東京大学情報基盤センター
INFORMATION TECHNOLOGY CENTER, THE UNIVERSITY OF TOKYO

東京大学
THE UNIVERSITY OF TOKYO

Wisteria/BDEC-01 と h3-Open-BDEC

Wisteria
BDEC-01

Hierarchical, Hybrid, Heterogeneous
h3-Open-BDEC
Big Data & Extreme Computing

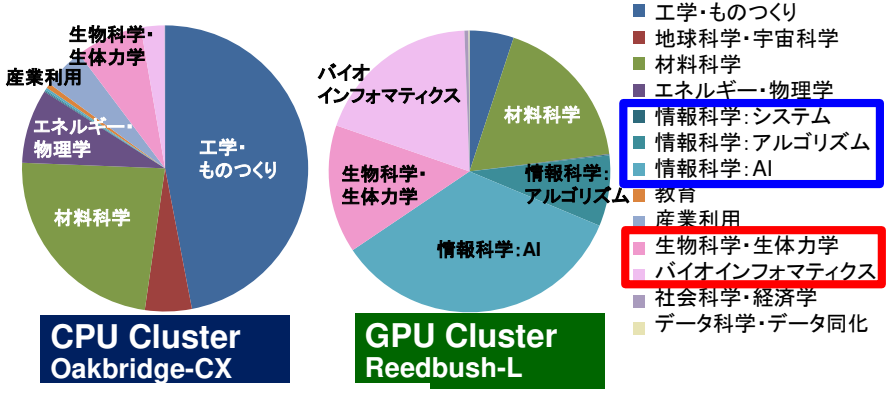
中島 研吾
東京大学情報基盤センター

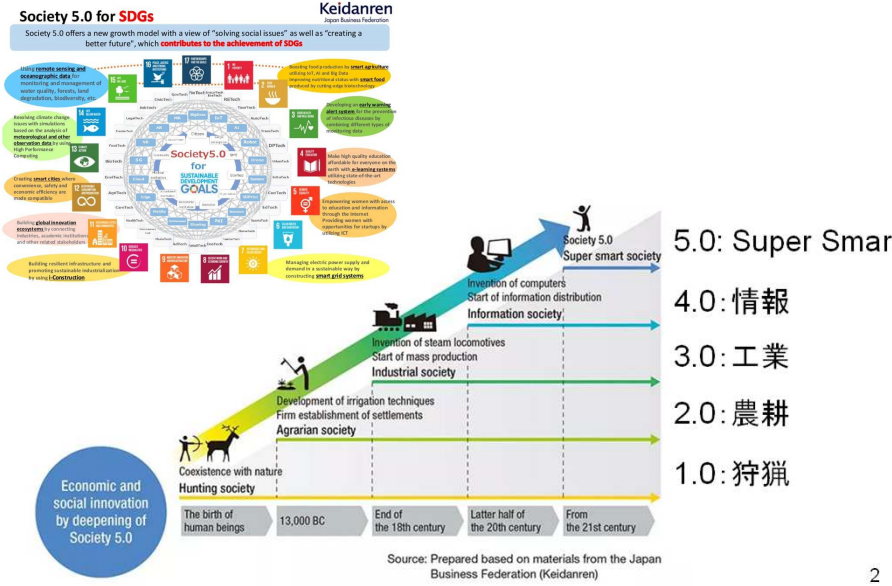
PCCC-OSSSワークショップ
2023年3月30日

PC Cluster Consortium

スーパーコンピューティングの今後

- ワークロードの多様化
  - 計算科学，計算工学：Simulations
  - 大規模データ解析
  - AI，機械学習
- **（シミュレーション（計算）＋データ＋学習）融合 ⇒ Society 5.0実現に有効：Digital Twin**
  - **フィジカル空間とサイバー空間の融合**
    - **S:シミュレーション（計算）（Simulation）**
    - **D:データ（Data）**
    - **L：学習（Learning）**
  - **Simulation + Data + Learning = S+D+L**





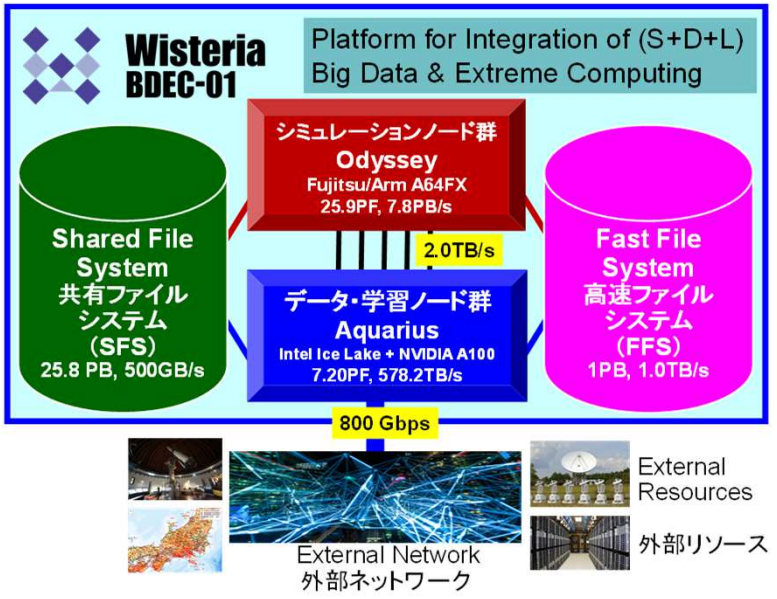
（シミュレーション（計算）＋データ＋学習）融合（S+D+L）

- 東大情報基盤センターでは，2015年頃から「（S+D+L）融合」の重要性に注目し，それを実現するためのハードウェア，ソフトウェア，アプリケーション，アルゴリズムに関する研究開発を開始
  - BDEC計画（Big Data & Extreme Computing）
  - 「データ＋学習」による，より高度な「シミュレーション」
    - AI for HPC, AI for Science
  - 地球科学関連では自然な発想（すでに実施されている）
- 2021年5月に運用を開始した「Wisteria/BDEC-01」は「BDEC計画」の1号機
  - Reedbush，Oakbridge-CXは「BDEC」のプロトタイプと位置づけられる
  - 「計算・データ・学習（S+D+L）」融合を実現する，世界でも初めてのプラットフォーム

Wisteria/BDEC-01

BDEC:「計算・データ・学習（S+D+L）」融合のためのプラットフォーム（Big Data & Extreme Computing）

- 2021年5月14日運用開始
  - 東京大学柏Ⅱキャンパス
- 33.1 PF, 8.38 PB/sec. , **富士通製**
  - ~4.5 MVA（空調込み）, ~360m^2
- Hierarchical, Hybrid, Heterogeneous (h3)
- **2種類のノード群**
  - **シミュレーションノード群（S, SIM）：Odyssey**
    - **従来のスパコン**
    - **Fujitsu PRIMEHPC FX1000 (A64FX), 25.9 PF**
      - **7,680ノード（368,640 コア）, 20ラック, Tofu-D**
  - **データ・学習ノード群（D/L, DL）：Aquarius**
    - **データ解析，機械学習**
    - **Intel Xeon Ice Lake + NVIDIA A100, 7.2 PF**
      - **45ノード（Ice Lake：90基，A100：360基）, IB-HDR**
    - **一部は外部リソース（ストレージ，サーバー，センサーネットワーク他）に直接接続**
  - ファイルシステム：共有（大容量）＋高速

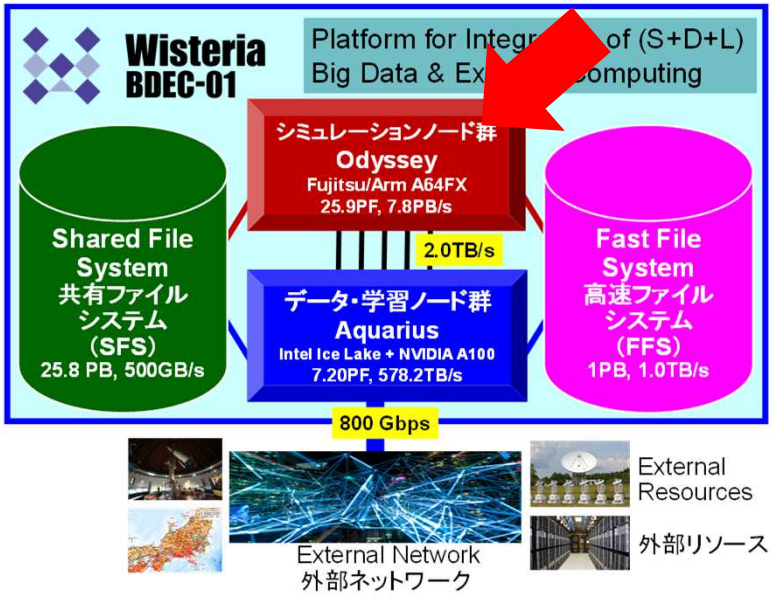


Wisteria
BDEC-01

Wisteria/BDEC-01

BDEC:「計算・データ・学習（S+D+L）」融合のためのプラットフォーム（Big Data & Extreme Computing）

- 2021年5月14日運用開始
  - 東京大学柏Ⅱキャンパス
- 33.1 PF, 8.38 PB/sec. , **富士通製**
  - ~4.5 MVA（空調込み）, ~360m^2
- Hierarchical, Hybrid, Heterogeneous (h3)
- **2種類のノード群**
  - **シミュレーションノード群（S, SIM）：Odyssey**
    - **従来のスパコン**
    - **Fujitsu PRIMEHPC FX1000 (A64FX), 25.9 PF**
      - **7,680ノード（368,640 コア）, 20ラック, Tofu-D**
  - **データ・学習ノード群（D/L, DL）：Aquarius**
    - **データ解析，機械学習**
    - **Intel Xeon Ice Lake + NVIDIA A100, 7.2 PF**
      - **45ノード（Ice Lake：90基，A100：360基）, IB-HDR**
    - **一部は外部リソース（ストレージ，サーバー，センサーネットワーク他）に直接接続**
  - ファイルシステム：共有（大容量）＋高速

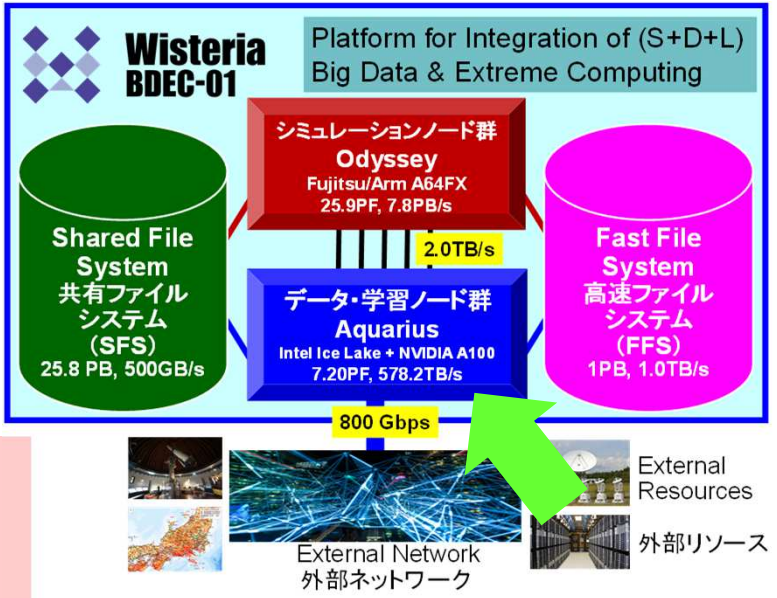


Wisteria
BDEC-01

Wisteria/BDEC-01

BDEC:「計算・データ・学習（S+D+L）」融合のためのプラットフォーム（Big Data & Extreme Computing）

- 2021年5月14日運用開始
  - 東京大学柏Ⅱキャンパス
- 33.1 PF, 8.38 PB/sec. , **富士通製**
  - ~4.5 MVA（空調込み）, ~360m^2
- Hierarchical, Hybrid, Heterogeneous (h3)
- **2種類のノード群**
  - **シミュレーションノード群（S, SIM）：Odyssey**
    - **従来のスパコン**
    - **Fujitsu PRIMEHPC FX1000 (A64FX), 25.9 PF**
      - **7,680ノード（368,640 コア）, 20ラック, Tofu-D**
  - **データ・学習ノード群（D/L, DL）：Aquarius**
    - **データ解析, 機械学習**
    - **Intel Xeon Ice Lake + NVIDIA A100, 7.2 PF**
      - **45ノード（Ice Lake：90基, A100：360基）, IB-HDR**
    - **一部は外部リソース（ストレージ, サーバー, センサーネットワーク他）に直接接続**
  - ファイルシステム：共有（大容量）＋高速



Wisteria
BDEC-01

Simulation Nodes
Odyssey
25.9 PF, 7.8 PB/s

Fast File System (FFS)
1.0 PB, 1.0 TB/s

Shared File System (SFS)
25.8 PB, 0.50 TB/s

Data/Learning Nodes
Aquarius
7.20 PF, 578.2 TB/s

Wisteria
BDEC-01

Wisteria/BDEC-01

計算科学コード

シミュレーション
ノード群，Odyssey

最適化されたモデル，
パラメータ

計算結果

機械学習，DDA

データ・学習ノード群
Aquarius

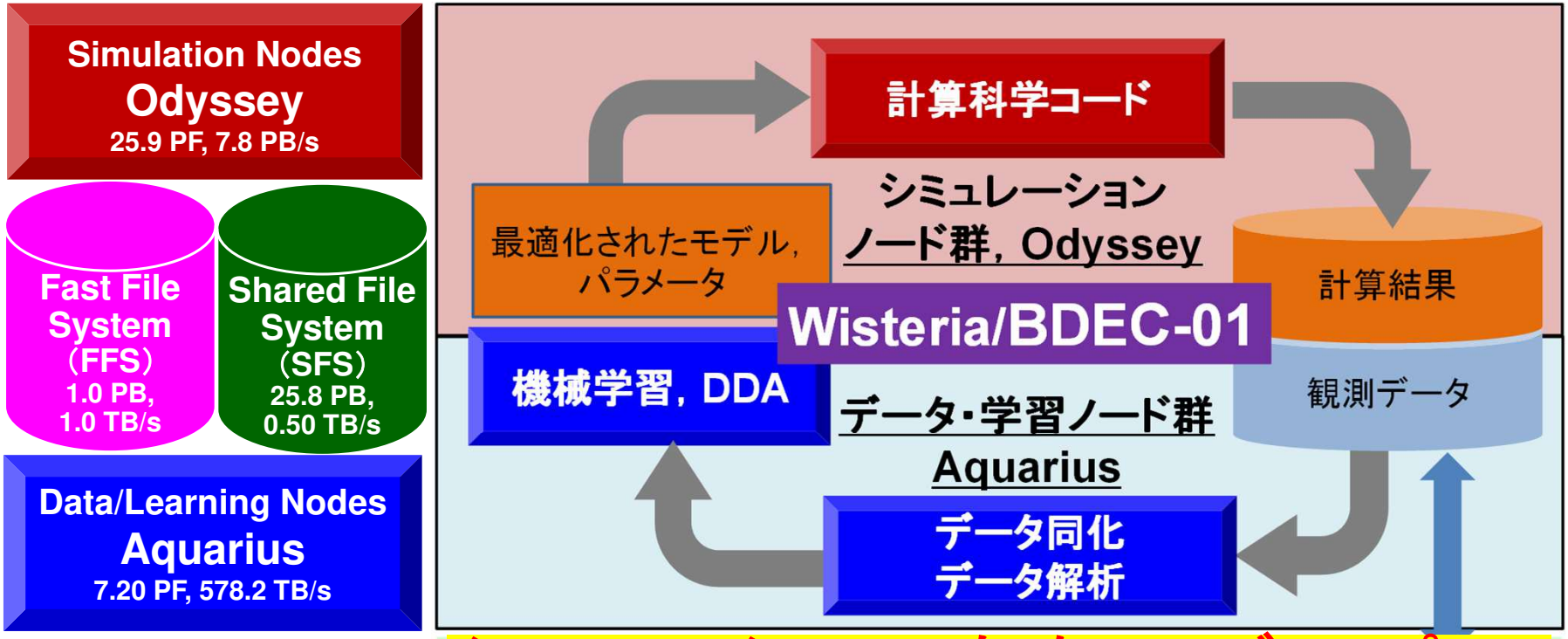
観測データ

データ同化
データ解析

サーバー
ストレージ
DB
センサー群
他

外部ネットワーク

外部
リソース



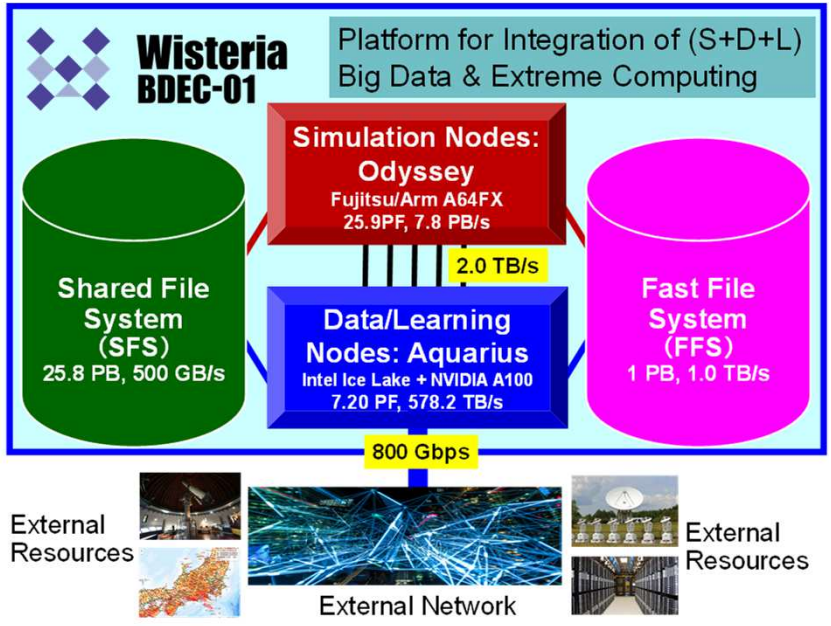
Wisteria
BDEC-01

シミュレーションのためのモデル・パラメータのデータ解析，AI/機械学習による最適化（S+D+L）

SC22における諸ランキング（2022年11月）

SC22 Dallas, TX | hpc accelerates.

	Odyssey	Aquarius
TOP 500	23	125
Green 500	45	28
HPCG	12	68
Graph 500 BFS	4	-
HPL-MxP (HPL-AI)	14	-



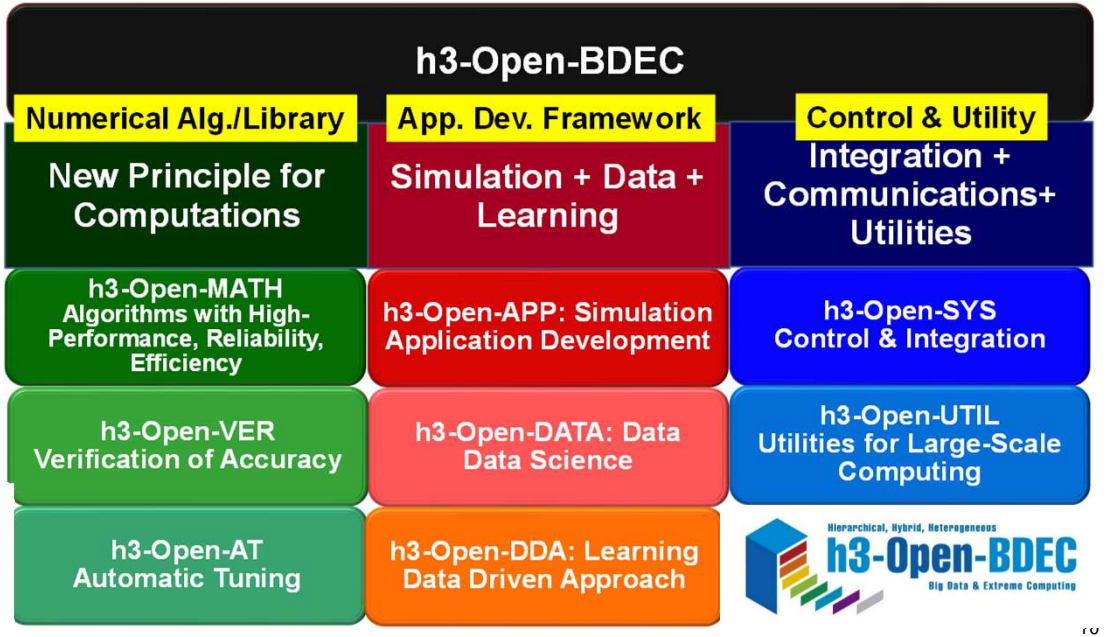
h3-Open-BDEC

「計算＋データ＋学習」融合を実現する革新的ソフトウェア基盤
科研費基盤研究（S）（2019年度～23年度，代表：中島研吾）
https://h3-open-bdec.cc.u-tokyo.ac.jp/

Hierarchical, Hybrid, Heterogeneous

Big Data & Extreme Computing

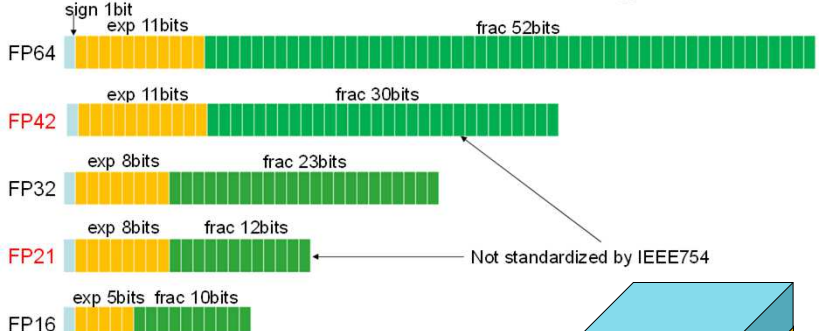
① 変動精度演算・精度保証・自動チューニングによる新計算原理に基づく革新的数値解法
② 階層型データ駆動アプローチ等に基づく革新的機械学習手法
③ ヘテロジニアス環境（e.g. Wisteria/BDEC-01）におけるソフトウェア，ユーティリティ群

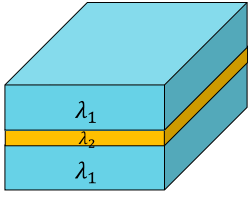


Hierarchical, Hybrid, Heterogeneous
h3-Open-BDEC
Big Data & Extreme Computing

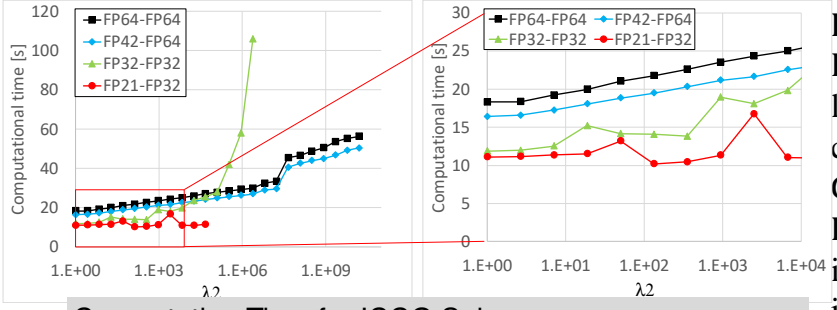
Adaptive Precision Computing with FP21/FP42

Masatoshi Kawai (kawai@cc.u-tokyo.ac.jp)





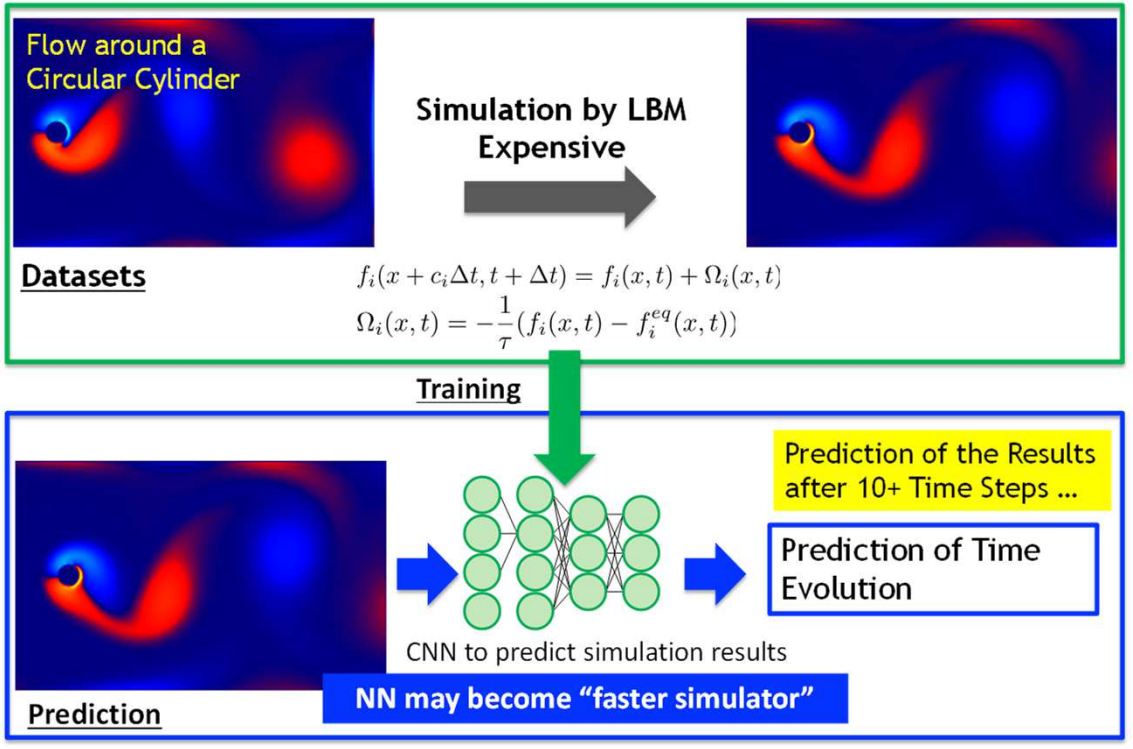
Heat Conduction with
Heterogeneous Material Property



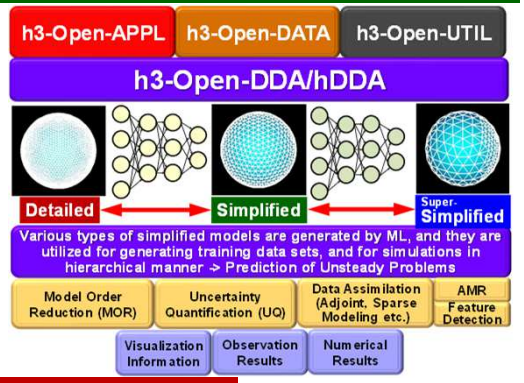
Computation Time for ICCG Solver
Various Types of Precisions on Intel Xeon Cascadelake

In recent years, the usefulness of low-precision floating-point representation has been studied in various fields such as machine learning. Low accuracy can be expected to have effects such as shortening calculation time and reducing power consumption. For example, in an application with a memory bandwidth bottleneck, the effect of reducing the calculation time by reducing the amount of memory transfer is significant. However, in fields such as iterative methods, it is common to use FP64 because the calculation accuracy strongly affects the convergence, and there are few application examples of low-precision arithmetic. This study investigates the applicability of low-precision representation to the Krylov subspace and stationary iterative methods. In this research, we focus on the FP32, FP16, and FP42, FP21, which are not standardized by IEEE754. Developed method has been evaluated for ICCG solver, which solves linear equations derived from 3D FVM code for steady-state head conduction with heterogeneous material property ($\lambda_1=10^0$, $\lambda_2=10^0$~10^9). Generally, computation with lower precision (e.g. FP32-FP32, FP21-FP32) becomes unstable, if condition number of the coefficient matrix is larger (λ_2 is larger), FP21-FP32 provides the best performance if λ_2 is up to 10^4. ("FP21-FP32" means "matrices are in FP21, and vectors are in FP32)

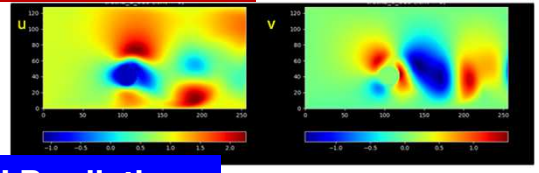
Acceleration of Transient CFD Simulations using ML/CNN Integration of (S+D+L), AI for HPC/AI for Science



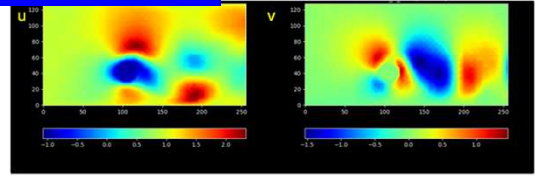
[c/o Takashi Shimokawabe (ITC/U.Tokyo)]



Simulations: LBM

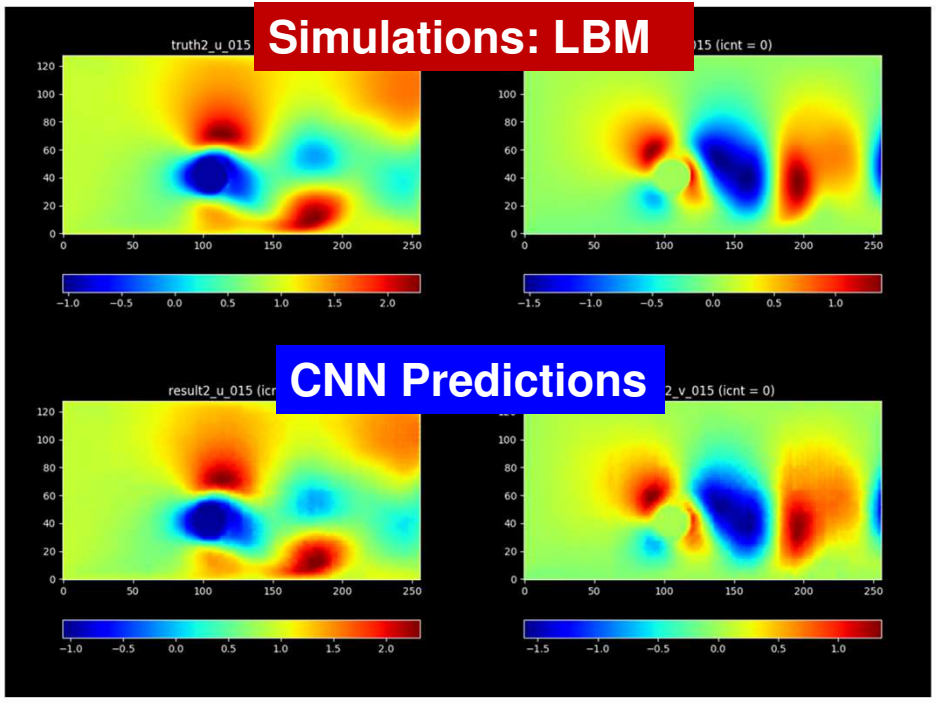


CNN Predictions



Prediction of CFD Simulation by ML/CNN

Takashi Shimokawabe (shimokawabe@cc.u-tokyo.ac.jp)

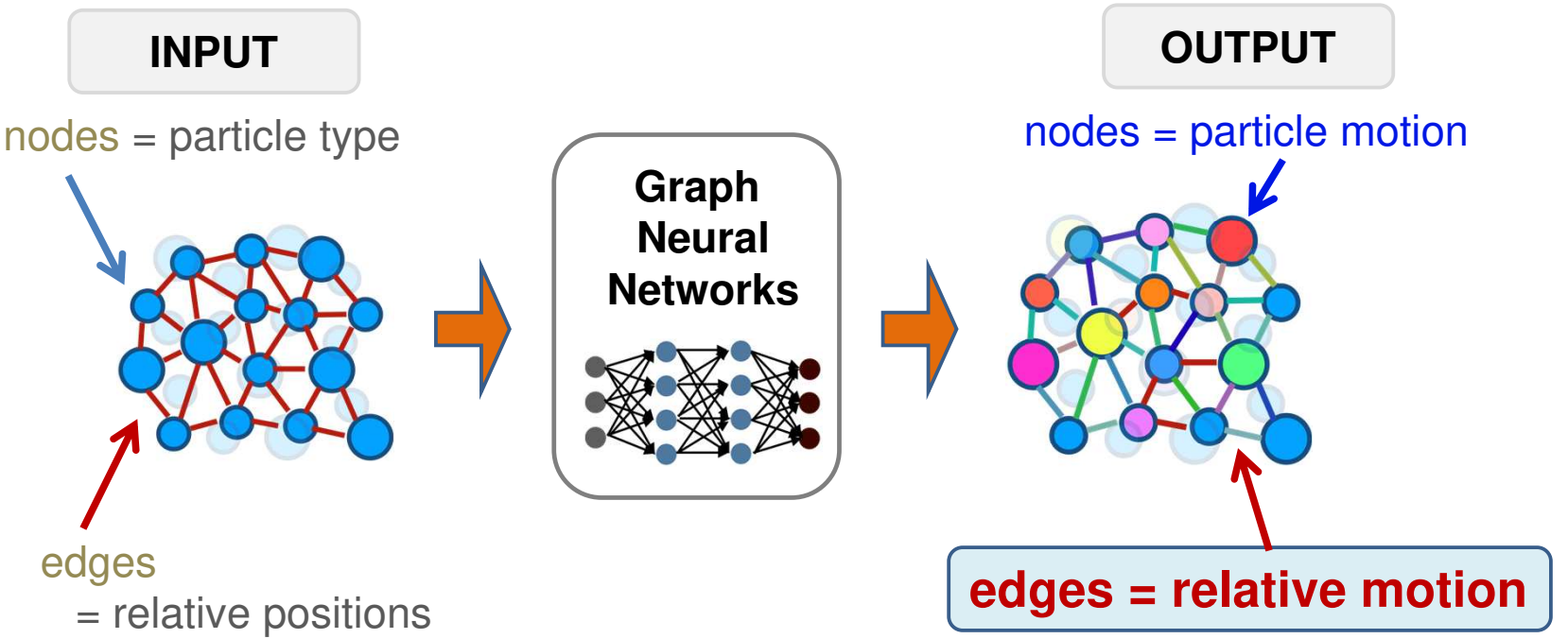


Comparison of the flow velocity results obtained by the conventional simulation (upper) and the prediction of these results by deep learning (lower)

Computational fluid dynamics (CFD) is widely used in science and engineering. However, since CFD simulations requires a large number of grid points and particles for these calculations, these kinds of simulations demand a large amount of computational resources such as supercomputers. Recently, deep learning has attracted attention as a surrogate method for obtaining calculation results by CFD simulation approximately at high speed. We are working on a project to develop a parallelization method to make it possible to apply the surrogate method based on the deep learning to large scale geometry. Unlike the model parallel computing, the method we are currently developing predicts large-scale steady flow simulation results by dividing the input geometry into multiple parts and applying a single small neural network to each part in parallel. This method is developed based on considering the characteristics of CFD simulation and the consistency of the boundary condition of each divided subdomain. By using the physical values on the adjacent subdomains as boundary conditions, applying deep learning to each subdomain can predict simulation results consistently in the entire computational domain. It is possible to predict the simulation results in about 36.9 seconds by the developed method, compared to about 286.4 seconds by the conventional numerical method. In addition to this, we are also attempting to develop a method for fast prediction of time evolution calculations using deep learning.

Machine learning slow molecular dynamics

Our proposal — **BOnd Targeting Network (BOTAN)**



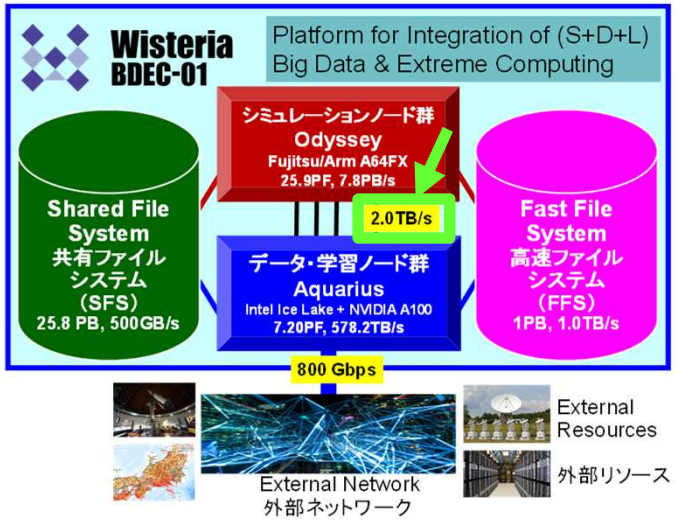
H. Shiba, M. Hanai, T. Suzumura, and T. Shimokawabe, arXiv:2206.14024 (2022)

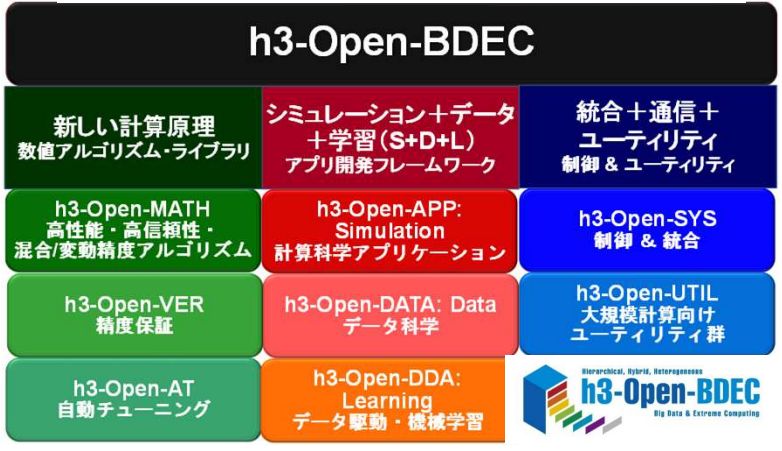
AI for HPC, AI for Science の実現へ向けて

Wisteria BDEC-01

Hierarchical, Hybrid, Heterogeneous
h3-Open-BDEC
Big Data & Extreme Computing

- Odyssey-Aquarius連携
  - MPIによる通信は不可
    - O-Aを跨いでMPIプログラムは動かない
  - Odyssey-Aquarius間はInfiniband-EDR（2TB/sec）で結合されている
- ソフトウェア開発
  - 高機能カプラー：h3-Open-UTIL/MP
  - O-A間通信：h3-Open-SYS/WaitIO
    - IB-EDR経由（WaitIO-Socket）
    - 高速ファイルシステム（FFS）経由連携（WaitIO-File）





h3-Open-BDEC

「計算＋データ＋学習」融合を実現する革新的ソフトウェア基盤
科研費基盤研究（S）（2019年度～23年度，代表：中島研吾）
https://h3-open-bdec.cc.u-tokyo.ac.jp/

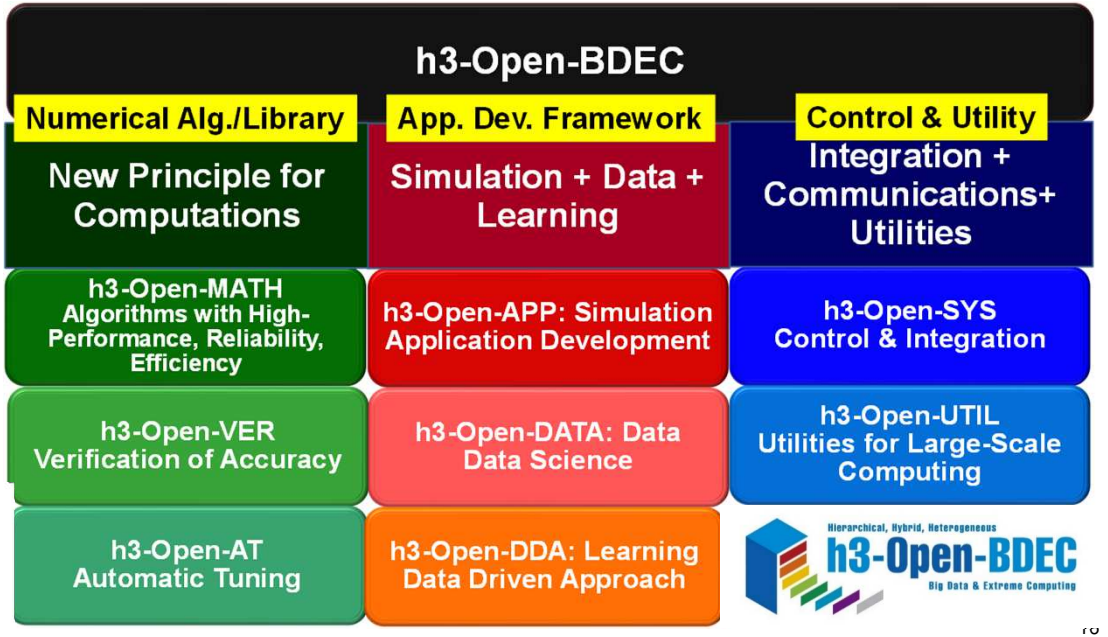
Hierarchical, Hybrid, Heterogeneous

Big Data & Extreme Computing

① 変動精度演算・精度保証・自動チューニングによる新計算原理に基づく革新的数値解法

② 階層型データ駆動アプローチ等に基づく革新的機械学習手法

③ ヘテロジニアス環境（e.g. Wisteria/BDEC-01）におけるソフトウェア，ユーティリティ群

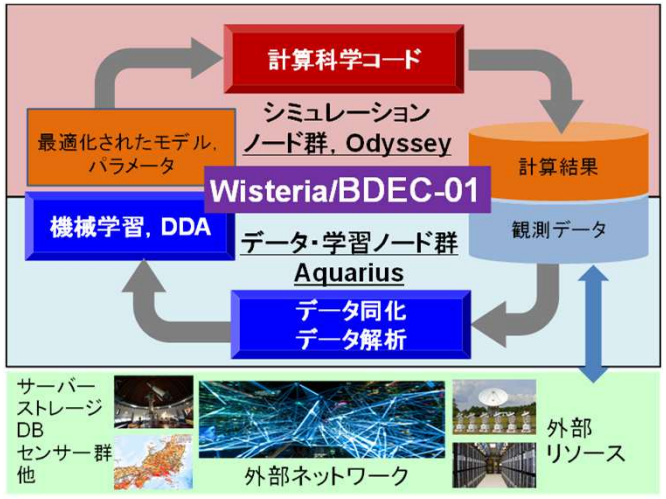


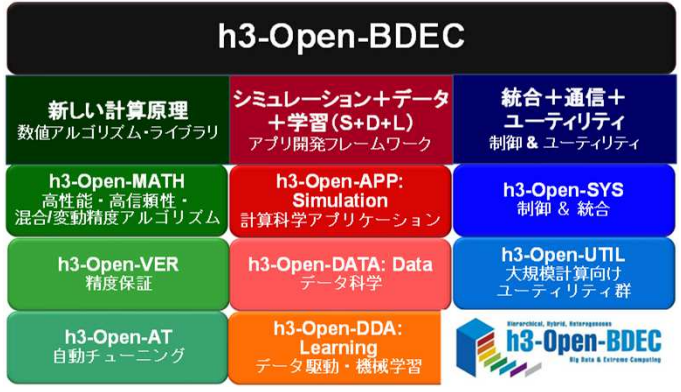
Hierarchical, Hybrid, Heterogeneous
h3-Open-BDEC
Big Data & Extreme Computing

h3-Open-SYS/WaitIO

データ受け渡しライブラリ〔松葉，2020〕〔住元他，HPC-181，2021〕

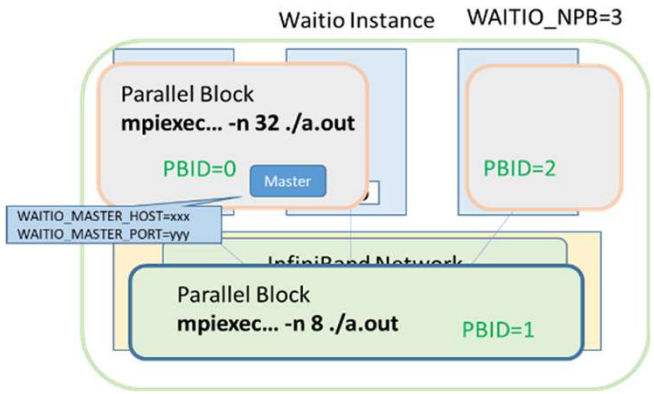
- ヘテロジニアス環境下での異なるコンポーネント間ファイル経由連携ライブラリとして考案
- 機能
  - ✓ Odyssey～Aquarius間連携
    - IB-EDR経由通信（WaitIO-Socket）
    - ファイル経由（WaitIO-File）
  - ✓ 外部からのデータ取得（観測データ等）
  - ✓ 読み込み・書き出しの同期
- API：C/C++，Fortranから呼び出し可能
  - ✓ MPIライクなインタフェースを提供





API of h3-Open-SYS/WaitIO-Socket PB (Parallel Block): Each Application

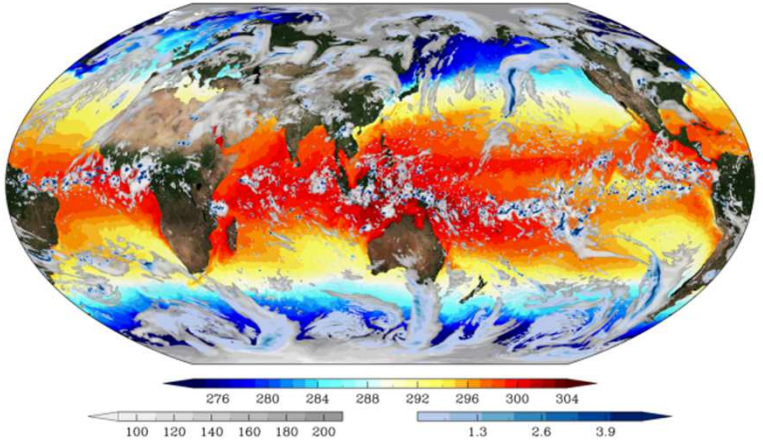
WaitIO API	Description
waitio_isend	Non-Blocking Send
waitio_irecv	Non-Blocking Receive
waitio_wait	Termination of waitio_isend/irecv
waitio_init	Initialization of WaitIO
waitio_get_nprocs	Process # for each PB (Parallel Block)
waitio_create_group waitio_create_group_wranks	Creating communication groups among PB's
waitio_group_rank	Rank ID in the Group
waitio_group_size	Size of Each Group
waitio_pb_size	Size of the Entire PB
waitio_pb_rank	Rank ID of the Entire PB

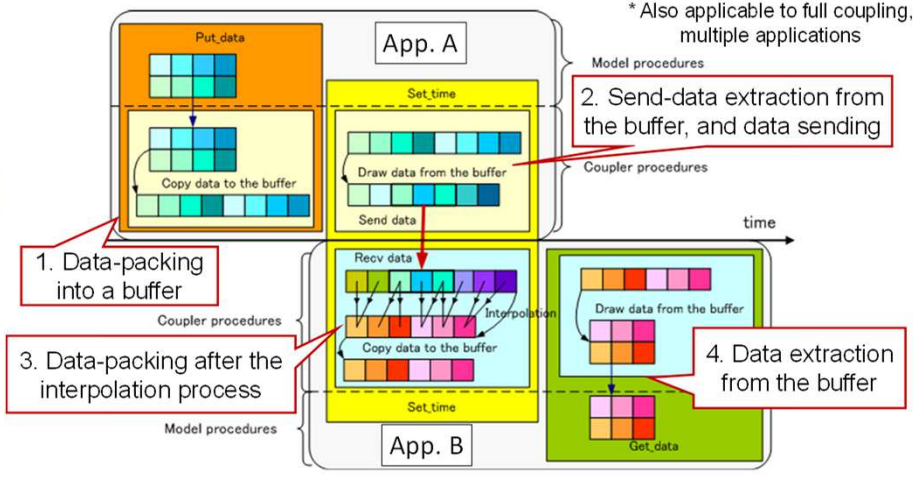


[Sumimoto et al. 2021]

連成シミュレーションのためのカプラー〔荒川，八代〕

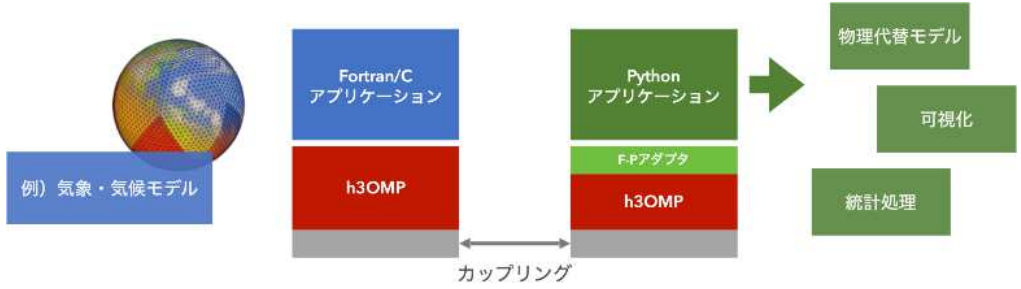
- 従来のカプラー（Coupler）：ppOpen-MATH/MP
  - 複数（通常2つ：大気（NICAM）＋海洋（COCO））のアプリケーションの弱連成（Weak Coupling）をサポート
  - 各アプリケーションは1種類の計算をやる





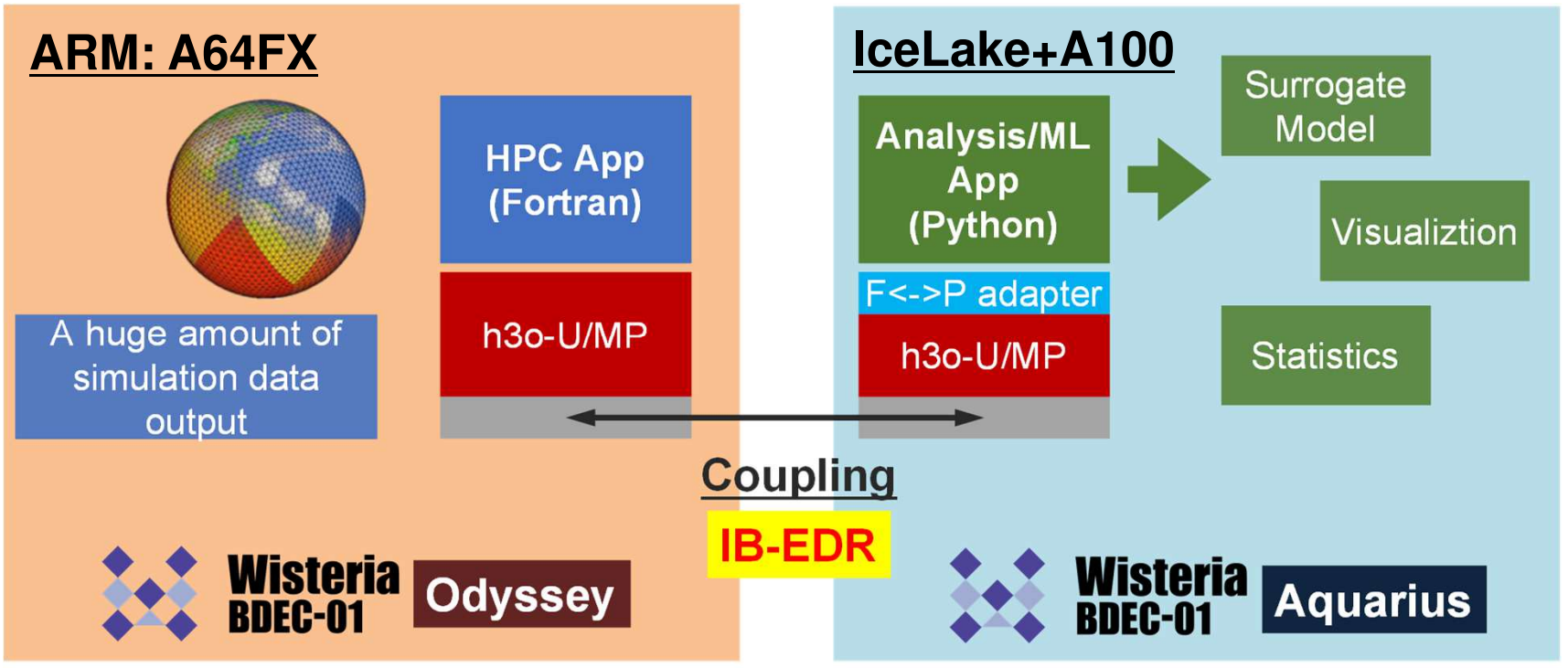
「計算＋データ＋学習」融合を支援する多機能カプラーh3-Open-UTIL/MP

- 異なる物理モデル連成のアンサンブル実行を支援・統合するための機能
  - MPI通信、時刻同期、格子系間マッピング等の管理機能の他、従来のカプラーには無い、複数の弱連成結合シミュレーションのアンサンブル実行、片側のモデルのみをアンサンブル実行する多対1の弱連成結合が可能
  - スパコン上で、全地球大気海洋連成シミュレーションによって動作検証済み
- Fortran/Cコード（物理モデル）とPythonコードの弱連成を実現する機能
  - FortranやCで記述されたプログラム同士の連成計算に限って開発を行ってきたカプラーを、Pythonによって記述されたAI・機械学習、可視化処理系のワークロードからも活用できるよう機能拡充。



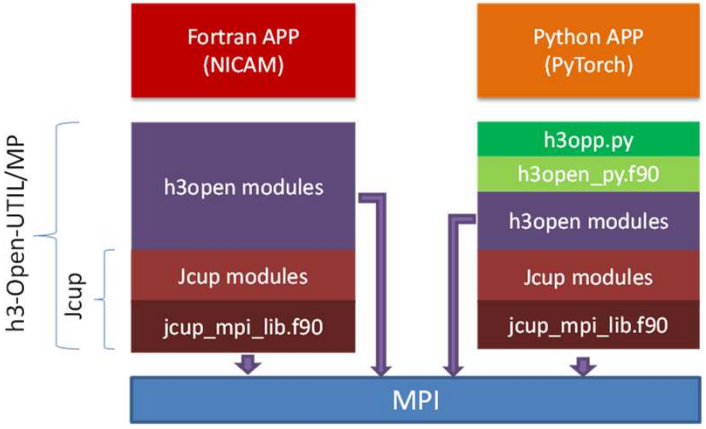
Fortran/CアプリとPythonアプリの連成計算の模式図
〔八代・荒川　2020〕

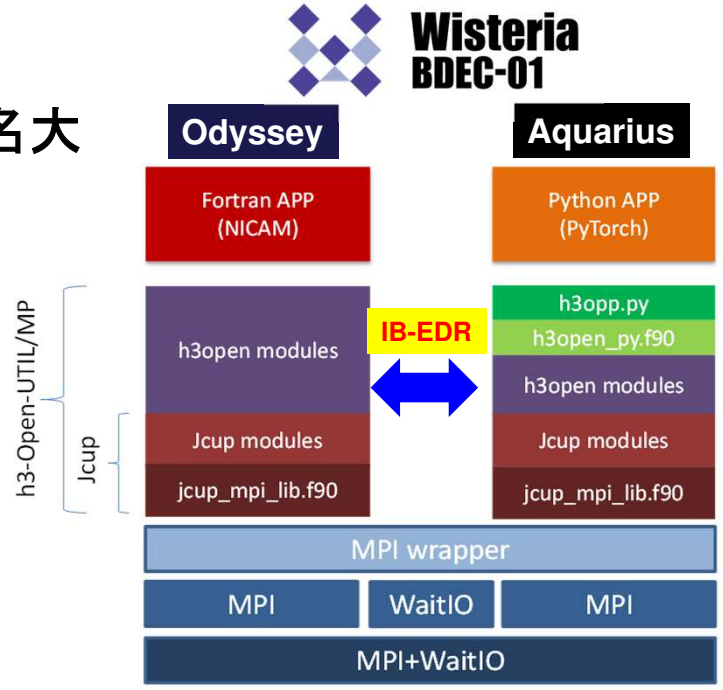
h3-Open-UTIL/MP (h3o-U/MP) + h3-Open-SYS/WaitIO-Socket



h3-Open-UTIL/MP・h3-Open-SYS/WaitIO-Socket連携2022年6月から利用可能

2022年度はFS経由のWaitIO-File整備：名大





2021年4月：MPI通信可能な環境を前提

2022年6月：Coupler＋WaitIO

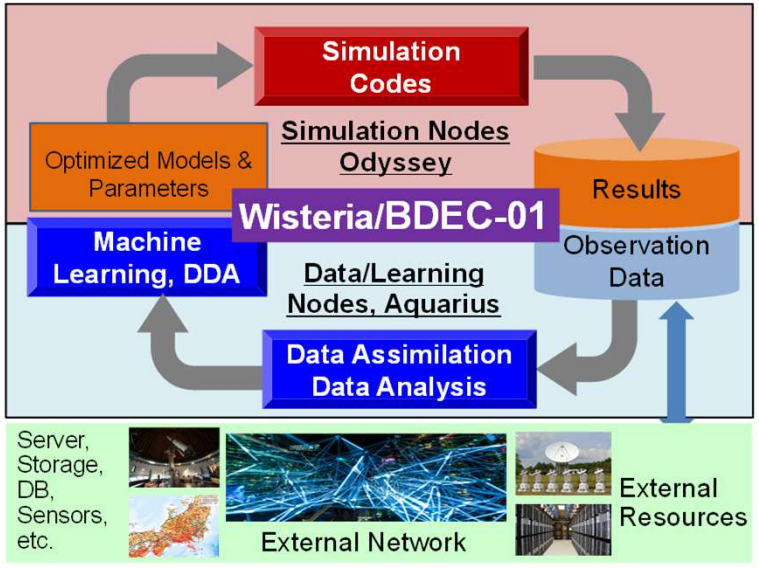
解説記事：h3-Open-UTIL/MP・h3-Open-SYS/WaitIO-Socket

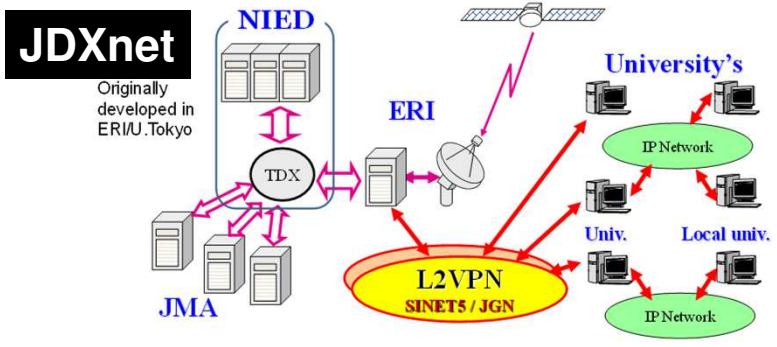
- h3-Open-UTIL/MP
  - https://www.cc.u-tokyo.ac.jp/public/VOL24/No3/13_202205-Wisteria-2.pdf
  - http://nkl.cc.u-tokyo.ac.jp/files/202207UtilMPfinal.pdf

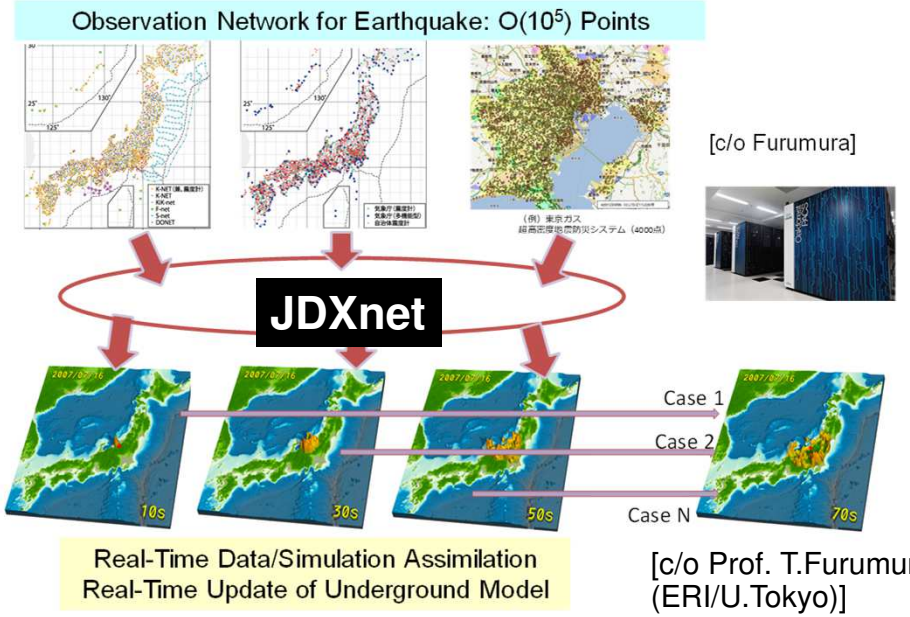
- h3-Open-SYS/WaitIO-Socket
  - https://www.cc.u-tokyo.ac.jp/public/VOL24/No2/10_202203Wisteria-1.pdf
  - https://www.cc.u-tokyo.ac.jp/public/VOL24/No3/12_202205-Wisteria-1.pdf

リアルタイムデータ同化＋3D強震動シミュレーション融合

JDXnetによるリアルタイム観測データ活用

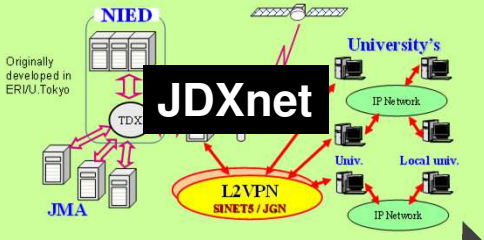






[c/o Prof. T.Furumura (ERI/U.Tokyo)]

長周期地震動シミュレーション＋観測データ同化



Observed Data

Obs. Data

SINET5

External Server for Observed Data

Wisteria BDEC-01

Data/Learning Nodes: Aquarius
Intel Ice Lake + NVIDIA A100
7.20 PF, 578.2 TB/s

2.0 TB/s

Simulation Nodes: Odyssey
Fujitsu/Arm A64FX
25.9PF, 7.8 PB/s

Output & Movie

Filter

Visualizer

Filtered Results

WaitIO-Socket

Comp. Results

Seism3D/ OpenSWPC-DAF

Mesh Data

UG Model

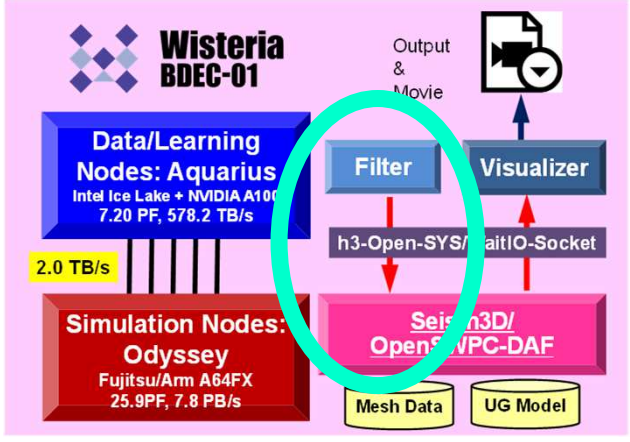
Communications by WaitIO-Socket [Kasai et al. 2021]

Aquarius: SEND

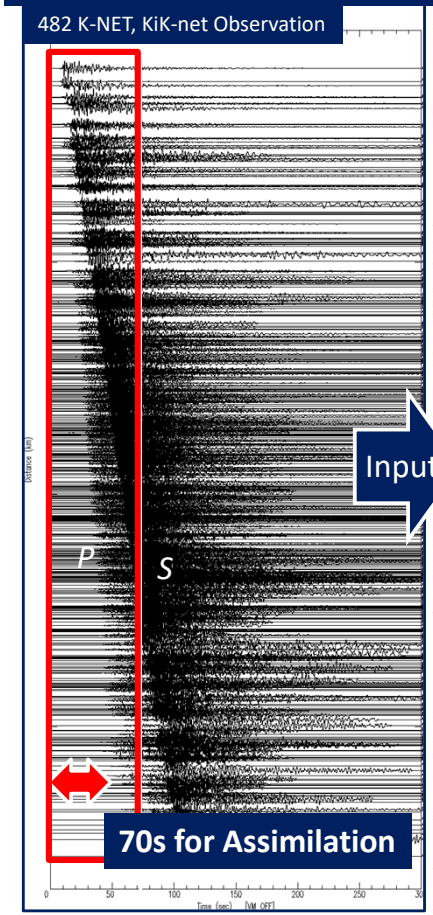
```
program dmy_filter
<省略: 型宣言等>
call mpi_init (ierr)
call mpi_comm_size (MPI_COMM_WORLD, nprocs, ierr)
call mpi_comm_rank (MPI_COMM_WORLD, myrank,  ierr)
call WAITIO_CREATE_UNIVERSE (WAITIO_COMM_UNIVERSE, ierr)

if (myrank==0) then
  open(100,file='./obsfile_list.txt', form='formatted', status='old', iostat=ierr)
  do i=1,300
    <省略: obsデータ読み込み処理>
    print *,"Send obs data ..... "
    call WAITIO_MPI_ISEND (NTMAX1_o, 1,           WAITIO_MPI_INTEGER,  2,1, WAITIO_COMM_UNIVERSE,req(1,1), ierr)
    call WAITIO_MPI_ISEND (DT_o,     1,           WAITIO_MPI_FLOAT,    2,2, WAITIO_COMM_UNIVERSE,req(1,2), ierr)
    call WAITIO_MPI_ISEND (NST_o,    1,           WAITIO_MPI_INTEGER,  2,3, WAITIO_COMM_UNIVERSE,req(1,3), ierr)
    call WAITIO_MPI_ISEND (AT_o,     1,           WAITIO_MPI_FLOAT,    2,4, WAITIO_COMM_UNIVERSE,req(1,4), ierr)
    call WAITIO_MPI_ISEND (T0_o,     1,           WAITIO_MPI_FLOAT,    2,5, WAITIO_COMM_UNIVERSE,req(1,5), ierr)
    call WAITIO_MPI_ISEND (ISO_X_o,  NSMAX,       WAITIO_MPI_INTEGER,  2,6, WAITIO_COMM_UNIVERSE,req(1,6), ierr)
    call WAITIO_MPI_ISEND (ISO_Y_o,  NSMAX,       WAITIO_MPI_INTEGER,  2,7, WAITIO_COMM_UNIVERSE,req(1,7), ierr)
    call WAITIO_MPI_ISEND (ISO_Z_o,  NSMAX,       WAITIO_MPI_INTEGER,  2,8, WAITIO_COMM_UNIVERSE,req(1,8), ierr)
    call WAITIO_MPI_ISEND (ISTX_o,   NST,         WAITIO_MPI_INTEGER,  2,9, WAITIO_COMM_UNIVERSE,req(1,9), ierr)
    call WAITIO_MPI_ISEND (ISTY_o,   NST,         WAITIO_MPI_INTEGER,  2,10,WAITIO_COMM_UNIVERSE,req(1,10),ierr)
    call WAITIO_MPI_ISEND (ISTZ_o,   NST,         WAITIO_MPI_INTEGER,  2,11,WAITIO_COMM_UNIVERSE,req(1,11),ierr)
    call WAITIO_MPI_ISEND (STC_o,    6*NST,       WAITIO_MPI_CHAR,     2,12,WAITIO_COMM_UNIVERSE,req(1,12),ierr)
    call WAITIO_MPI_ISEND (VxAll_obs,NST*NOBS_LEN,WAITIO_MPI_FLOAT,    2,13,WAITIO_COMM_UNIVERSE,req(1,13),ierr)
    call WAITIO_MPI_ISEND (VyAll_obs,NST*NOBS_LEN,WAITIO_MPI_FLOAT,    2,14,WAITIO_COMM_UNIVERSE,req(1,14),ierr)
    call WAITIO_MPI_ISEND (VzAll_obs,NST*NOBS_LEN,WAITIO_MPI_FLOAT,    2,15,WAITIO_COMM_UNIVERSE,req(1,15),ierr)
    call WAITIO_MPI_WAITALL (15,req, status, ierr)
    call sleep(1)
  enddo
  close (100)
endif
call WAITIO_FINALIZE (ierr)
call mpi_finalize (ierr)
end
```

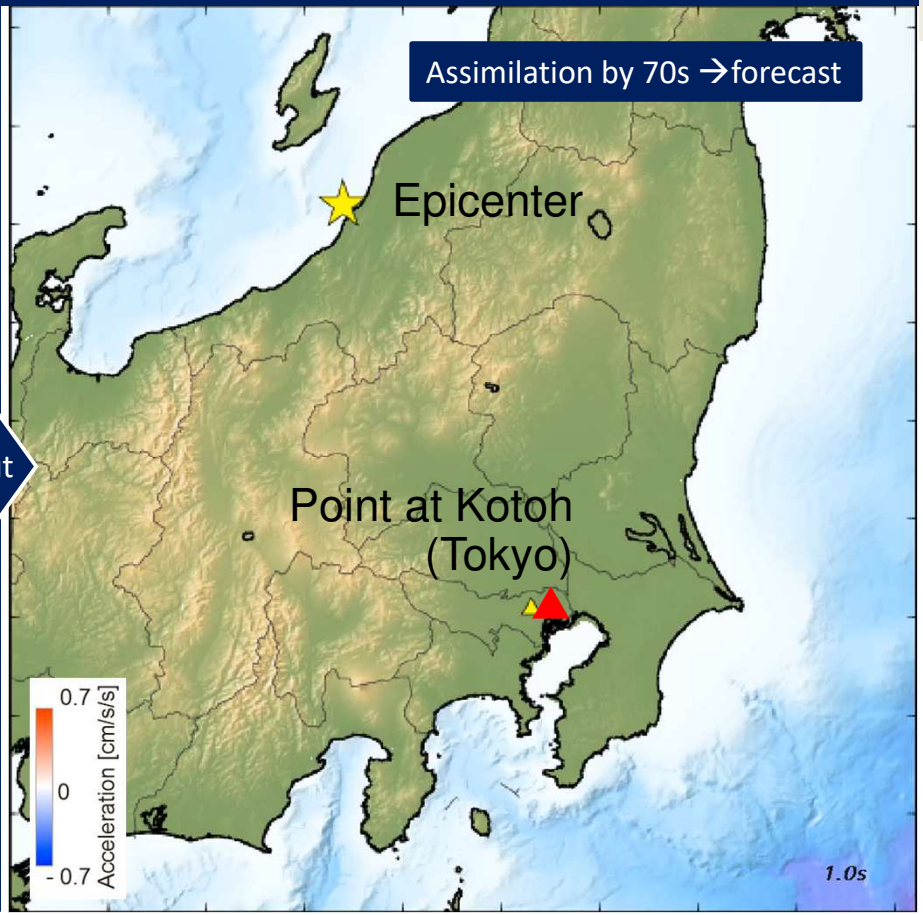
Odyssey: RECV

```
call WAITIO_MPI_IRECV (NTMAX1_o, 1,           WAITIO_MPI_INTEGER, 0,1, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (DT_o,     1,           WAITIO_MPI_FLOAT,   0,2, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (NST_o,    1,           WAITIO_MPI_INTEGER, 0,3, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (AT_o,     1,           WAITIO_MPI_FLOAT,   0,4, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (T0_o,     1,           WAITIO_MPI_FLOAT,   0,5, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (ISO_X_o,  NSMAX,       WAITIO_MPI_INTEGER, 0,6, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (ISO_Y_o,  NSMAX,       WAITIO_MPI_INTEGER, 0,7, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (ISO_Z_o,  NSMAX,       WAITIO_MPI_INTEGER, 0,8, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (ISTX_o,   NST,         WAITIO_MPI_INTEGER, 0,9, WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (ISTY_o,   NST,         WAITIO_MPI_INTEGER, 0,10,WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (ISTZ_o,   NST,         WAITIO_MPI_INTEGER, 0,11,WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (STC_o,    6*NST,       WAITIO_MPI_CHAR,    0,12,WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (VxAll_obs,NST*NOBS_LEN,WAITIO_MPI_FLOAT,   0,13,WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (VyAll_obs,NST*NOBS_LEN,WAITIO_MPI_FLOAT,   0,14,WAITIO_COMM_UNIVERSE,...)
call WAITIO_MPI_IRECV (VzAll_obs,NST*NOBS_LEN,WAITIO_MPI_FLOAT,   0,15,WAITIO_COMM_UNIVERSE,...)
```

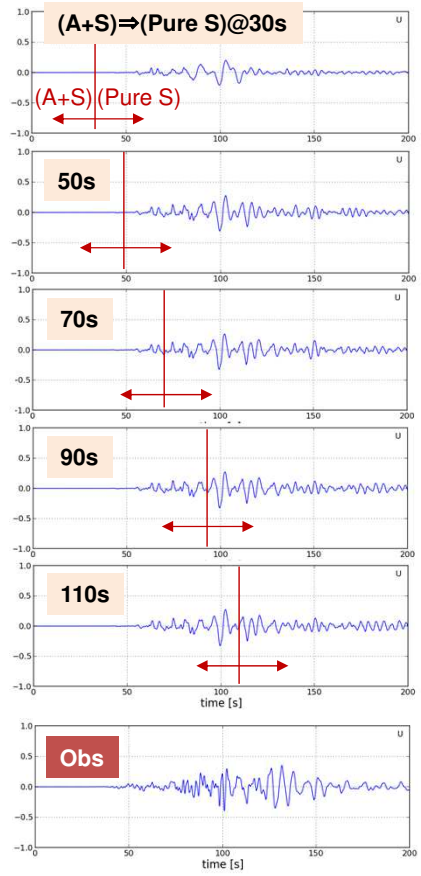


Data Assimilation + Pure Simulation/Forecast



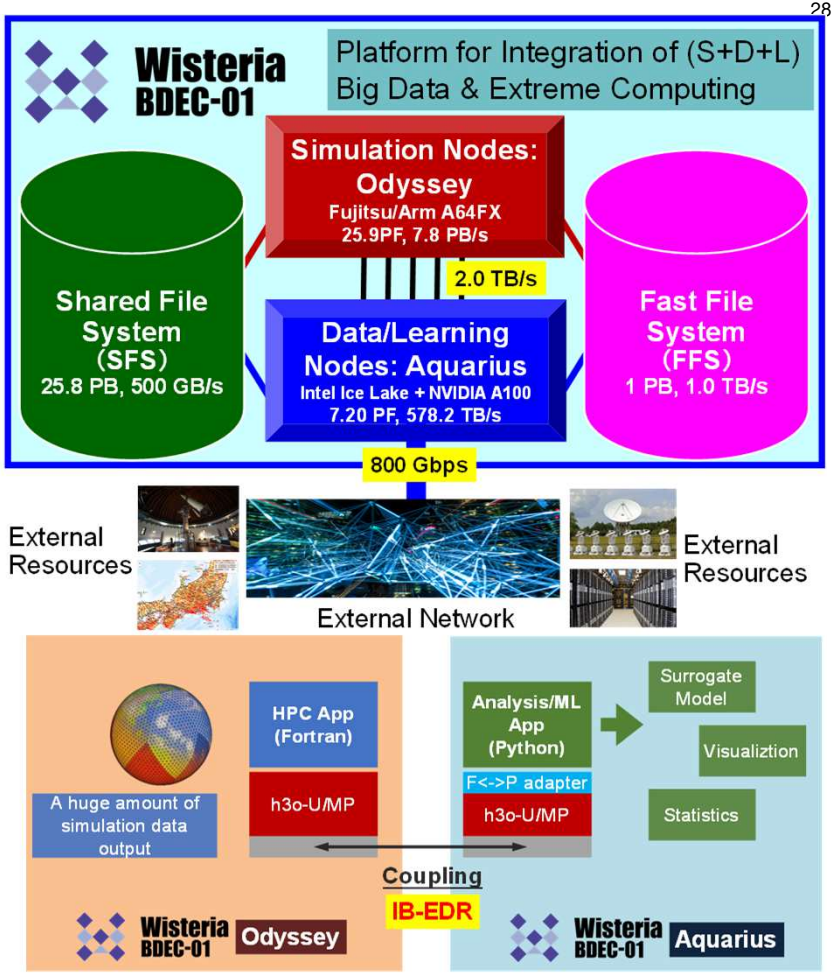


Results at Kotoh ▲ (N.KOTH)
N 35° 37.0'
E 139° 46.9'



Odyssey-Aquarius連携

- 総ノード数
  - Odyssey：7,680ノード，やや空いている
  - Aquarius: 45ノード, 360 GPUs，混雑
- Aquariusのうち1ノードを（S+D+L）融合型ワークロード向けにリザーブ
  - Odyssey，Aquariusそれぞれに対する2つのジョブスクリプトをサブミットする必要がある
  - 両ジョブがリソースを確保⇒実行開始
- より柔軟な仕組みを整える必要あり
  - このようなシステム，運用例は世界的に見ても例がほとんどない



ジョブスクリプト例 [Sumimoto, Arakawa]

Odyssey for Simulation

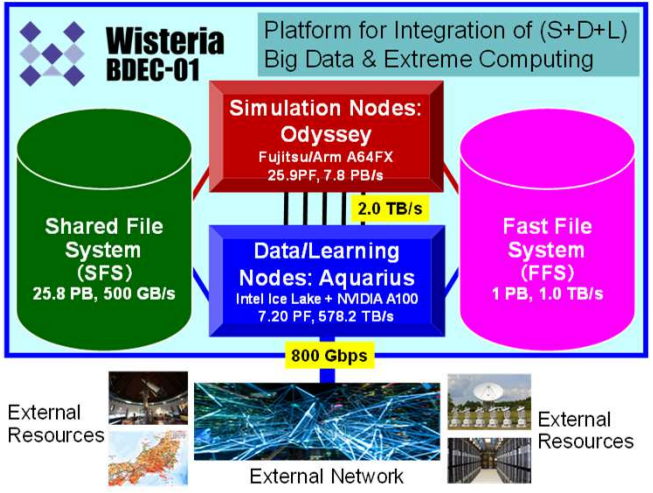
```
#!/bin/bash
#PJM -N "test_waitio"
#PJM -L rscgrp=coupler-lec-o
#PJM -L node=10:noncont
#PJM --mpi proc=80
#PJM -L elapse=00:10:00
#PJM -g gt00
#PJM -j
#PJM -e err

module load fj
module load fjmpi
module load waitio

export WAITIO_MASTER_HOST=`hostname`
export WAITIO_MASTER_PORT=7100
export WAITIO_PBID=0
export WAITIO_NPB=2

hostname
waitio-serv-a64fx -d -m $WAITIO_MASTER_HOST

#mpiexec -oferr-proc errnicam -np 160 ./nicam
mpiexec -np 80 ./nicam
```

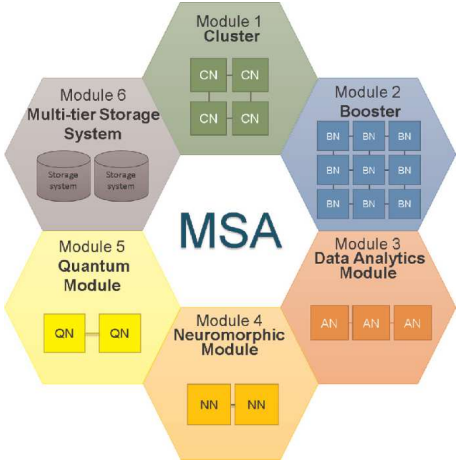
Aquarius for AI

```
#!/bin/bash
#PJM -N "test_waitio"
#PJM -L rscgrp=coupler-lec-a
#PJM -L node=1
#PJM --mpi proc=10
#PJM -L elapse=00:10:00
#PJM -g gt00
#PJM -j
#PJM -e err

module unload aquarius
module unload gcc ompi
module load intel
module load impi
module load waitio

export WAITIO_MASTER_HOST=`waitio-serv -c`
export WAITIO_MASTER_PORT=7100
export WAITIO_PBID=1
export WAITIO_NPB=2

module unload intel
module unload impi
module load gcc ompi

mpiexec -n 10 ./ada
```

Heterogeneous Architecture

- **If you want to integrate (Simulation/Data/Learning) (S+D+L), the architecture should be heterogeneous (My Personal Perspective)**
  - **Wisteria/BDEC-01 (U.Tokyo)**
  - **Modular Supercomputing Architecture (JSC, Germany)**





Joint Proposal for FY.2023 JHPCN (accepted)

https://jhpcn-kyoten.itc.u-tokyo.ac.jp/en/

- Innovative Computational Science by Integration of Simulation/Data/Learning under Heterogeneous Computing Environment
  - FY.2021 & 2022: Focused on Earthquake Simulations
    - Univ. Tokyo (ITC, ERI), Nagoya U., Kyushu U., NIES, Fujitsu
  - FY.2023-2025 (plan): Other applications and International Collaborations
    - Jülich Supercomputing Centre（JSC）：Modular Supercomputing
    - Rudjer Boskovic Institute, Centre for Informatics and Computing, Croatia
    - Friedrich-Alexander-Universität Erlangen-Nürnberg（FAU）
    - French Atomic Energy Commission (CEA)
- Target Systems in Japan
  - Wistreia/BDEC-01，Flow@Nagoya U., mdx

JÜLICH FORSCHUNGSZENTRUM

Institut Ruđer Bošković

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

cea