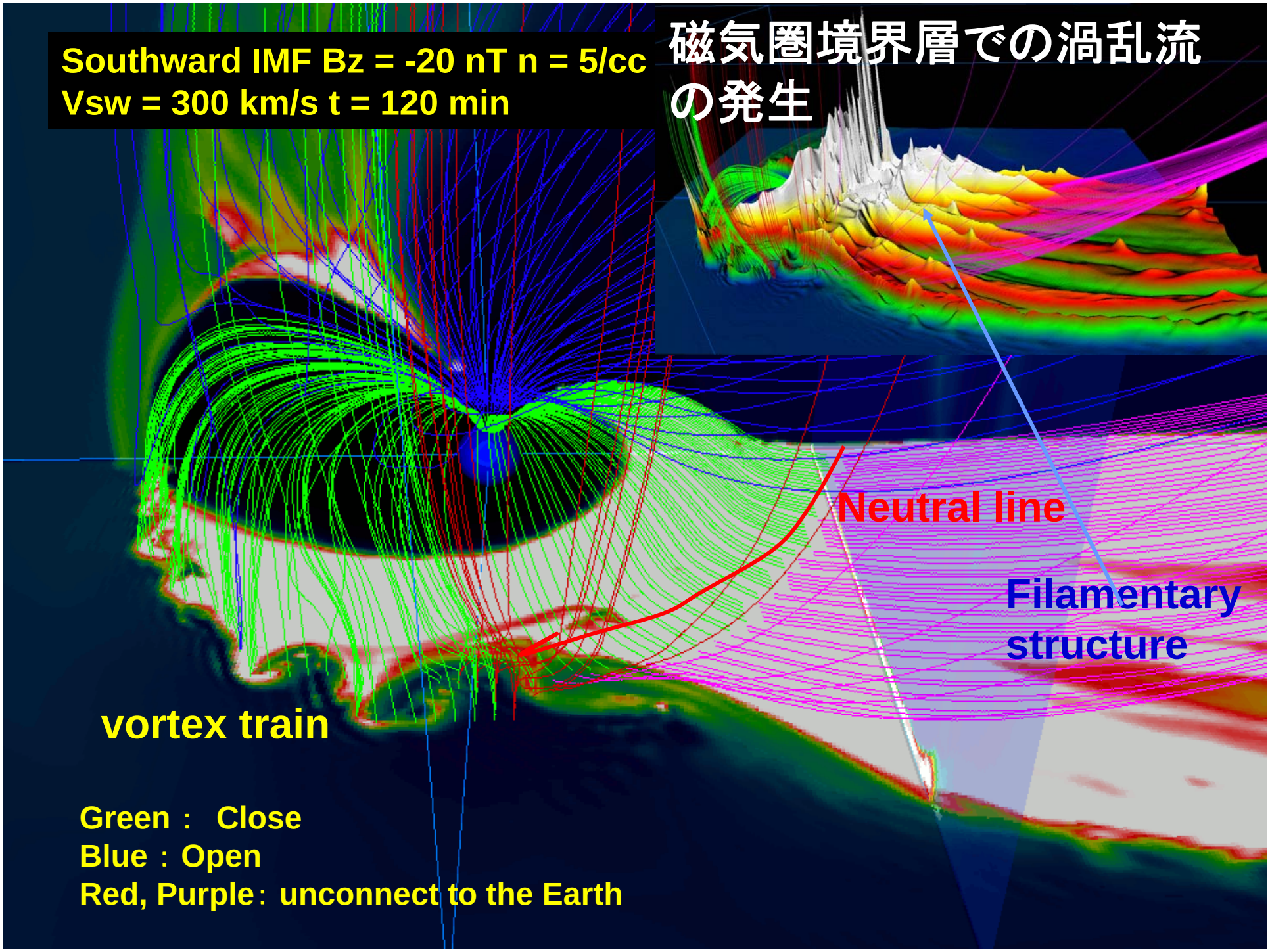
磁気圏境界層での渦乱流の発生とプラズマの異常輸送

Southward IMF Bz = -20 nT n = 5/cc Vsw = 300 km/s t = 120 min

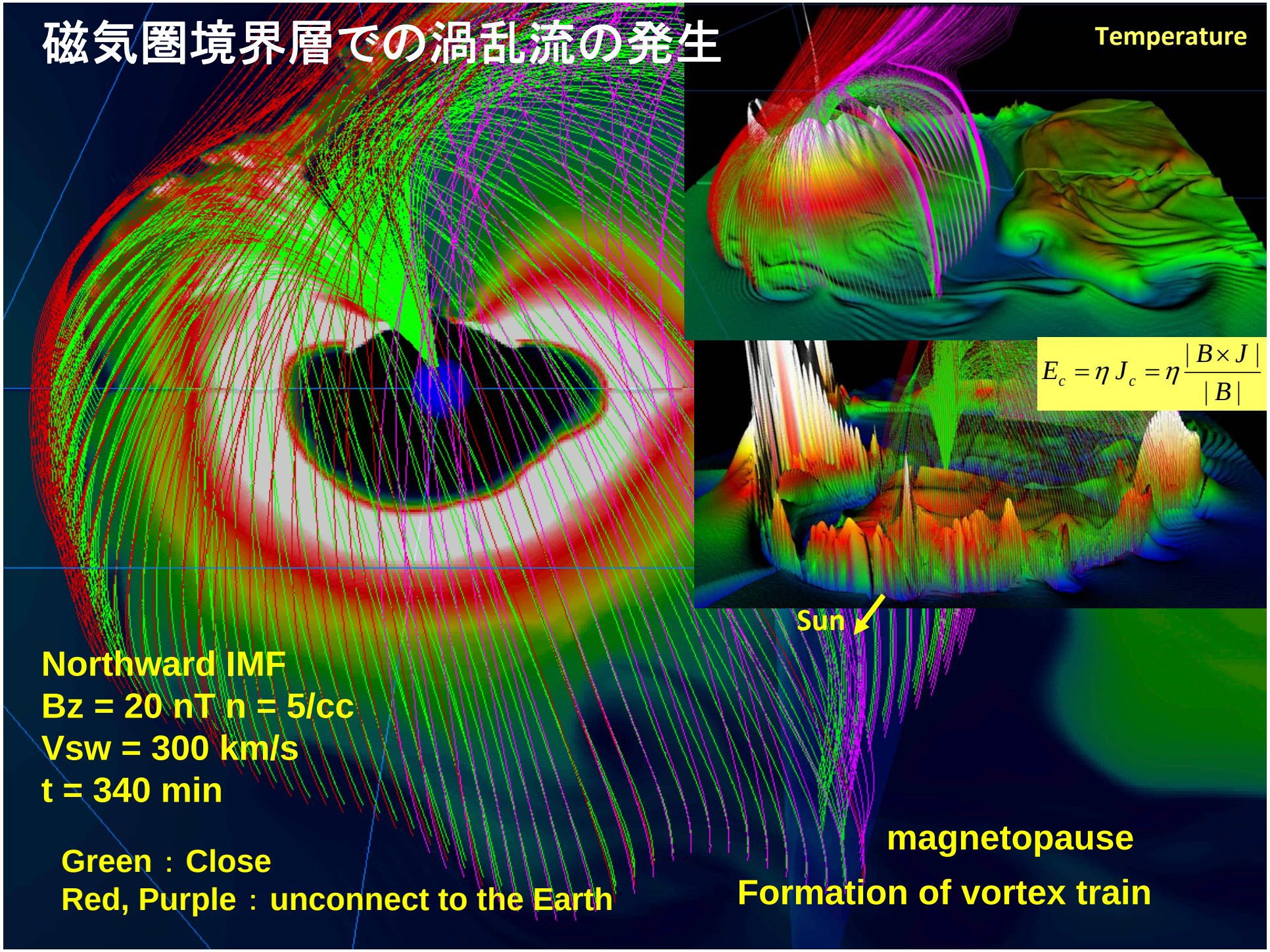


Northward IMF Bz = 20 nT n = 5/cc Vsw = 300 km/s t = 340 min



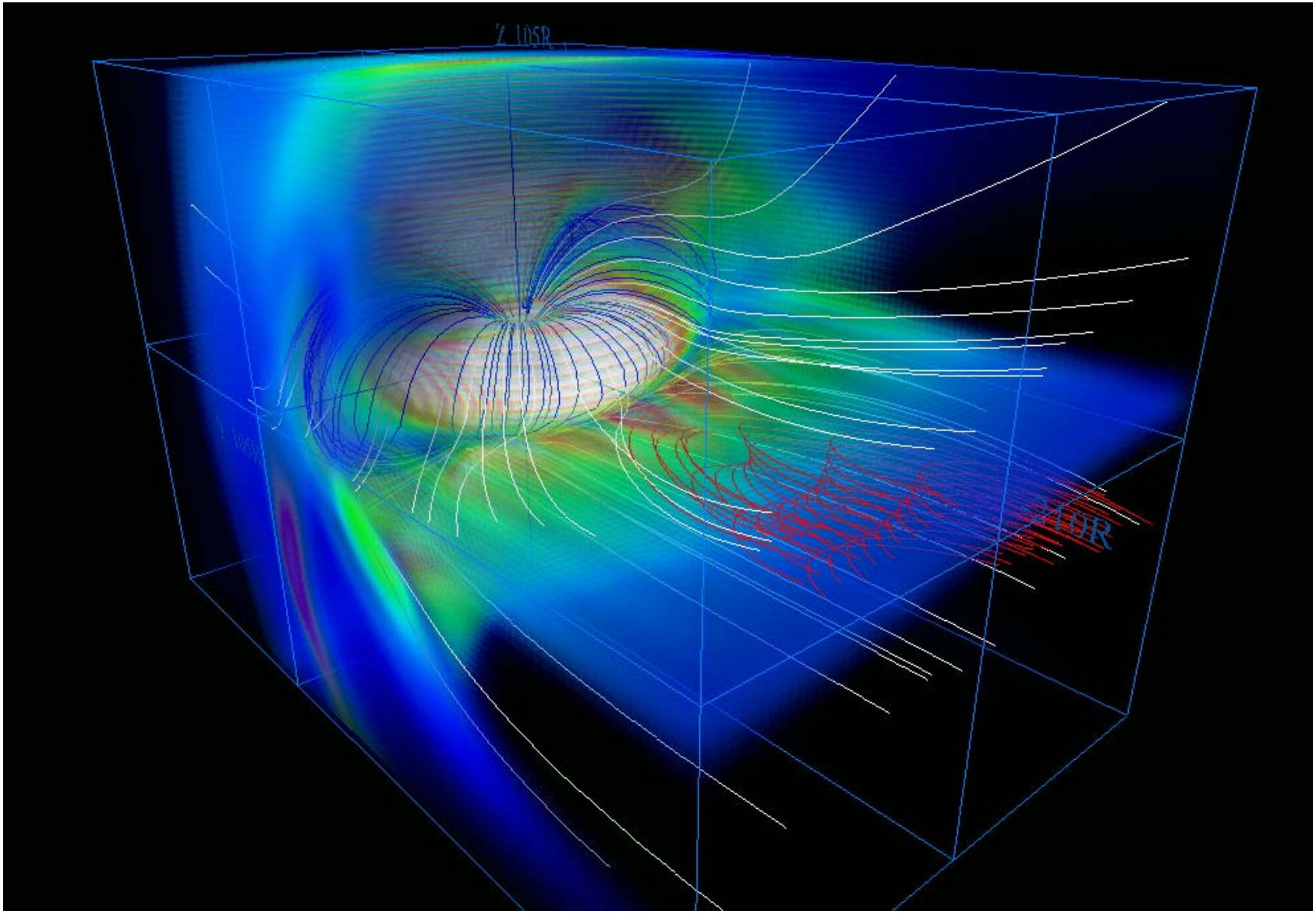

Southward IMF Bz = -20 nT n = 5/cc
Vsw = 300 km/s t = 120 min
磁気圏境界層での渦乱流の発生
Neutral line
Filamentary structure
vortex train
Green： Close
Blue：Open
Red, Purple：unconnect to the Earth

磁気圏境界層での渦乱流の発生



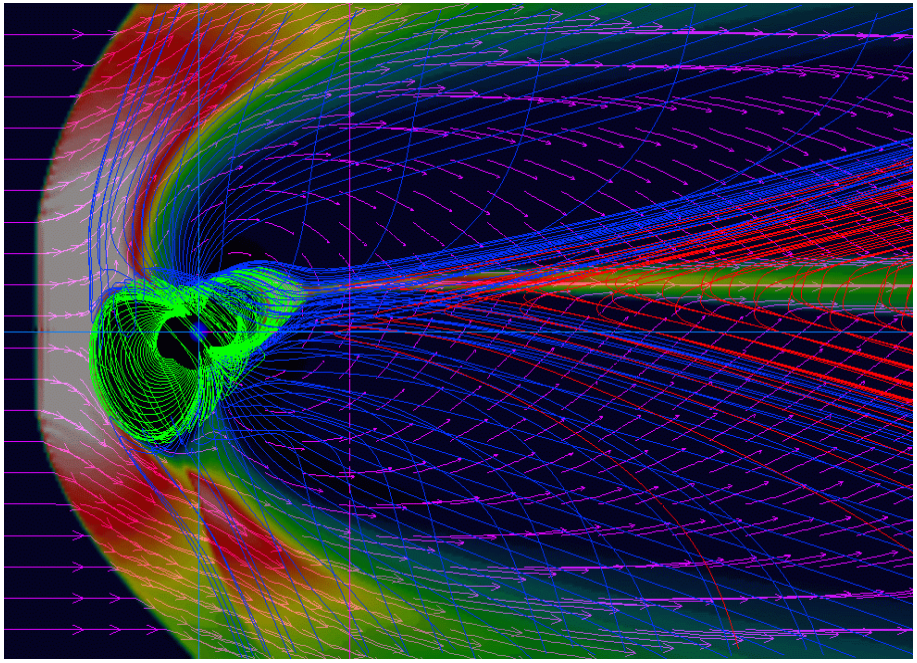
高速回転する木星磁気圏でのプラズモイドの構造

Patchy plasma sheet with IMF By and Bz

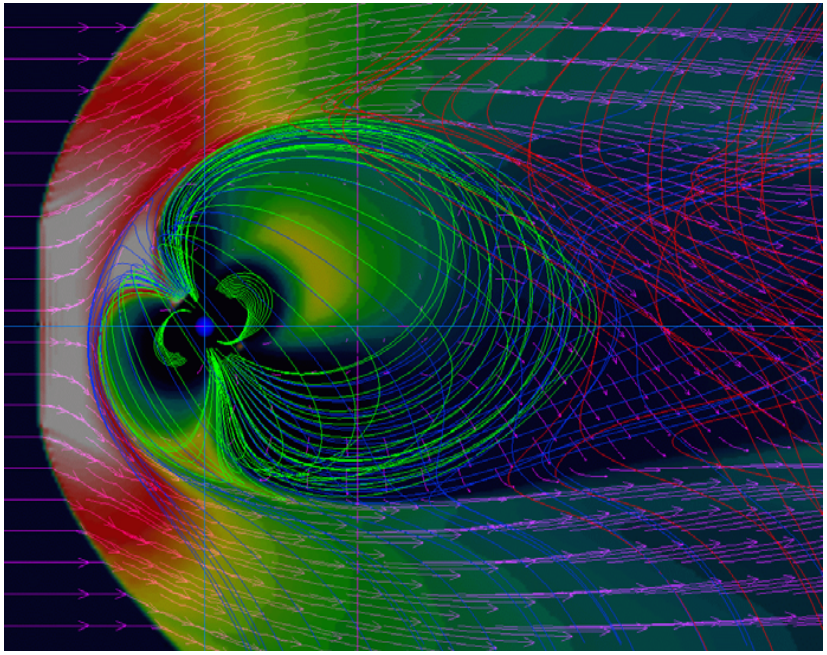


２．地球磁気圏の３次元グローバル MHDシミュレーションモデル

南向き惑星間磁場



北向き惑星間磁場



基礎方程式：MHD方程式とMaxwell方程式

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\mathbf{v}\rho) + D\nabla^2 \rho \qquad (1)$$

$$\frac{\partial \mathbf{v}}{\partial t} = -(\mathbf{v} \cdot \nabla)\mathbf{v} - \frac{(\nabla p)}{\rho} + \frac{\mathbf{J} \times \mathbf{B}}{\rho} + g + \frac{\mathbf{\Phi}}{\rho} \qquad (2)$$

$$\frac{\partial p}{\partial t} = -(\mathbf{v} \cdot \nabla) p - \gamma p \nabla \cdot \mathbf{v} + D_p \nabla^2 p \qquad (3)$$

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{v} \times \mathbf{B}) + \eta \nabla^2 \mathbf{B} \qquad (4)$$

$$\mathbf{J} = \nabla \times (\mathbf{B} - \mathbf{B}_d) \qquad (5)$$

スカラー並列機で高速計算を実現するために

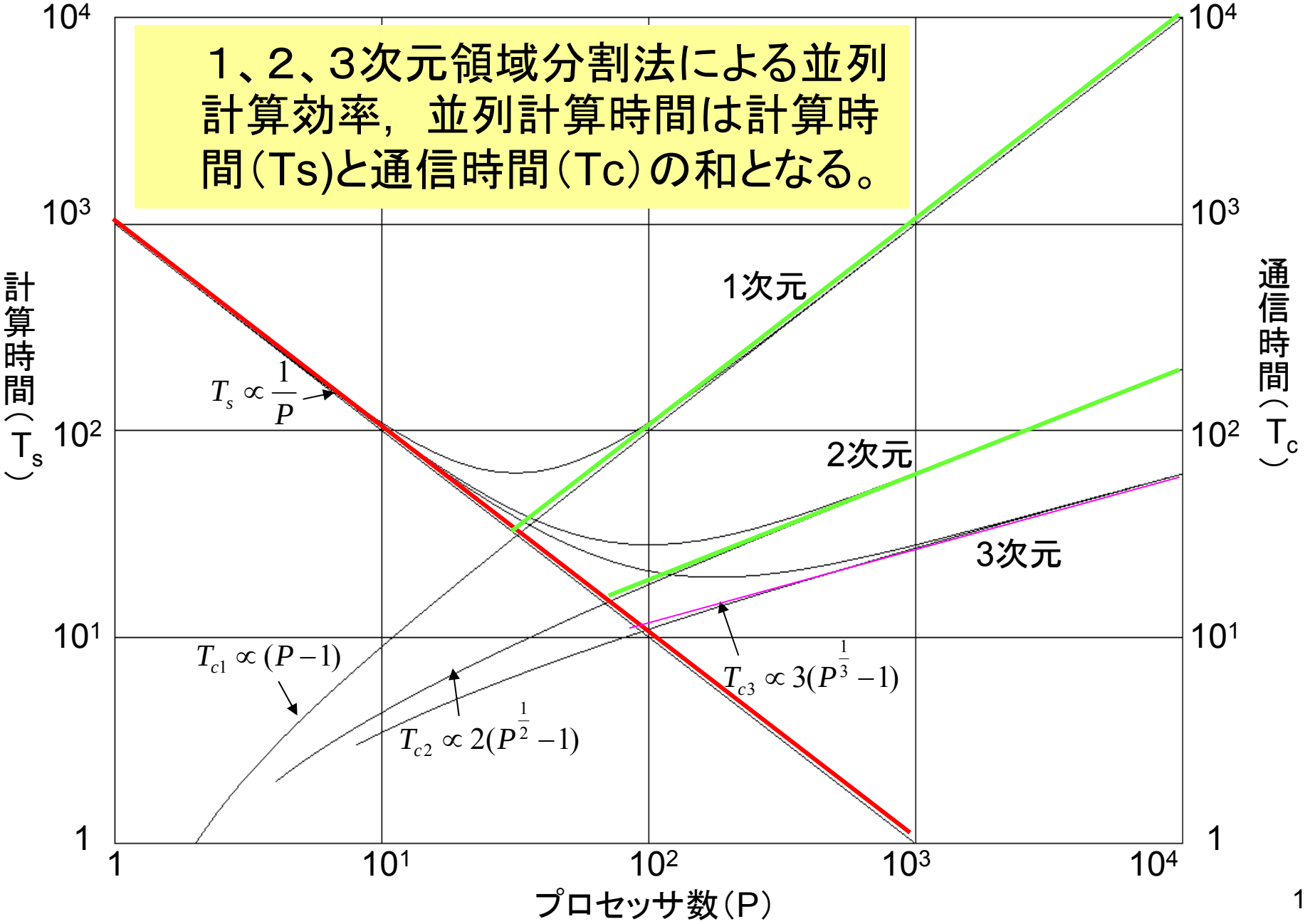
並列計算のcpu数が大幅に増加
通信量を大幅に減らす
キャッシュのヒット率を上げる

3次元MHDコードの領域分割法による並列化

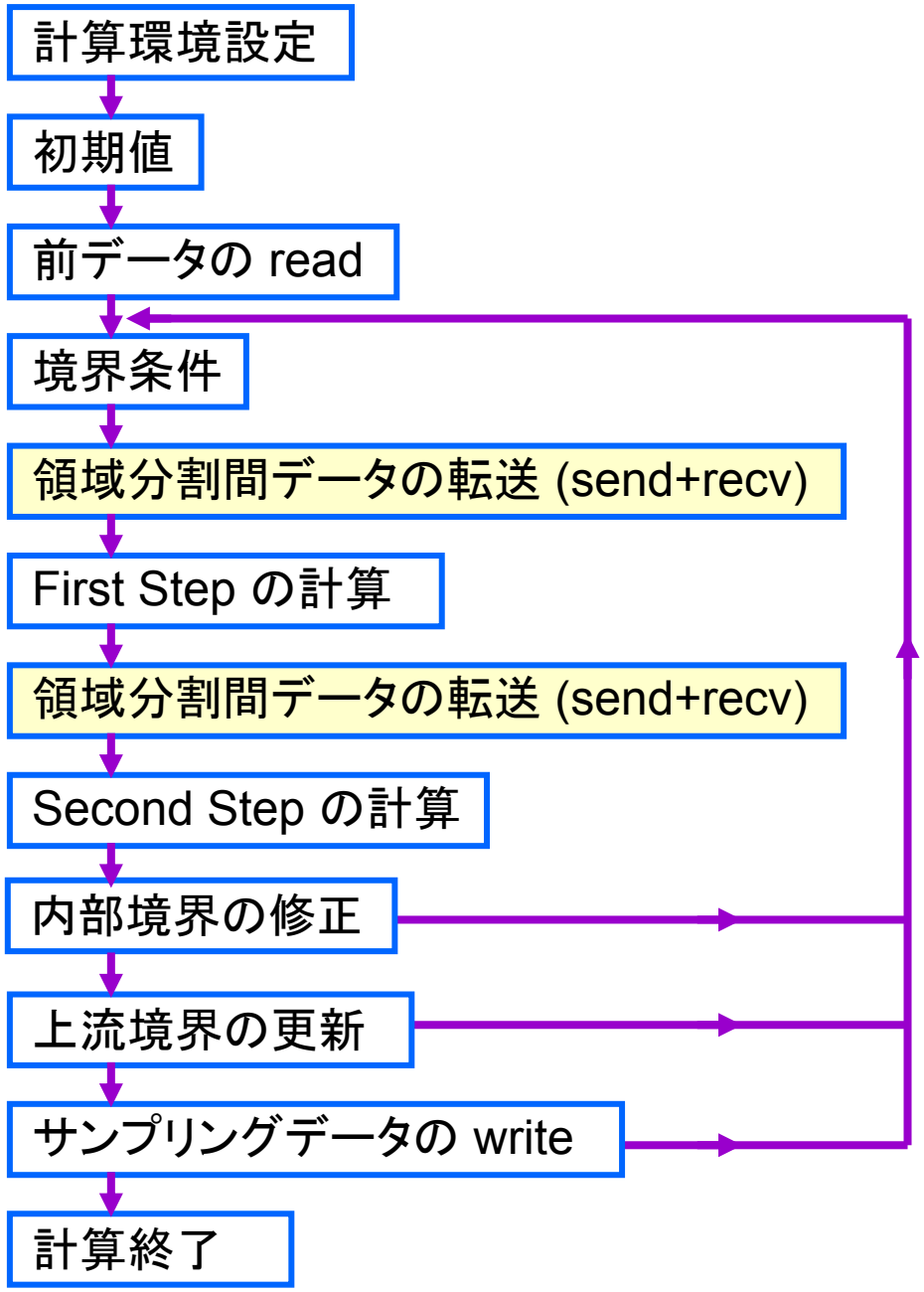
	計算時間	通信時間
1次元領域分割	$Ts=k_1 N^3 / P$	$Tc=k_2 N^2 (P-1)$
2次元領域分割	$Ts=k_1 N^3 / P$	$Tc=2k_2 N^2 (P^{1/2}-1)$
3次元領域分割	$Ts=k_1 N^3 / P$	$Tc=3k_2 N^2 (P^{1/3}-1)$

k1とk2:　一定の係数
N:　3次元配列の1方向の変数量
P:　並列CPUの数

領域分割による並列計算効率



MPIを用いたMHDコードの構造



CPU間のデータ転送を2箇所に集めて一括転送

その場合、
条件文と
do loop文を
不使用

３次元領域分割の方法

一括転送の２次元バッファを導入

```
CC MPI START
      parameter (npex=2,npey=2,npez=2)
      parameter (npe=npex*npey*npez,npexy=npex*npey)
      integer itable(-1:npex,-1:npey,-1:npez)
c
      parameter(nzz=(nz2-1)/npez+1)
      parameter(nyy=(ny2-1)/npey+1)
      parameter(nxx=(nx2-1)/npex+1)
      parameter(nxx3=nxx+2,nyy3=nyy+2,nzz3=nzz+2)
c
      dimension f(nb,0:nxx+1,0:nyy+1,0:nzz+1)
      dimension ftemp1x(nb,nyy3,nzz3),ftemp2x(nb,nyy3,nzz3)
      dimension ftemp1y(nb,nxx3,nzz3),ftemp2y(nb,nxx3,nzz3)
      dimension ftemp1z(nb,nxx3,nyy3),ftemp2z(nb,nxx3,nyy3)
c
```

領域分割されたCPU間のデータ転送1

```
c
      do k=-1,isizez
      do j=-1,isizey
      do i=-1,isizex
      itable(i,j,k)=MPI_PROC_NULL
      end do
c
      irank1=0
      do k=0,isizez-1
      do j=0,isizey-1
      do i=0,isizex-1
      itable(i,j,k)=irank1
      if(irank.eq.irank1) then
      irankx=i
      iranky=j
      irankz=k
      end if
      irank1=irank1+1
      end do
c
```

MPIで両端と中間のCPU間の転送を同じ形にするために、不実行を宣言する3次元のテーブルを作成

itable(-1:npex,-1:npey,-1:npez)

```
CC MPI START
      irightx = itable(irankx+1,iranky,irankz)
      ileftx  = itable(irankx-1,iranky,irankz)
      irighty = itable(irankx,iranky+1,irankz)
      ilefty  = itable(irankx,iranky-1,irankz)
      irightz = itable(irankx,iranky,irankz+1)
      ileftz  = itable(irankx,iranky,irankz-1)
C
CC MPI END
```

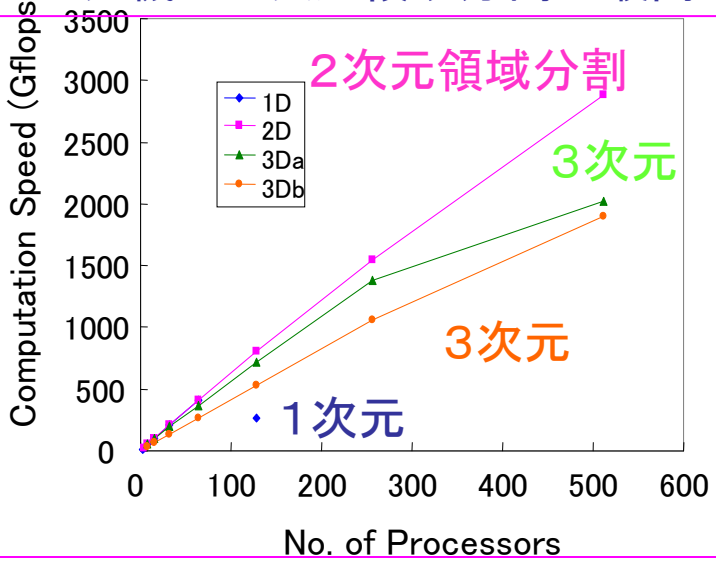
領域分割されたCPU間のデータ転送2

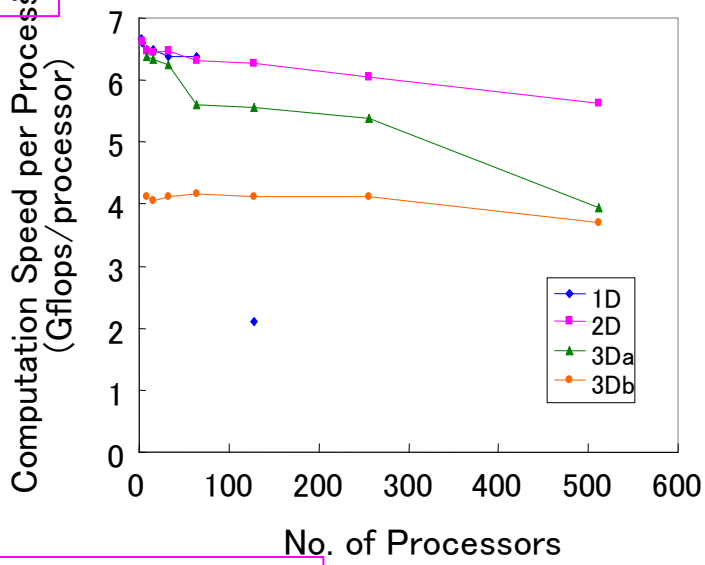
同期型の一括転送を利用

```
CC MPI START
      ftemp1x=f(:,is,:,:)
      ftemp1y=f(:,:,js,:)
      ftemp1z=f(:,:,:,ks)
      call mpi_sendrecv(ftemp1x,nwyz,mpi_real,ileftx,200,
     &                  ftemp2x,nwyz,mpi_real,irightx,200,
     &                  mpi_comm_world,istatus,ier)
      call mpi_sendrecv(ftemp1y,nwzx,mpi_real,ilefty,210,
     &                  ftemp2y,nwzx,mpi_real,irighty,210,
     &                  mpi_comm_world,istatus,ier)
      call mpi_sendrecv(ftemp1z,nwxy,mpi_real,ileftz,220,
     &                  ftemp2z,nwxy,mpi_real,irightz,220,
     &                  mpi_comm_world,istatus,ier)
C
      f(:,ie+1,:,:)=ftemp2x
      f(:,:,je+1,:)=ftemp2y
      f(:,:,:,ke+1)=ftemp2z
CC MPI END
```

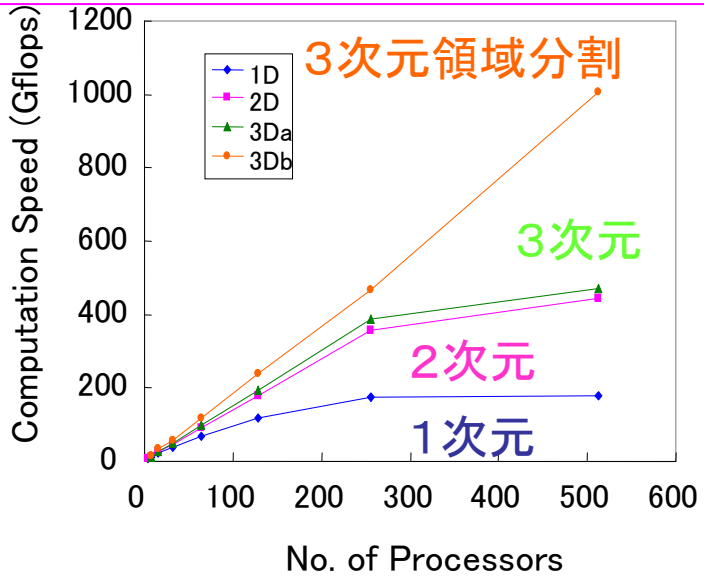
ベクトル並列機（Earth Simulator）とスカラー並列機 PRIMEPOWER HPC2500での計算速度の比較

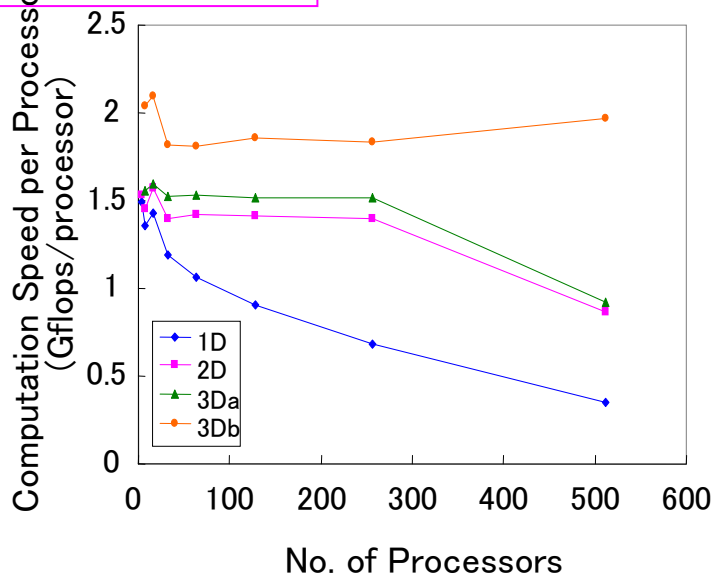
ベクトル並列機： ２次元領域分割が最高効率





スカラー並列機:： ３次元領域分割で配列変更が最高効率





3. PRIMEPOWER HPC2500から FX1とHX600への移行

（名古屋大学情報基盤センター）

システムS1
（M9000）

3ノード（SPARC）

システムS2
（HX600）

160ノード（Opteron）

システムS3
（FX1）

768ノード （SPARC）

名古屋大学情報基盤センターの新スーパーコンピュータ

システムS1　M9000
大規模な共有メモリ
3ノード x 128コア （1ノードはFX1のフロントエンド）
ノード当り：演算性能：1.28TFlops, メモリ：1TB
シミュレーション結果の解析と画像処理に利用

システムS2　HX600
クラスター型計算機（T2K型）ノード内は共有メモリ
160ノード x 16コア
ノード当り：演算性能：160GFlops, メモリ：64GB
中規模シミュレーションに利用

システムS3　FX1
大規模分散並列型、次世代スパコンとの連携
768ノード x 4コア
ノード当り：演算性能：40GFlops, メモリ：32GB
メモリバンド幅：40GB/s
大規模シミュレーションに利用

大規模シミュレーションはシステムS3 (FX1) Hybrid (Impact)の利用を推奨

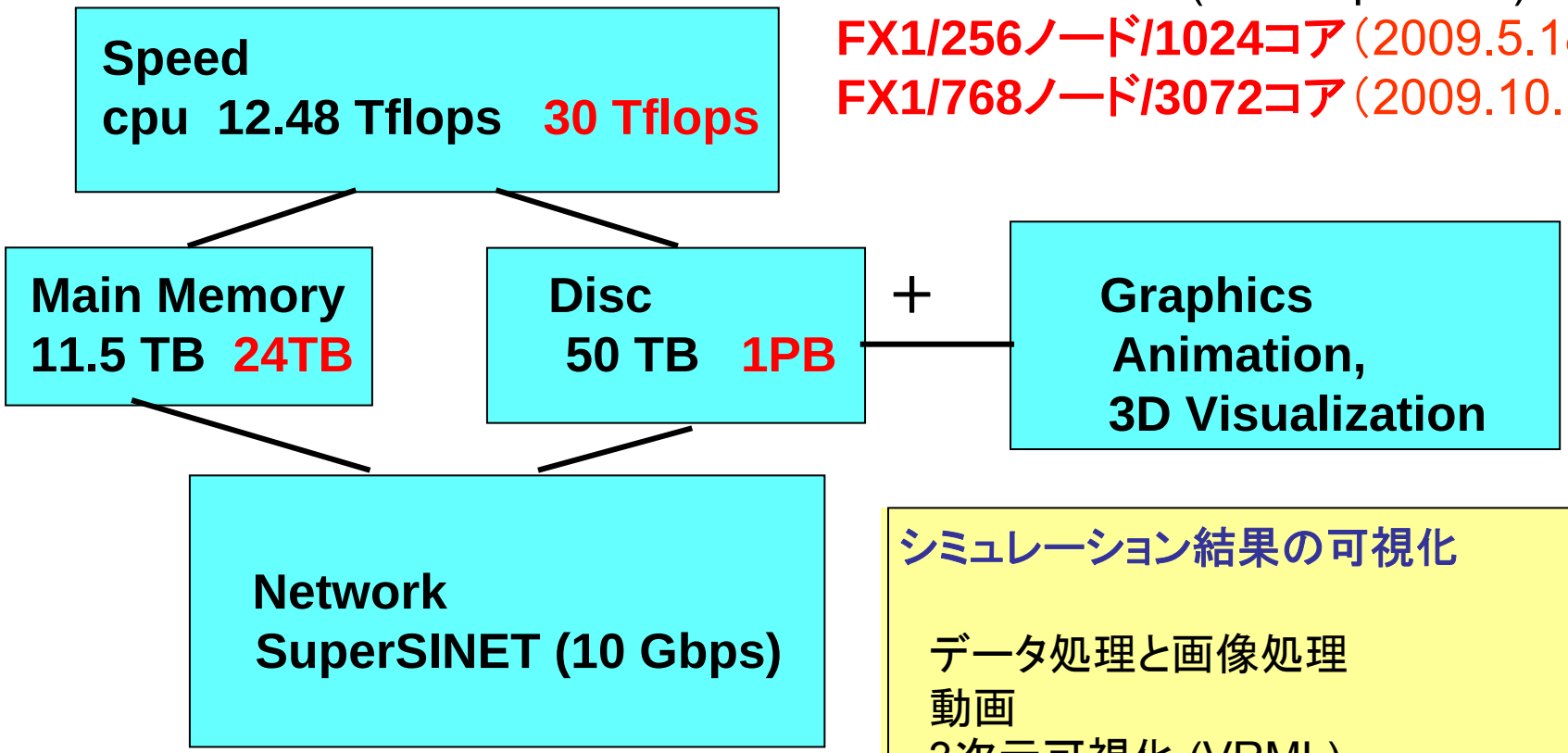
総ノード／コア数：768／3072
総演算性能：30.72TFlops
総メモリ：24TB

FX1利用の計算規模

コア数	256	512	1024
メモリ (TB)	2	4	8
演算性能 (Tflops)	2.56	5.12	10.24

Four Key Functions of Supercomputer

HPC2500/1536 (scalar-parallel)
FX1/256ノード/1024コア（2009.5.18）
FX1/768ノード/3072コア（2009.10.1）



シミュレーション結果の可視化

データ処理と画像処理
動画
3次元可視化 (VRML)
VR (Virtual Reality)

スパコンで行うか、WSやPCで行うか？
巨大なデータの可視化をどうするか

スパコンでの高効率並列計算の実現
スパコンの最大性能を引き出すプログラム開発
絶対性能の10-20%程度以上の演算速度で
よいスケーラビリティが保たれることが目標

Flat MPIとHybrid（user並列＋自動並列）はどちらが最速か？

Hybridでは

「Node内はベンダーが最高効率の自動並列化を提供する

Node間はユーザーが最高効率のプロセス並列プログラムを作成する

この結果、最高効率のプログラムが得られるはずである？？？」

この命題は正しいか？

それともFlat MPIが最速か？

Hybridではノード内の並列化とノード間の並列化に齟齬は生じないのか？

- メモリ上に変数がどの様に配置されるのか？
- ノード間でどの様に領域分割して、分散メモリに配分しているかは本質的に重要
- ノード内でどの様に領域分割して、メモリに割り付けるか、疑問？（ノード内自動並列化はどうしているのか？）
- **キャッシュのヒット率向上と通信量の最小化**

Flat MPIでは全部の配列変数をどの様にメモリに割り付けるかを指定できる

通信量の最小化とキャッシュのヒット率をトータルで最適化

ただし、高度な技術が必要

そのテンプレート（プログラムの構造）を作れば誰でも利用できるはず

もちろん、**Flat MPIとHybridはどちらが最速か**はスパコンによる

- Fujitsu FX1は1node 4 coreで、1 node内では共有メモリを持つ
- 従って、Flat MPIとHybrid（Impact）はどちらが最速か、検証が必要

完全な分散型スパコンであれば、Flat MPIが最速になると一般に信じてられている。
その他の全てのスパコンでFlat MPIが最速になると予想している。**これも現在までの経験からの予想**

通信量最小化＋並列化オーバーヘッド（Flat MPI）と**共有メモリ間衝突とキャッシュのヒット率変化**（Hybrid）はトレードオフの関係

スーパースカラー並列スパコンで最高効率の並列プログラムを作成するには

1．先ず、プログラムの構造の確立が最も重要

・計算での変数参照を元にプログラム構造を決定
　配列変数定義の最小化
　計算の明確なブロック化
　計算と通信の完全な分離
　定数と変数は意識して区別（dependencyを解く）
　　a(i)=a(i+j)の形で j は定数で定義

・　配列の大きさ、do loopの回数（初数と終数）、
　配列内の変数は定数で定義
　　定数定義はparameter文を利用

スーパースカラー並列スパコンで最高効率の並列プログラムを作成するには（その2）

・領域分割法の導入（2、3次元領域分割）
　通信では一括転送を実施
・Dependencyを可能な限り取り除く
　計算のメイン部分ではIf文や条件文は絶対に使用不可
　定数定義の利用
　Mask、List vector、Gather and scatterの最適な使用
・変数の一括変換と一括転送
　これを条件文やdo loopを導入せずに実行

If文や条件文が並列化（擬似ベクトル化）できたとの表示にだまされてはいけない。⟶ その効率は実際非常に悪い。
（必ず実測すること）

４． FX1とHX600おける３次元MHDコードの計算速度

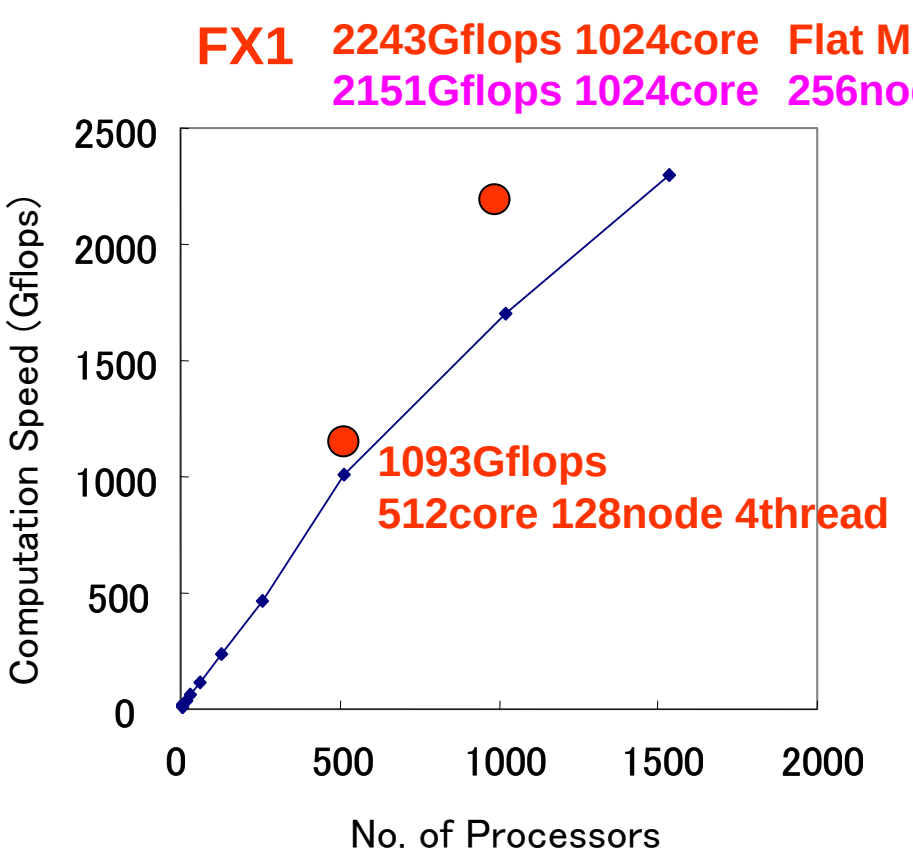
コンピュータ	グリッド数	総CPU CORE 数	プロセス並列数	自動並列数	CPU 時間 (sec)	1グリッド当りの計算時間 (nsec)	1グリッド当りの計算時間 x 総CPU CORE数
HPC2500	1024*1024*1024	512	512	1	2.487	2.316	1186
HPC2500	2048*2048*2048	1024	512	2	10.763	1.253	1283
VPP5000	1002*1002*1120	56	56	1	5.794	5.152	289
SX6	512*256*256	16	16	1	0.693	20.662	331
Earth Sim	2048*1024*1024	1024	1024	1	0.62	0.289	296
HX600	512*512*512	64	64	1	1.451	10.813	692
HX600	1024*1024*1024	512	512	1	1.906	1.775	909
HX600	2048*2048*2048	512	512	1	19.654	2.288	1171
FX1	512*512*512	64	64	1	1.337	9.964	638
FX1	1024*1024*1024	512	512	1	1.377	1.283	657
FX1	1024*1024*1024	1024	1024	1	0.752	0.701	684
FX1	2048*2048*2048	512	512	1	10.525	1.225	627
FX1	2048*2048*2048	1024	1024	1	5.215	0.607	622
SR16000	508*508*508	64	64	1	1.236	9.424	603
SR16000SMT	508*508*508	32	64	1	1.239	9.448	605
SR16000	1018*1018*510	256	256	1	1.256	2.376	608
SR16000SMT	1018*1018*1018	256	512	1	1.29	1.223	626

６列目：Tc：１回時間ステップを進めるのに要するCPU時間（sec）

７列目：Tcg：１グリッドにつき１回時間ステップを進めるのに要するCPU時間（nsec）

８列目：Tcg*Ng：Tcgと総CPU/CORE数の積（nsec）、数値が小さいほど高効率計算を示す。

PRIMEPOWER HPC2500とFX1での計算速度の比較



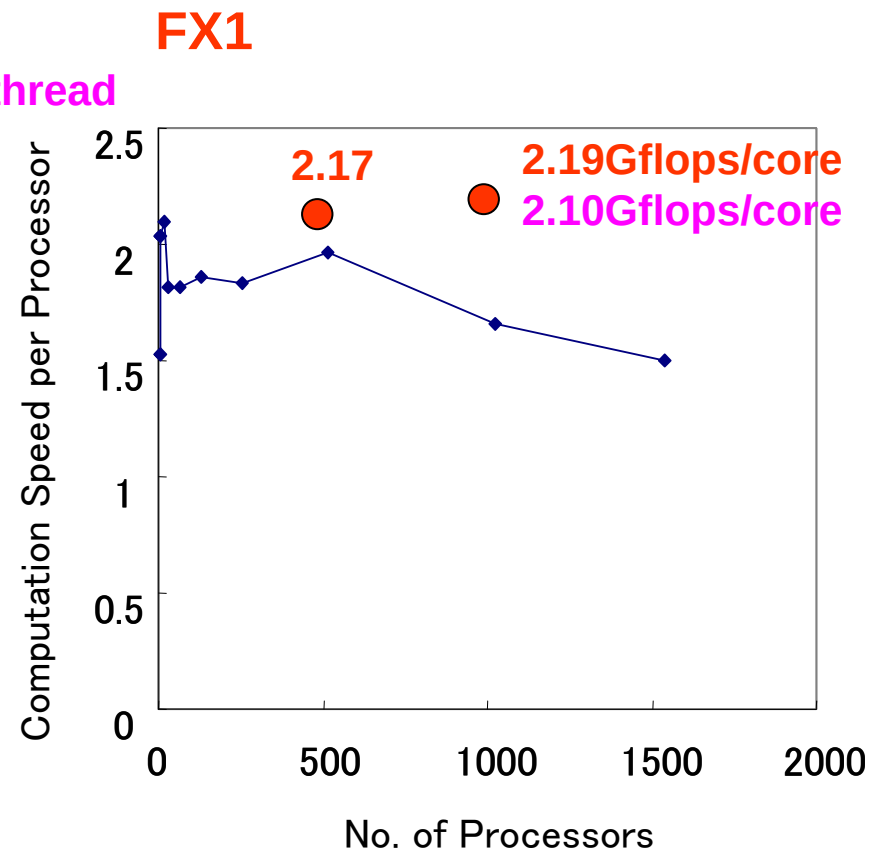


図6 PRIMEPOWER HPC2500でCPU数を増やした場合の計算速度（Gflops）と1CPU当たりの計算速度（Gflops/cpu）。・1536 cpuまでスケーラビリティは非常に良い

Fujitsu FX1は、3次元領域分割とキャッシュヒットを改善した3次元MHDコードで128node, 512coreで22%が得られ、256node, 1024coreで21%の高効率計算.

大粒度計算での最高速は1024core Flat MPIで21%を達成。

FX1の計算速度 1

コンピュータ	プログラム	グリッド数	総コア数	プロセス並列数	自動並列数	CPU時間 (sec)	計算速度 (Gflops)	効率 (%)
FX1	1Da	1024**3	256	64	4	152.0	9	0.4
FX1	1Da	1024**3	128	128	1	232.2	6	0.5
FX1	1Da	1124*1024**2	128	128	1	28.4	52	4.1
FX1	2Da	1024**3	128	128	1	17.1	79	6.2
FX1	2Da	1124*1024**2	128	128	1	8.23	180	14.1
FX1	3Da	1024**3	128	128	1	7.32	185	14.4
FX1	3Db	1024**3	128	128	1	5.22	259	20.2
FX1	2Da	3072*2048**2	3072	3072	1	17.2	1004	3.3
FX1	2Da	3072*2048**2	3072	768	4	19.7	822	2.7
FX1	3Da	3072*2048**2	3072	3072	1	3.91	4145	13.5
FX1	3Da	3072*2048**2	3072	768	4	4.36	2782	9.1
FX1	3Db	3072*2048**2	3072	3072	1	3.29	4932	16.1
FX1	3Db	3072*2048**2	3072	768	4	3.76	4321	14.1

1Da：1次元領域分割、2Da：2次元領域分割、3Da：3次元領域分割、3Db：3次元領域分割＋f(m,i,j,k)
グリッド数の選び方によっては極端に計算効率が劣化する場合があるが、少し調節すると改良できる。

FX1の計算速度 2

コンピュータ	プログラム	グリッド数	総コア数	プロセス並列数	自動並列数	CPU時間 (sec)	計算速度 (Gflops)	効率 (%)
FX1	3Db	1024**3	1024	1024	1	0.645	2096	20.5
FX1 (SMT)	3Db	1024**3	1024	1024	1	0.661	2043	20.0
FX1	3Db	1024**3	1024	258	4	0.732	1846	18.0
FX1 (SMT)	3Db	1024**3	1024	258	4	0.783	1810	17.7
FX1	3Db	1024**3	2048	2048	1	0.355	3813	18.6
FX1	3Db	1024**3	2048	512	4	0.380	3555	17.4
FX1 (SMT)	3Db	2048**3	1024	1024	1	5.09	2127	20.8
FX1	3Db	2048**3	2048	2048	1	2.72	3979	19.4
FX1	3Db	2048**3	2048	512	4	2.88	3752	18.3
FX1	3Db	3072* 2048**2	3072	3072	1	2.70	5997	19.5
FX1	3Db	3072* 2048**2	3072	768	4	2.92	5556	18.1

2Da：2次元領域分割、3Da：3次元領域分割、3Db：3次元領域分割＋f(m,i,j,k)
3Dbは計算効率が高い、またFlat MPIがHybrid（自動並列数=4）より僅かに高効率

FX1の計算速度 3

コンピュータ JAXAJAXA JAXA	プログラム	グリッド数	総コア数	プロセス並列数	自動並列数	CPU時間 (sec)	計算速度 (Gflops)	効率 (%)
FX1 (SMT)	3Db	2048**3	256	256	1	19.8	541	21.3
FX1 (SMT)	3Db	2048**3	1024	1024	1	5.09	2127	20.8
FX1	3Db	2048**3	2048	2048	1	2.72	3979	19.4
FX1	3Db	3072*2048**2	3072	3072	1	2.70	5997	19.5
FX1	3Db	2048**3	512	512	1	10.53	1028	20.1
	3Db	1024**3						
JAXA FX1	3Db	1024**3	1024	1024	1	0.701	1929	18.8
JAXA FX1	3Db	1024**3	4096	4096	1	0.220	6160	15.0
JAXA FX1	3Db	1024**3	4096	1024	4	0.211	6406	15.6
JAXA FX1	3Db	1000*1200**2	5760	5760	1	0.247	7338	12.7
JAXA FX1	3Db	1000*1200**2	5760	1440	4	0.228	7966	13.8

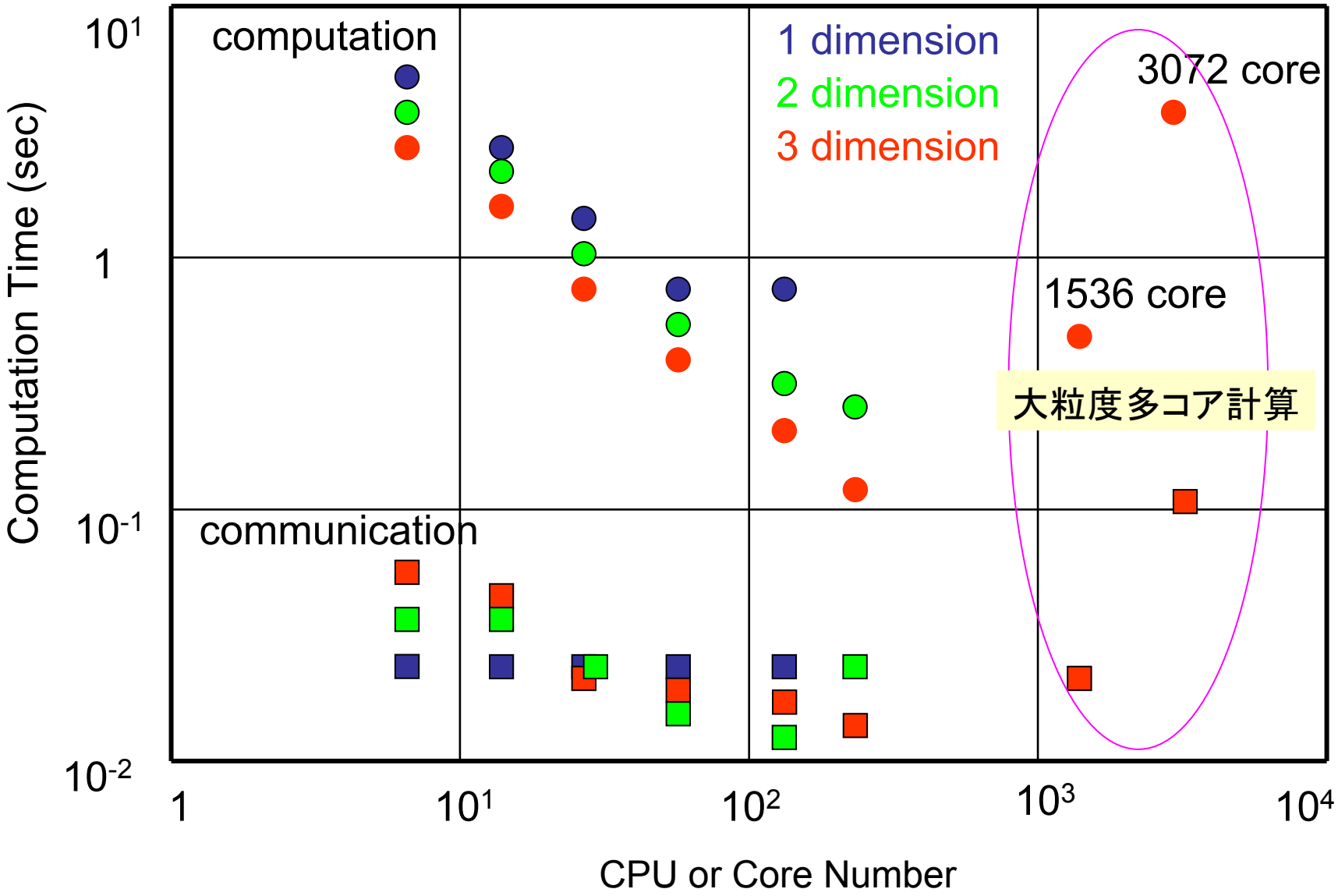
1Da：1次元領域分割、2Da：2次元領域分割、3Da：3次元領域分割、3Db：3次元領域分割＋f(m,i,j,k)
多コア数の計算も高効率、JAXA FX1では多コアでHybridがFlat MPIよりも高効率の場合もあった。

HX600の計算速度

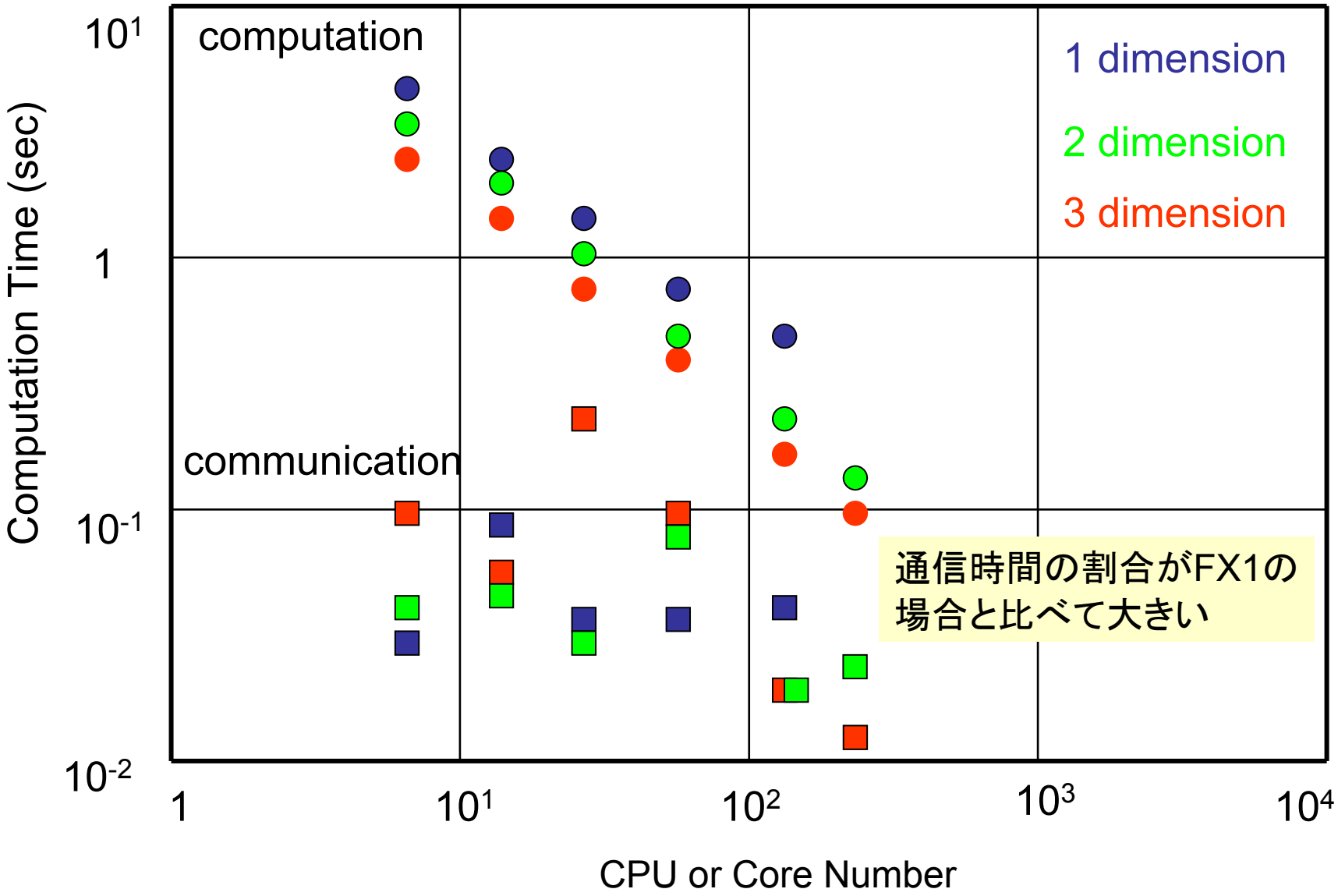
コンピュータ	プログラム	グリッド数	総コア数	プロセス並列数	自動並列数	CPU時間 (sec)	計算速度 (Gflops)	効率 (%)
HX600	2Da	1024**3	256	256	1	4.95	273	10.7
HX600	3Da	1024**3	256	256	1	3.56	379	14.8
HX600	3Db	1024**3	256	256	1	5.22	259	10.1
HX600	3Db	1024**3	2048	2048	1	0.464	2912	15.9
HX600	3Db	1024**3	2048	512	4	0.685	1972	9.6
HX600	3Db	1024**3	2048	128	16	0.894	1511	7.4
HX600	3Da	1024**3	2048	2048	1	0.406	3327	16.3
HX600	2Da	1024**3	2048	2048	1	0.674	2005	9.8
HX600	2Da	2048**3	2048	2048	1	6.27	1724	8.4
HX600	3Da	2048**3	2048	2048	1	3.54	3047	14.9
HX600	3Db	2048**3	2048	2048	1	4.99	2217	10.6
HX600	3Da	2048**3	2048	128	16	3.54	151	1.5

2Da：2次元領域分割、3Da：3次元領域分割、3Db：3次元領域分割＋f(m,i,j,k)
大粒度計算ではHX600はFX1より計算効率は低い、また、自動並列数を上げると計算効率がかなり低くなる。

FX1 Flat MPI

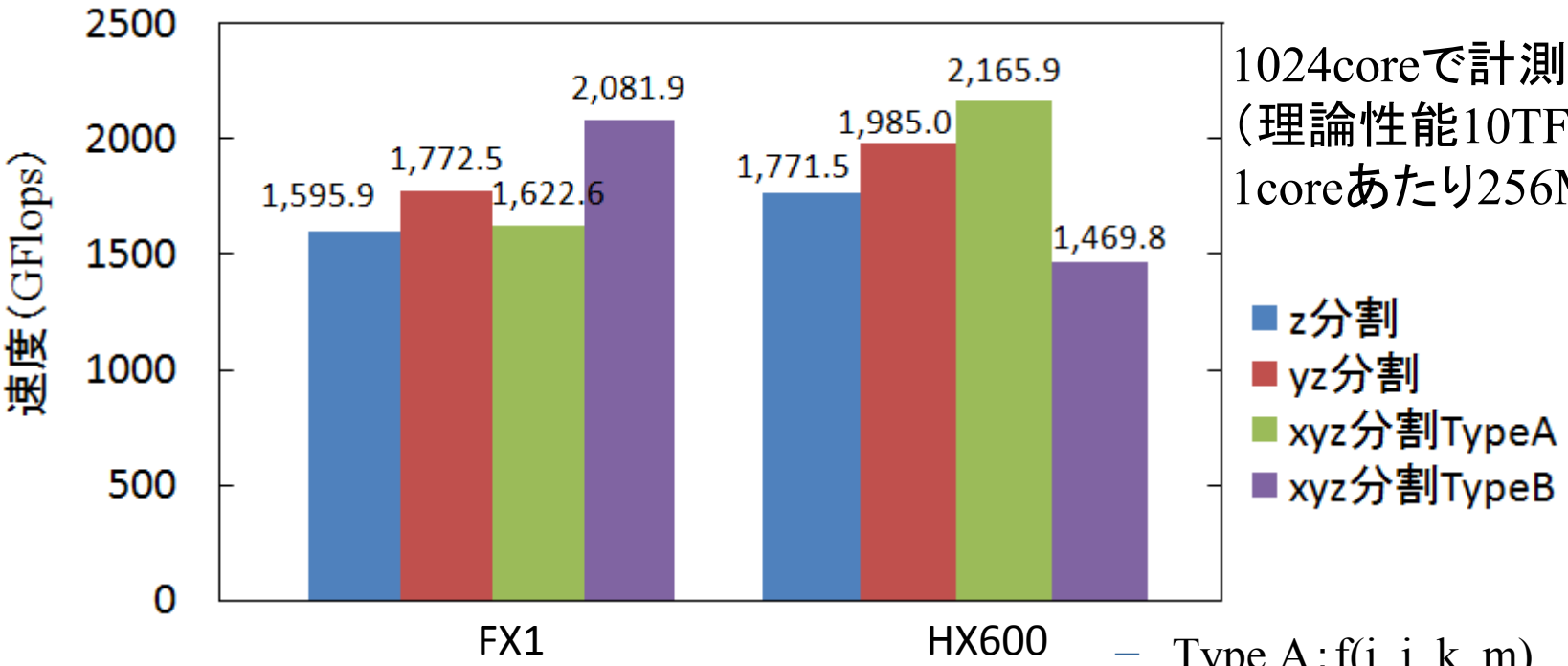

computation
communication
1 dimension
2 dimension
3 dimension
3072 core
1536 core
大粒度多コア計算
Computation Time (sec)
CPU or Core Number
10^1
1
10^{-1}
10^{-2}
1
10^1
10^2
10^3
10^4

HX600 Flat MPI



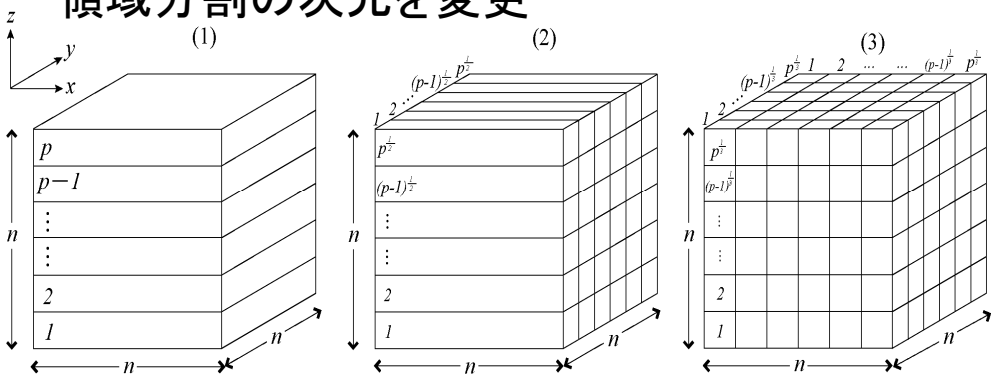
通信時間の割合がFX1の場合と比べて大きい

MHDコードの性能評価



1024coreで計測
（理論性能10TFlops）
1coreあたり256MBに設定

領域分割の次元を変更

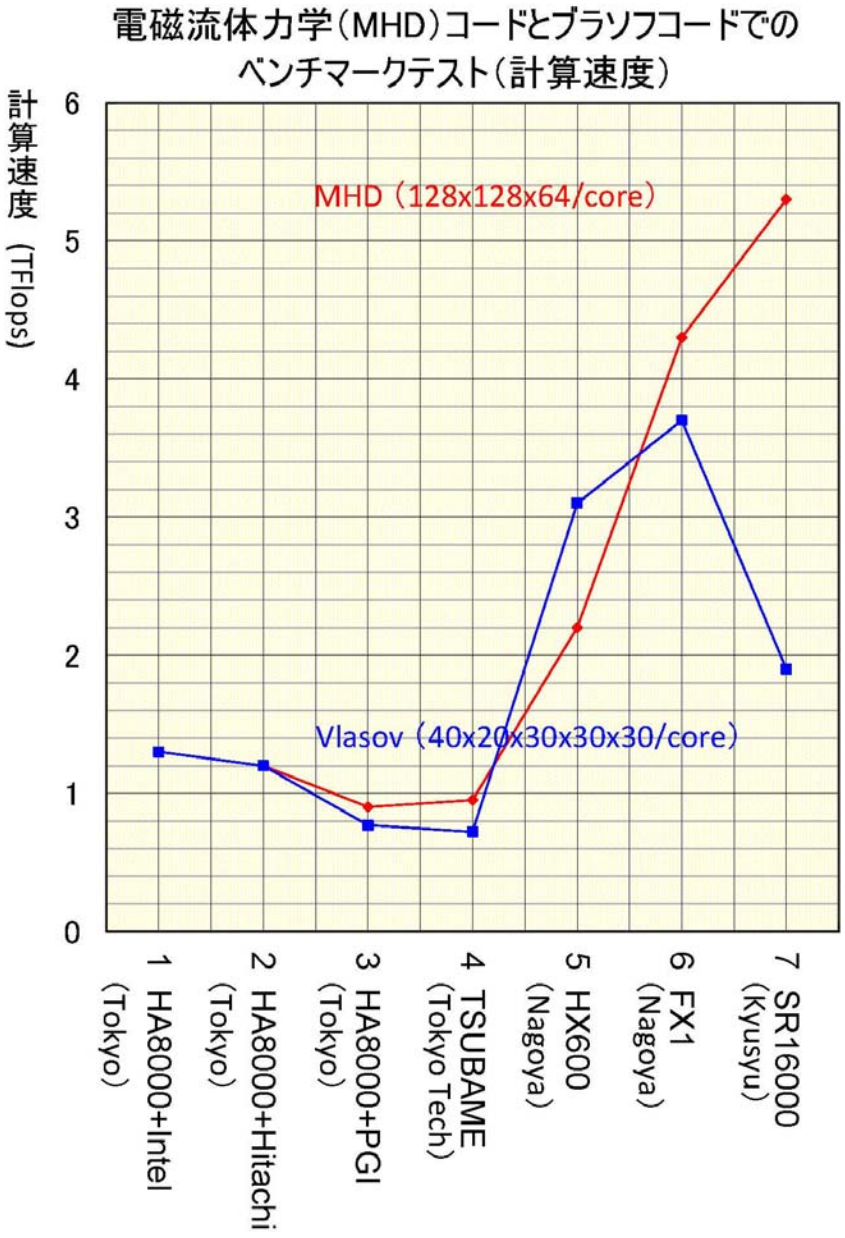


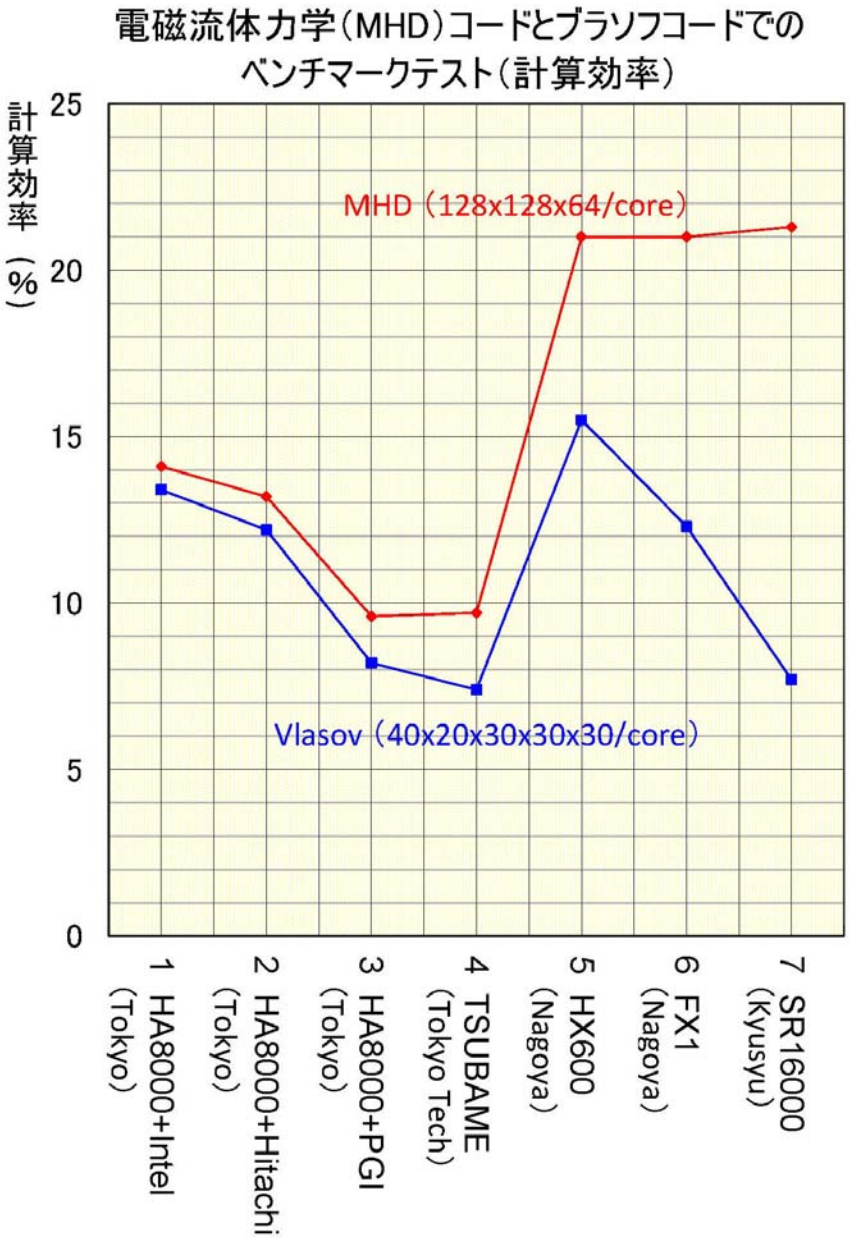
- Type A : f(i, j, k, m)
- Type B : f(m, i, j, k)

i, j, k が空間番号、m が変数番号
⇒同じ空間で変数に連続アクセス
HPC2500では有効だった

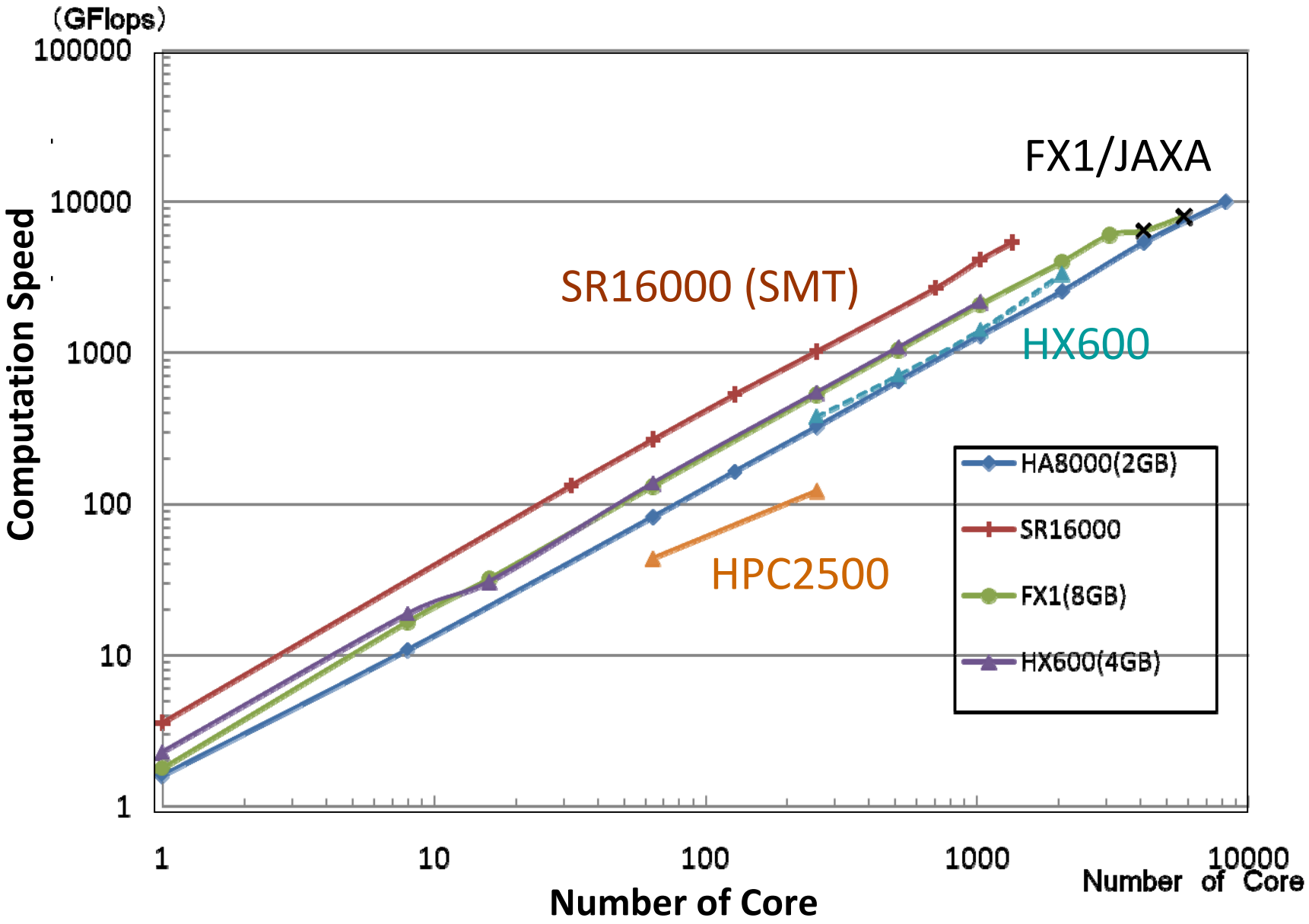
- FX1はHPC2500と同じ傾向
- HX600はSSEが有効
- 実効性能はどちらも約21%

MHDコードとVlasovコードでの性能評価





3D MHDコードによる計算速度の比較



名大スパコンのまとめ

・名古屋大学情報基盤センターの新スパコンシステムは実用の3次元MHDプログラムを用いて期待したとおりの性能がでている。

・全国大学共同利用情報基盤センターで唯一のプリペタコンであるFX1はFlat MPIの2048 coreで最高速4.3 Tflops（絶対効率21%）を実現した。
1024 coreでの最高速は2.2 Tflops（絶対効率22%）。

```
mpifrt progmpi.f -o progmpi -Kprefetch_model=FX1 -Z mpilist
```

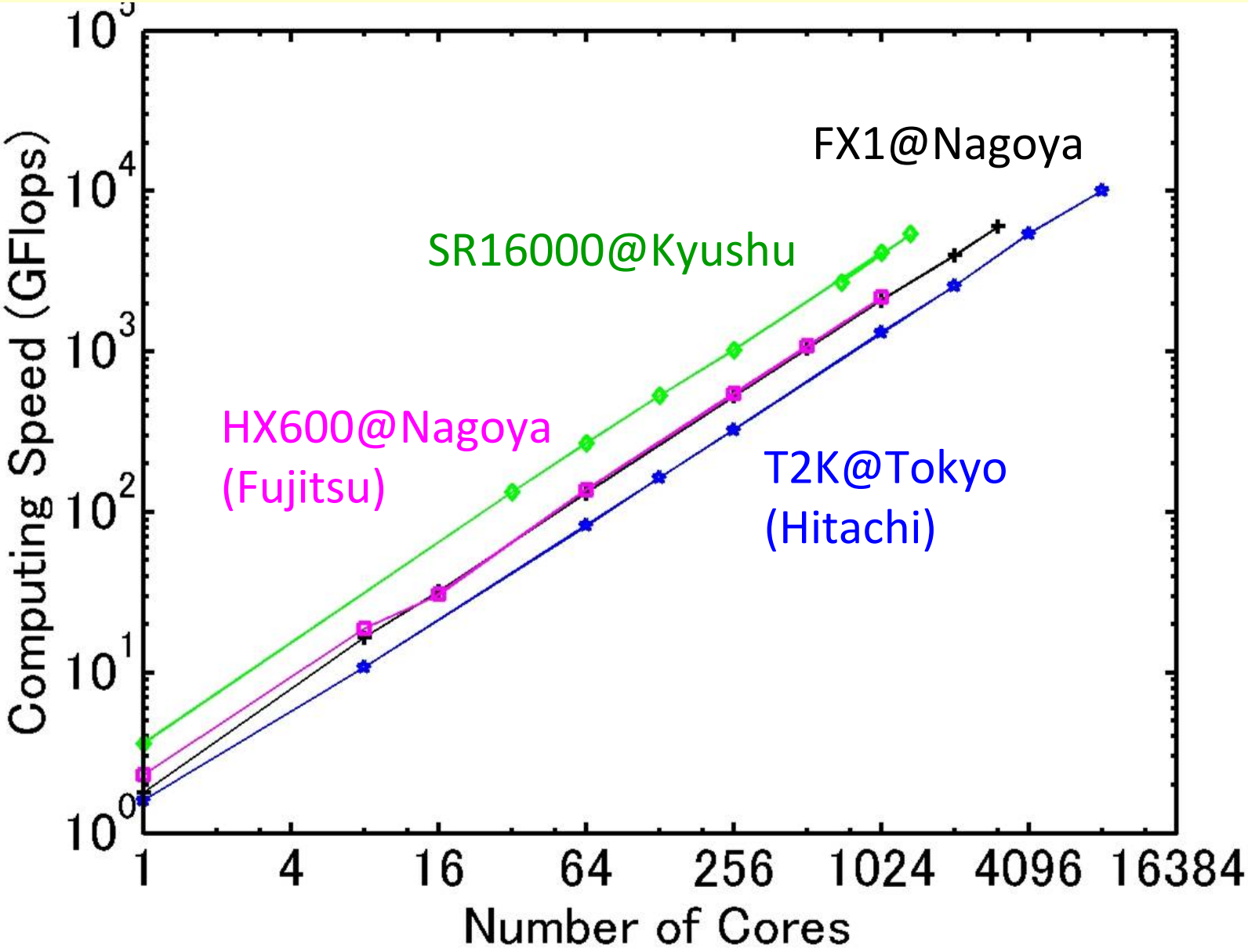
・FX1のハイブリッド並列（プロセス並列＋自動並列）は512 node 4 threadで4.1 Tflops（絶対効率20%）を得た。

```
mpifrt progmpi.f -o progmpi -Kimpact -Kprefetch_model=FX1 -Z mpilist
```

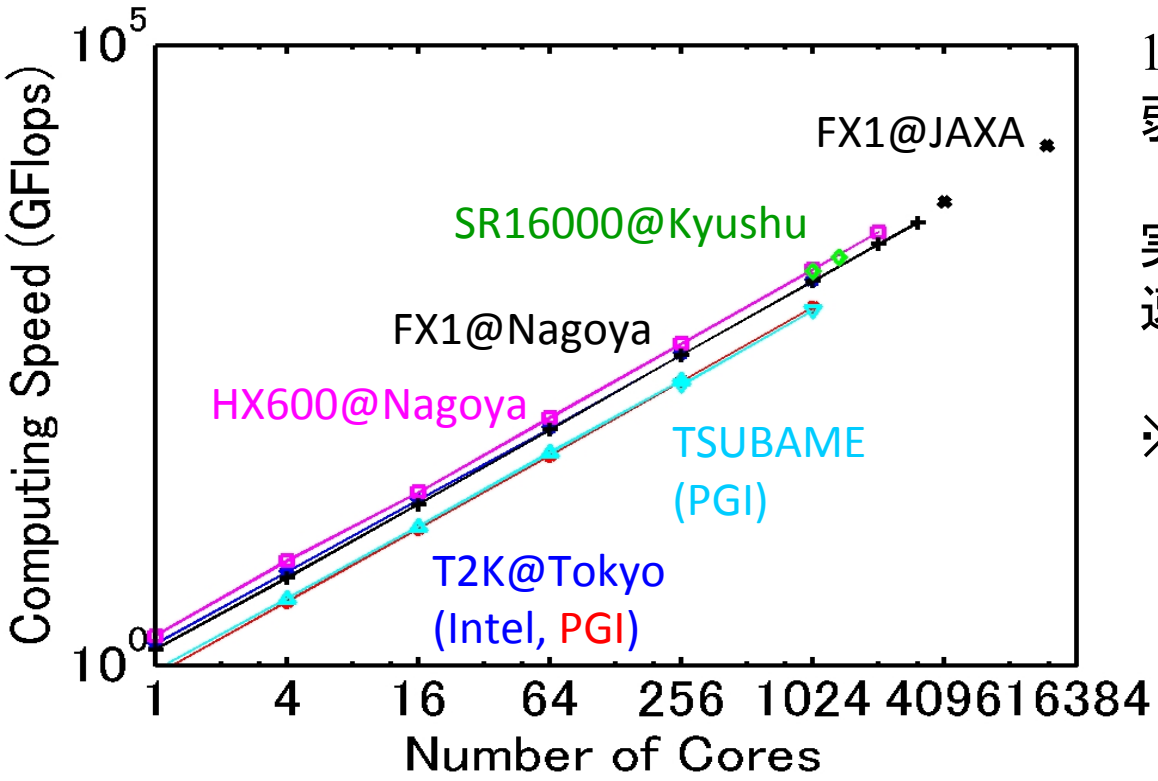
・HX600のFlat MPIの1024 coreでは最高速2.2 Tflops（絶対効率21%）を得た。

５．種々のスパコンの計算速度の比較と考察

MHDコードのベンチマークテスト



ブラソフコード性能評価

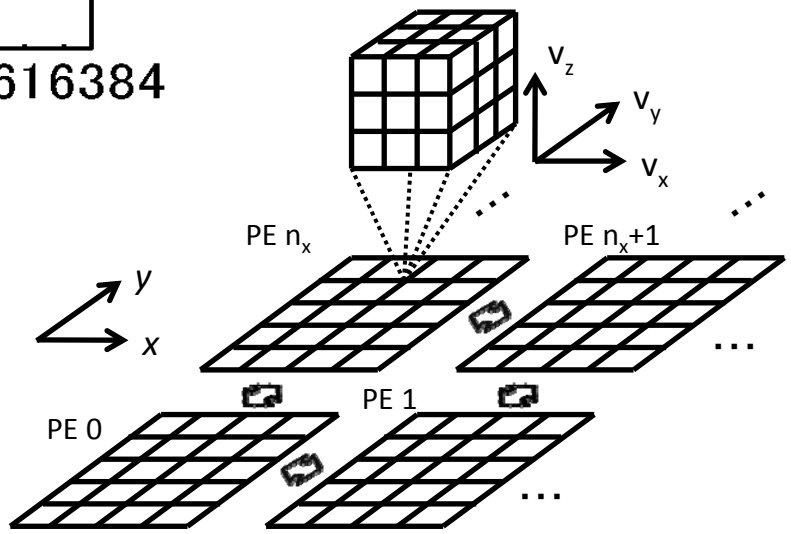


1coreあたり1Gに設定した
弱いスケーリング

実空間(x,y,z)をMPI並列
速度空間をOpenMP並列

※4096と12032はJAXAで計測

- HX600で実効性能約15%
- FX1では実効性能約12%
  - T2K@東大とほぼ同速



◆ 種々のスーパーコンピュータの計算速度の比較

並列ベクトル計算機から超並列スカラ計算機まで
- ベクトル機：Fujitsu：VPP5000、NEC：SX-6、SX-8R、SX-9、ES
- スカラ機（x86除く）：Fujitsu：HPC2500、FX1、PRIMQUEST、Hitachi：SR16000
- X86機：Hitachi HA8000、Fujitsu HX600、Sun：TSUBAME、Fujitsu PRIMERGY

	Core数/CPU数	実効性能(GFlops)	理論性能(GFlops)	実行効率(%)	領域分割	CPU種類
SX-9	64/64	2058	6553	31	2次元	ベクトル
HA8000	8192/1024	10038	75366	13	3次元A	Opteron
HX600	1024/256	2166	10240	21	3次元A	Opteron
FX1	1024/256	2081	10240	21	3次元B	SPARC64VII
SR16000	1344/672	5375	25267	21	3次元B	POWER6

	SX-9	HA8000	HX600	FX1	SR16000
@core	1.0000	0.0400	0.0658	0.0632	0.1244
@CPU	1.0000	0.1595	0.2631	0.2528	0.2487

新しい技術について

SMT（Simultaneous Multithreading）は有効か？

Core内で複数の実行スレッドを同時に実行するプロセッサの機能

Node – Multi-Core – Multi-Thread

日立SR16000ではSMT効果が顕著に現れる

富士通FX1ではSMT効果は微少

SMTやコア内複数演算機がいつも機能するかは疑問

９９．９９%を超える並列化率と２０%程度以上の絶対効率を得るためのヒント

１．プログラムのメイン部分で全てのdo loopを並列化
（１個でも残したらダメ）

２．全てのdo loopで並列化OKの表示が出ても安心してはいけない（if文や条件文は低効率の並列化をやっただけかも知れない）ほとんどの場合数倍程度

３．if文や条件文（**dependency**）を並列化するためには、**Mask, List vector, Gather and scatter**の中から最も適切なものを最も適切な方法で使う必要がある

４．実測する場合には粒度（１個のCPUのメモリ占有率）を揃える必要がある

５．最善のテストは**最大数のCPUを用いて、粒度を意識して絶対効率を実測**すること

今後解明すべきこと

1. Core数を1-10万個と更に増やした場合もscalabilityは保たれるか？（どうしたら予測できるか？）
2. 通信は本当にボトルネックになるのか？この限界を明確に示すにはどうすればよいのか？
3. Cacheのヒット率を更に上げるにはどうするのか？
4. Flat MPIとHybrid（プロセス並列＋自動並列）はCore数が増えた時、どちらが高効率なのか？
5. スパコンの種類によって高効率のプログラムは異なるのか（FX1型とT2K型）？

６．地球磁気圏の３次元MHDとVlasovシミュレーションの将来

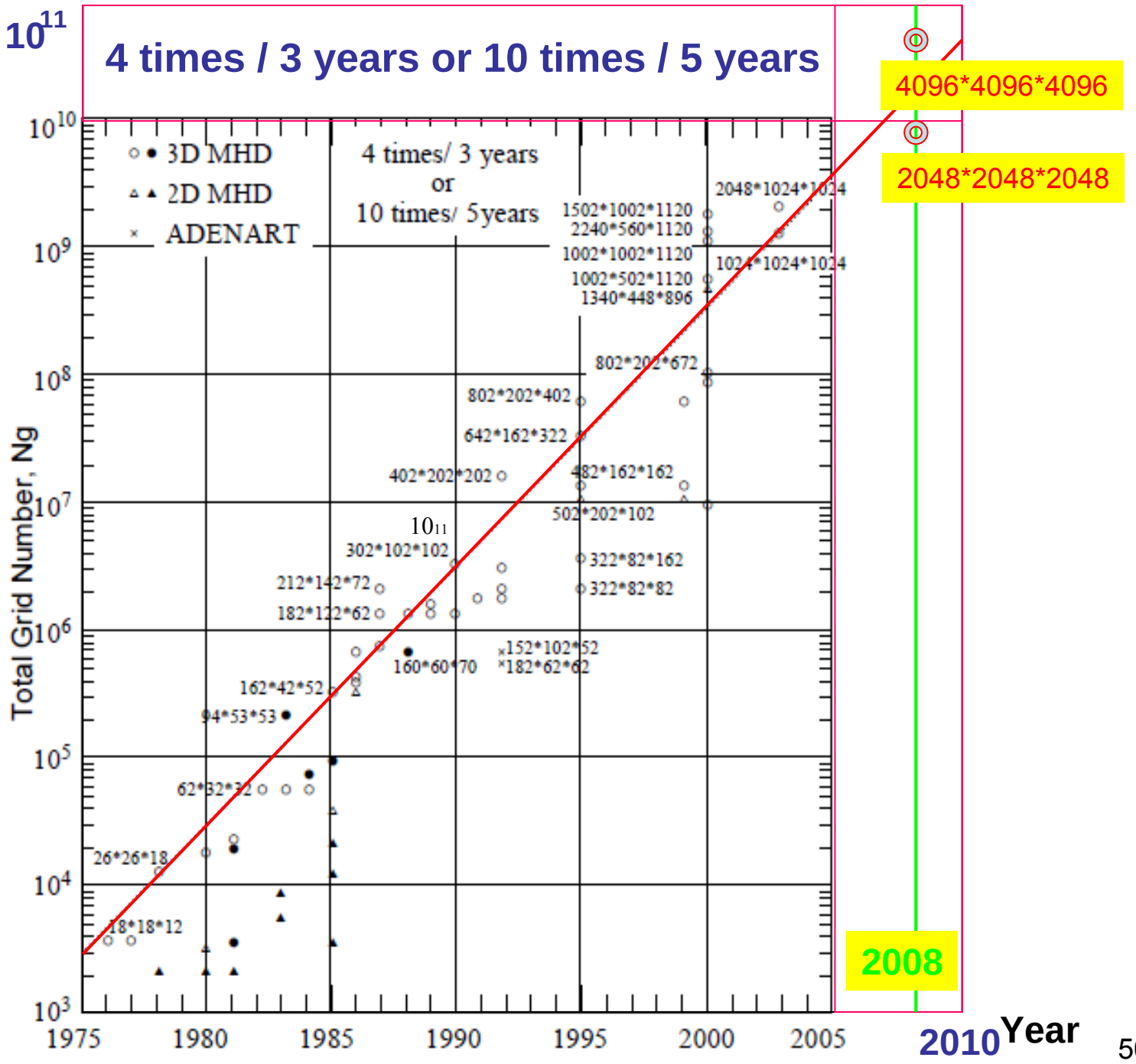
・どのシステムにおいてもほぼ線形なスケーラビリティ
・どのシステムにおいても、実効効率は10 - 20%以上

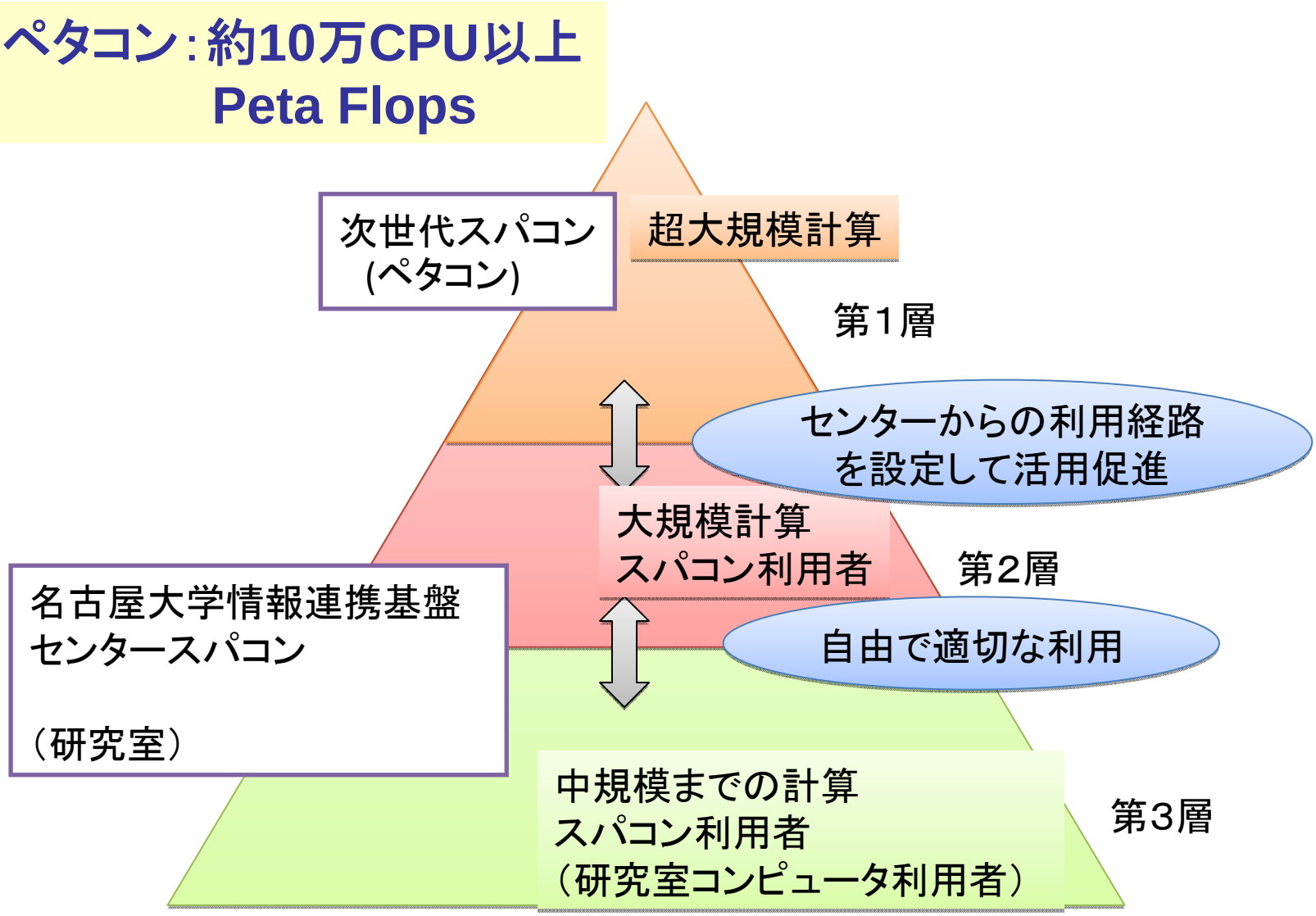
・今後は、128 – 1024 コアの並列計算により、スケール間結合の研究を推進

☆高解像度磁気圏グローバルMHDシミュレーション
☆中間スケール規模の第一原理ブラソフシミュレーション

2D and 3D MHD Simulation

3年で4倍、5年で10倍程度の割合で計算規模が増大





名古屋大学情報連携基盤センターのスパコン利用者からみた計算機資源の階層的配置

数値計算・シミュレーションの計算規模

第1層　超大規模計算：情報連携基盤センターのスパコンを越えての利用
（cpu数1000程度を超えて）、ペタコン利用が必要

第2層　大規模計算：情報連携基盤センターのスパコンを大規模に利用
（cpu数1000程度まで）

第3層　中規模までの計算：各グループ所有のクラスター計算機でも計算可能
（cpu数100程度まで）

ペタコン利用の研究計画の具体例（名古屋大学）

1：　一様非等方MHD乱流の減衰法則
2：　平行平板間乱流の超大規模直接数値計算
3：　タンパク質のシミュレーション
4：　ジオスペースにおける領域間結合
5：　無衝突衝撃波による粒子加速
6：　ブラソフシミュレーションで見る惑星磁気圏の3次元運動論的描像
7：　分子動力学法による液膜破壊現象の直接計算
8：　大規模非線形固有値問題の解法とその応用
9：　生体分子の立体構造予測シミュレーション
10：　強相関多体電子系シミュレーション
11：　雲のバーチャルリアリティ
12：　自己組織化マップを用いた気候変動の評価

4－6：
太陽地球環境研究所の研究計画

大規模高精度計算

1．研究課題：ジオスペースにおける領域間結合（磁気圏の境界層乱流と構造形成を解明）

2．研究の概要：

・太陽地球系の境界層プラズマ不安定の成長による構造形成とエネルギー・物質輸送

・プラズマ乱流を介してのマクロとメソスケール結合の非線形物理の解明

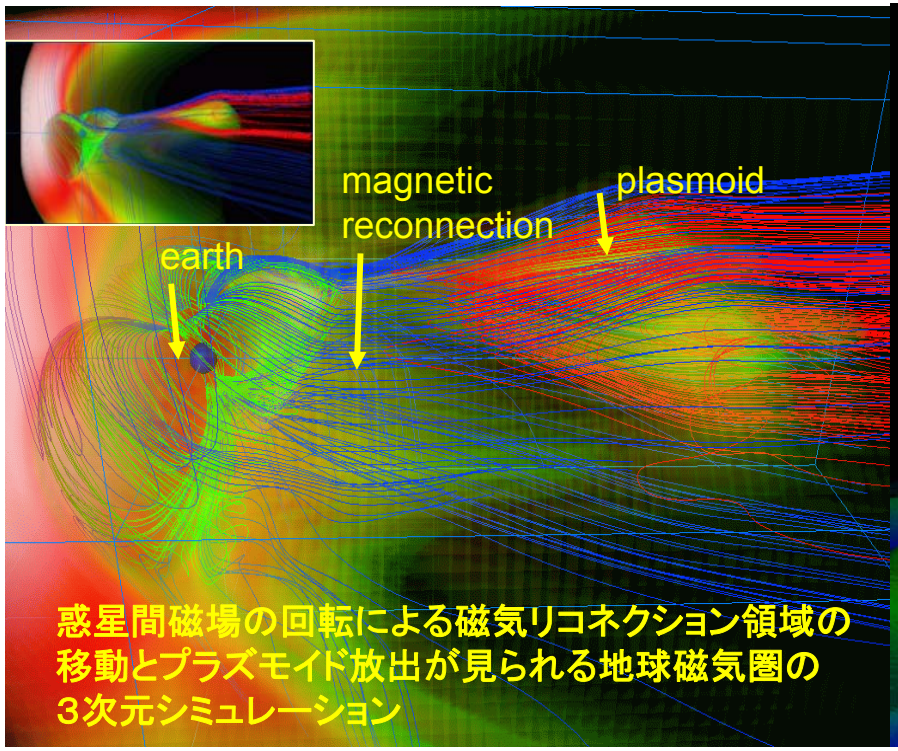
3．計算の概要：MHD、拡張MHD 及び流体モデルによるグローバルシミュレーション

4．現在までの到達度：３次元MHD コードで20%以上の高計算効率（1024 cpu）

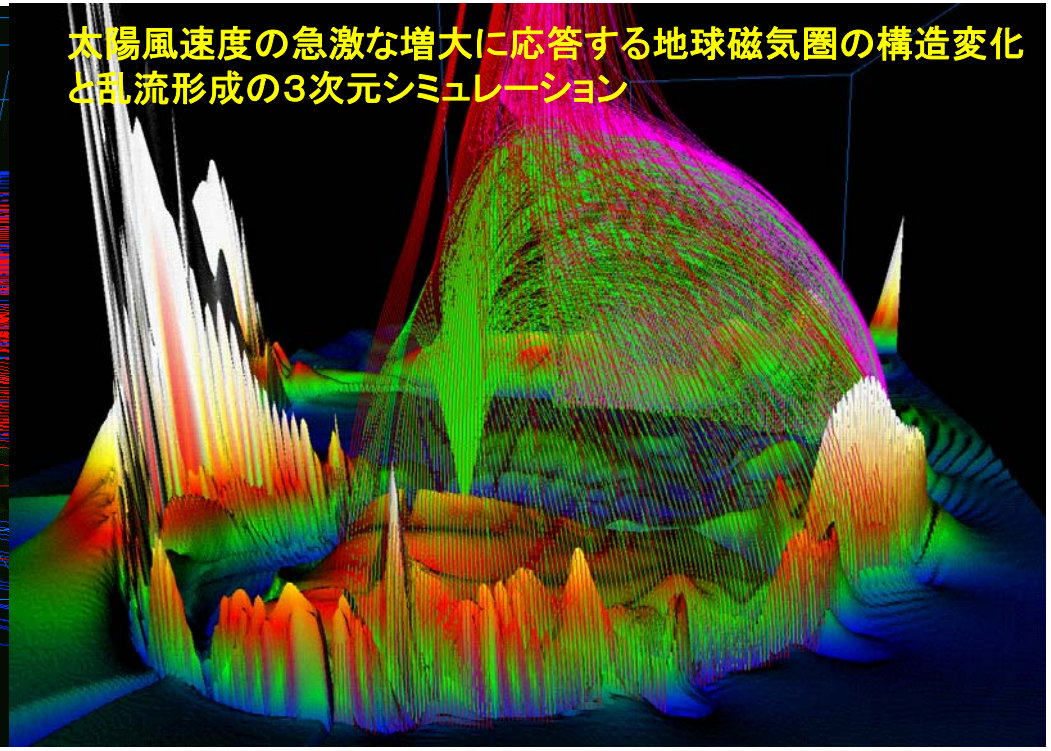
5．ペタコンで狙うものとその計算規模

・4096 の3乗の格子点を用いて現在のHPC2500 で10,000 時間のオーダーの計算量

・多流体、Hall MHD 、及びプラズマと中性流体の結合への拡張の必要性



惑星間磁場の回転による磁気リコネクション領域の移動とプラズモイド放出が見られる地球磁気圏の3次元シミュレーション



太陽風速度の急激な増大に応答する地球磁気圏の構造変化と乱流形成の3次元シミュレーション

7. まとめ

スーパースカラー並列機、スーパーコンピュータで
最高速の並列計算プログラムを作成するために

1. プログラムの構造を確立することが
   最も重要（Dependencyのない構造）
2. キャッシュのヒット率を最大化
3. 通信量の最小化と一括転送

これらを実現して、種々のスパコンにおいて、
実効効率は10 - 20%以上を達成

4. プログラムサイズの最小化は今後の目標
   （1次元化で領域分割は使用不可、実現可能か？）

FX1とHX600計算速度

コンピュータ	プログラム	グリッド数	総コア数	プロセス並列数	自動並列数	CPU時間 (sec)	計算速度 (Gflops)	効率 (%)
FX1	1Da	1024**3	256	64	4	152.0	9	0.4
FX1	1Da	1024**3	128	128	1	232.2	6	0.5
FX1	1Da	1124*1024**2	128	128	1	28.4	52	4.1
FX1	1Da	602*402*402	256	64	4	0.417	294	11.5
FX1	1Da	902*402*402	256	64	4	0.704	261	10.2
FX1	1Da	1124*1024**2	128	128	1	8.23	180	14.1
FX1	3Da	1024**3	128	128	1	7.32	185	14.4
HX600	1Da	512**3	64	64	1	4.06	42	6.5
HX600	1Da	1024**3	256	64	4	9.81	138	5.4

1Da：1次元領域分割＋f(i,j,k,m)　m：MHD方程式の8個成分

2Da：2次元領域分割＋f(i,j,k,m)

3Da：3次元領域分割＋f(i,j,k,m) 3Db：3次元領域分割＋f(m,i,j,k)

1次元領域分割も配列数を適切に選べばかなりの計算速度が得られる。

FX1の計算速度：1次元コード：f(i1)

・1次元配列f(i1)MHDコードはメモリを最小化したプログラム
・領域分割法を利用できないので新しいユーザー（プログラム）分割法の開発が必要

コンピュータ	グリッド数	総コア数	プロセス並列数	自動並列数	CPU時間 (sec)	計算速度 (Gflops)	効率 (%)
FX1	62*32*32	4	1	4	0.0227	3.5	8.8
FX1	602*302*302	4	1	4	14.4	4.8	12.0
FX1	802*402*402	4	1	4	32.6	5.0	12.5
FX1	1002*502*502	4	1	4	65.7	4.8	12.1
FX1	50*50*26	4	1	4	0.0238	3.4	8.6
FX1	402*402*202	4	1	4	8.78	4.7	11.7
FX1	602*602*302	4	1	4	28.9	4.8	11.9
FX1	802*802*402	4	1	4	66.7	4.9	12.2

f(i1): i1=i+n1*(j-1)+n2*(k-1)+n3*(m-1), n1=nx2, n2=n1*ny2, n3=n2*nz2
f(i1)=f(i,j,k,m), f(n4)=f(nx2,ny2,nz2,nb), n4=n3*nb