Pillar II: Dynamic and adaptive workflows for climate modelling

Alessandro D’Anca – CMCC

Euro-mediterranean Center on Climate Change (CMCC)

Barcelona Supercomputing Center (BSC)

Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung (AWI)

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955558. The JU receives support from the European Union’s Horizon 2020 research and innovation programme and Spain, Germany, France, Italy, Poland, Switzerland, Norway.

Objectives

The goal of the Pillar II is to take advantage of the **eFlows4HPC architecture** to enhance innovation for intelligent and integrated end-to-end **HPDA-enabled workflow for Earth System Models (ESM)**.

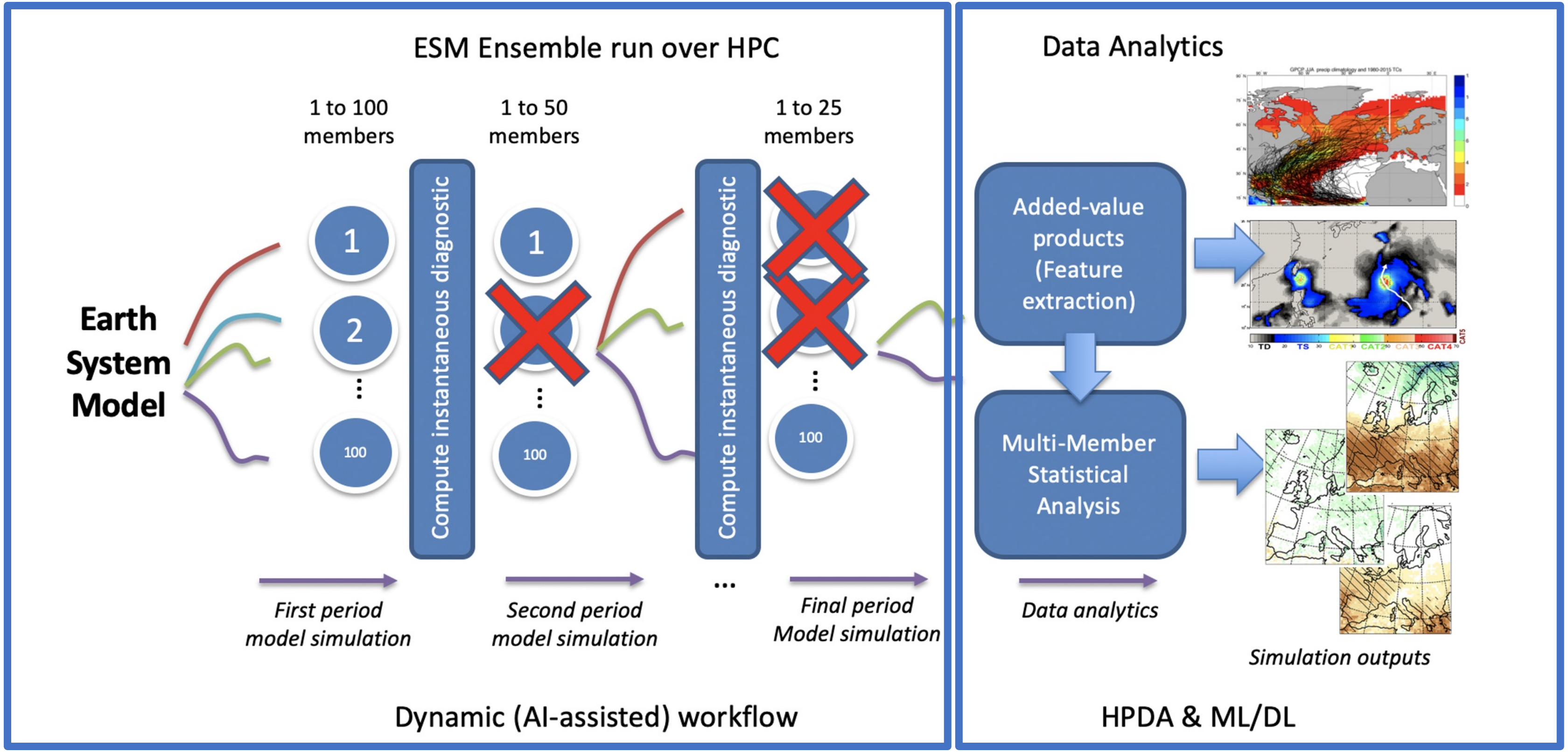
- Development of intelligent and novel end-to-end ESM workflows able to (i) rapidly **adapt and evolve** according to the dynamic conditions of the climate simulations, and (ii) make a **better use of computational and storage** resources by performing a smart (AI-driven) pruning of ensemble members (and releasing resources accordingly) at runtime;
- Seamless, transparent and efficient **integration of different components of the ESM workflow** (from simulation, to post-processing, HPDA and learning) into the same experiment, overcoming current gaps and barriers;
- Development of **ML/DL models** to help understanding and to produce added-value products (i.e. TC detection) from climate simulations;
- Evaluation of **data-intensive vs. data-driven approaches** with respect to key features (like TC detection).

Two workflows:

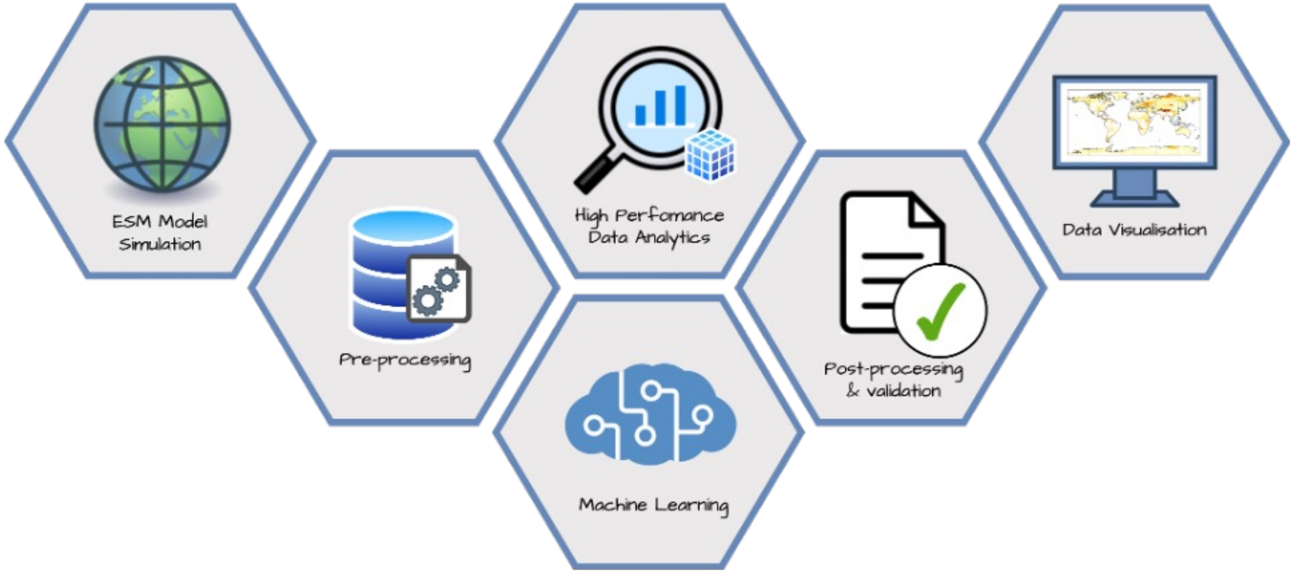
- **Statistical analysis and feature extraction workflow**
- **Dynamic (AI-assisted) ESM workflow**

Pillar II General Overview

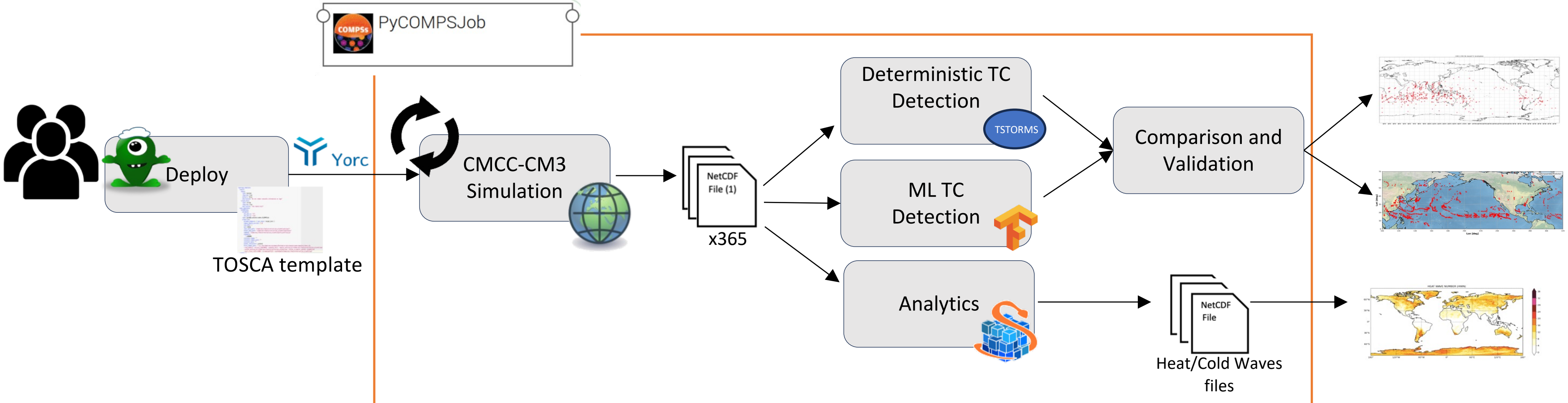
Two macro-workflow defined:



STATISTICAL ANALYSIS AND FEATURE EXTRACTION WORKFLOW



Statistical analysis and feature extraction



- **High level workflow** orchestrated by Alien4Cloud and YORC
- **Tasks workflow** managed by PyCOMPSs
- Based on **CMCC-CM3** coupled climate model
- CMCC-CM3 production drives the subsequent blocks
- **Deterministic** vs **ML based** TC detection and mutual comparison
- **Data analytics** performed by **Ophidia**
- Produced outputs moved to **B2SHARE** repo by **DLS**

DLS

B2SHARE EUDAT

CMCC-CM3 model

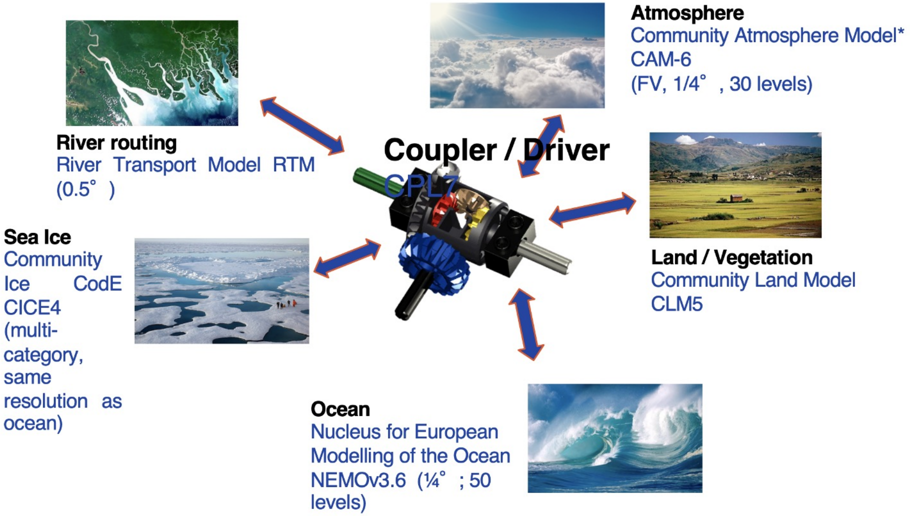
State of the art CMCC-CM3 climate model CAM6 – NEMO3.6

CMCC-CM3 fully coupled

Resolution: 25 km Atmosphere,
25 km degree Ocean

Number of cores: 540 Atmosphere,
180 Ocean

SYPD: ~0.3

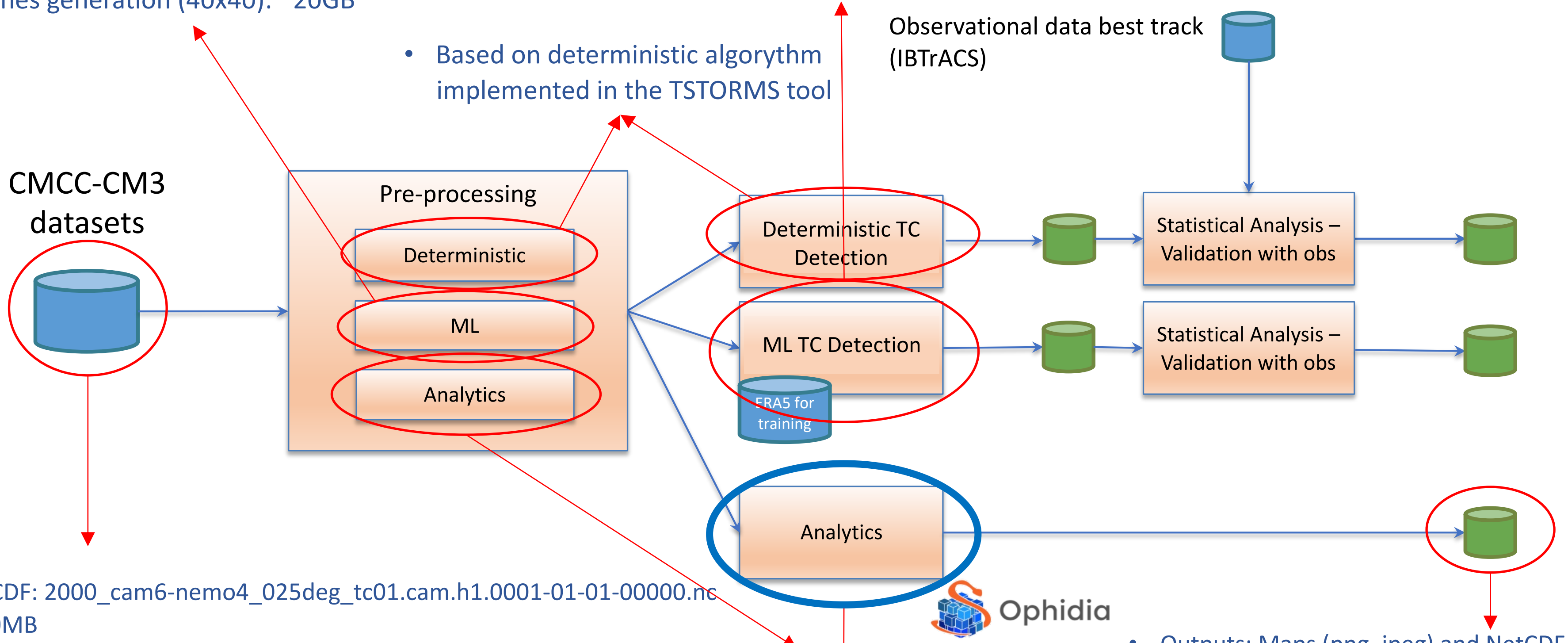


Statistical analysis and feature extraction workflow

- Input data for NN training
  ERA5 NetCDF (North Pacific region): ~240GB
- Patches generation (40x40): ~20GB

- Inference on CMCC-CM3 datasets

- Based on deterministic algorythm implemented in the TSTORMS tool



- NetCDF: 2000_cam6-nemo4_025deg_tc01.cam.h1.0001-01-01-00000.nc
  ~200MB
  4 timesteps (6 hourly)
  Global domain

- Climate indicators related to Heat and Cold Waves

- Outputs: Maps (png, jpeg) and NetCDF

Ophidia HPDA framework

Ophidia (http://ophidia.cmcc.it) is a CMCC Foundation research project addressing data challenges for eScience

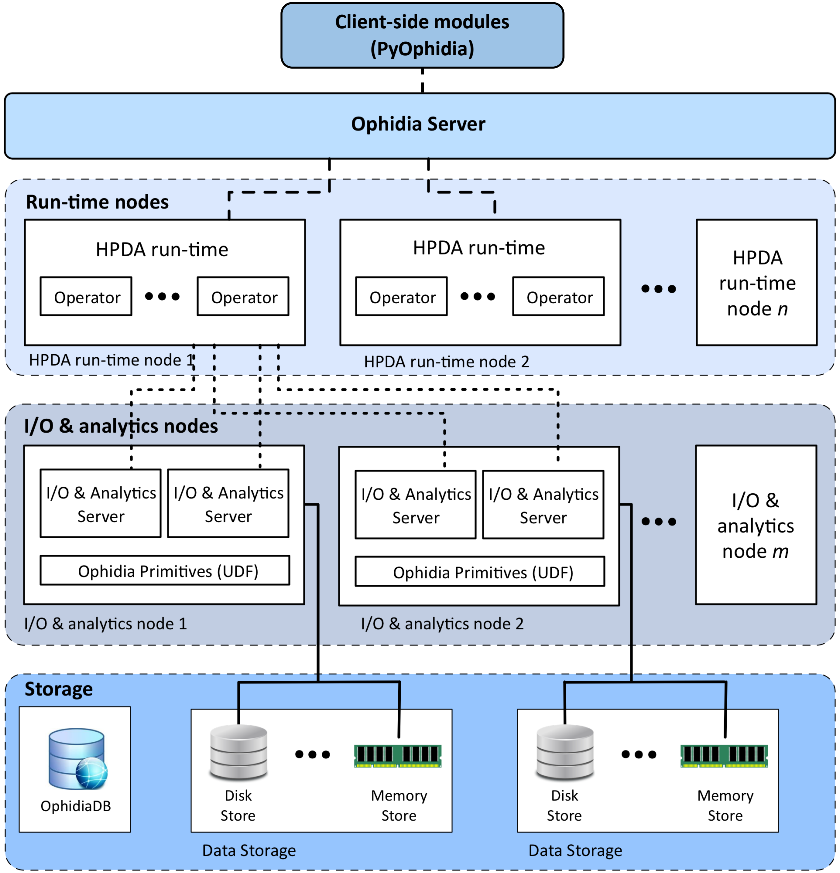
- A **HPDA framework** for multi-dimensional scientific data joining HPC paradigms with scientific data analytics approaches
- **In-memory** and **server-side** data analysis exploiting parallel computing techniques
- Multi-dimensional, array-based, storage model and partitioning schema for scientific data leveraging the **datacube** abstraction
- Support for **interactive analysis, complex experiments** and **workflows** on scientific data

S. Fiore, D. Elia, C. Palazzo, F. Antonio, A. D'Anca, I. Foster, G. Aloisio, "Towards High Performance Data Analytics for Climate Change", ISC High Performance 2019, LNCS Springer, 2019

Ophidia architecture

The framework has been enhanced to support **large-scale HPDA use cases**:

- **Modular, extensible and scalable** software stack
- **User-friendly** Python interface (PyOphidia)
- **HPDA runtime** for executing parallel data operators
- Support for **in-memory analytics**
- Data partitioned in binary arrays and distributed across the I/O & analytics nodes using a **key-value approach**

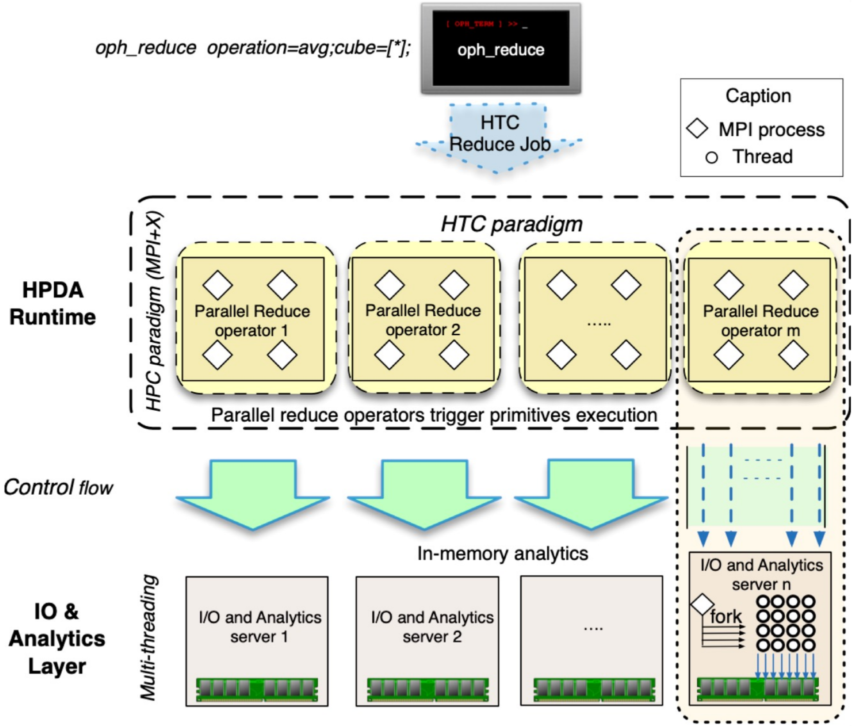


A parallel runtime for HPDA

Hierarchical parallel execution model for data analytics functions, with two levels of parallelism:

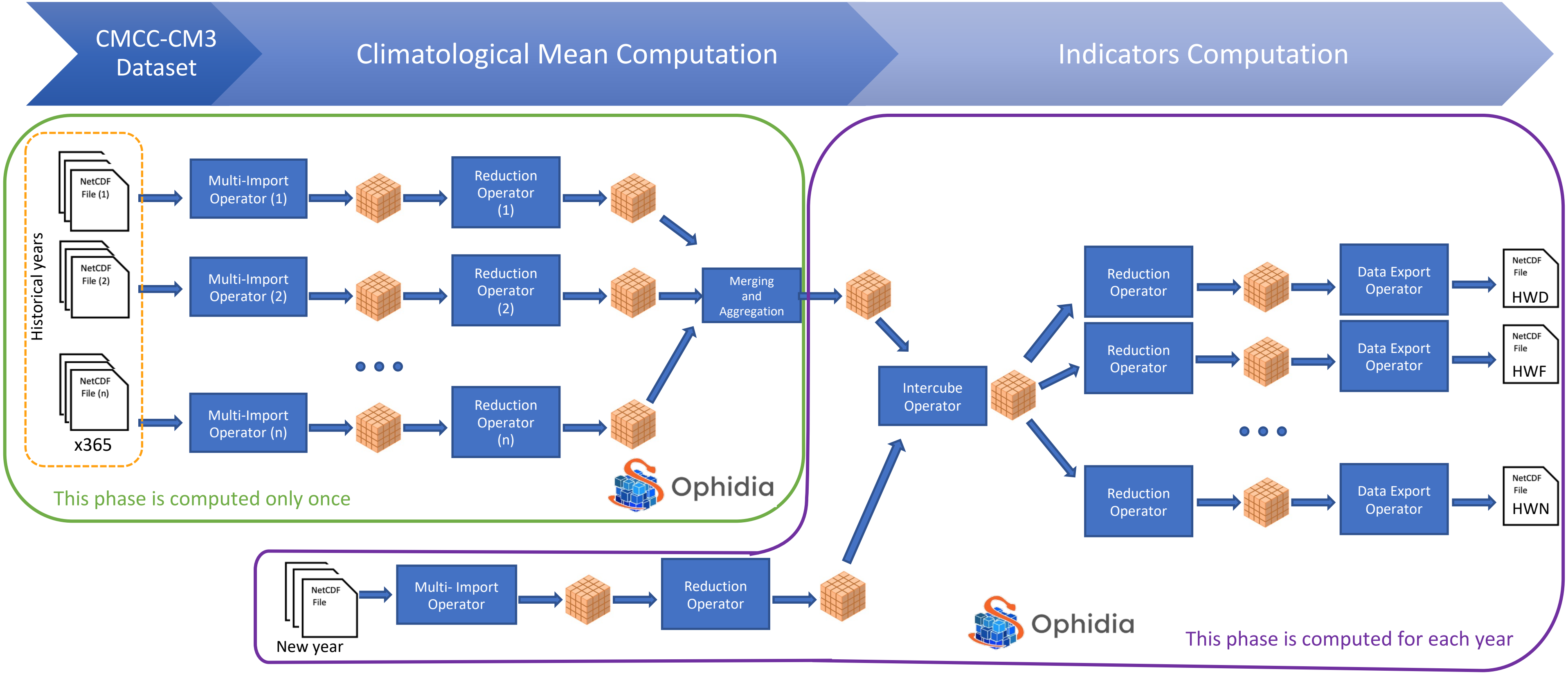
- **Datacube-level**: execute multiple operators on different input data (HTC paradigm)
- **Fragment-level**: MPI+X model for execution of single analytics operators on a datacube (HPC paradigm)
    - **Multi-thread** for intra-node parallelism
    - **MPI** to scale processing on multiple nodes

The analytics function on the data fragments are managed and executed by the I/O servers.



D. Elia, S. Fiore and G. Aloisio, "Towards HPC and Big Data Analytics Convergence: Design and Experimental Evaluation of a HPDA Framework for eScience at Scale," in IEEE Access, vol. 9, pp. 73307-73326, 2021

Analytics block: Indicators Computation Workflow



Analytics block: extreme events indicators

eFlows4HPC

Index	Name	Description
HWD	Heat Wave Duration	Starting from the daily maximum temperature (**TSMX**), the Heat Wave Duration index is the maximum number of days at intervals of at least 6 days with TSMX > 5°C + baseline **BASELINE**: Average calculated for each calendar day (based on 30 years) using a 5-day window
CWD	Cold Wave Duration	Starting from the daily minimum temperature (**TSMN**), the Cold Wave Duration index is the maximum number of days at intervals of at least 6 days with TSMN < 5°C + baseline
HWN	Heat Wave Number	Number of heatwaves in a year
CWN	Cold Wave Number	Number of coldwaves in a year
HWF	Heat Wave Frequency	Number of days that contribute to heatwaves in a year
CWF	Cold Wave Frequency	Number of days that contribute to coldwaves in a year

Analytics block: baseline computation

Average calculated for each calendar day using a 5-day window

Input

Input files: daily CMCC-CM3 NetCDF files (365 x 20 years)
Dimension: 14 MB x 2 (TSMX/TSMN). Total expected ~200 GB
Measure: Surface temperature TSMX/TSMN

Calculation of baseline:

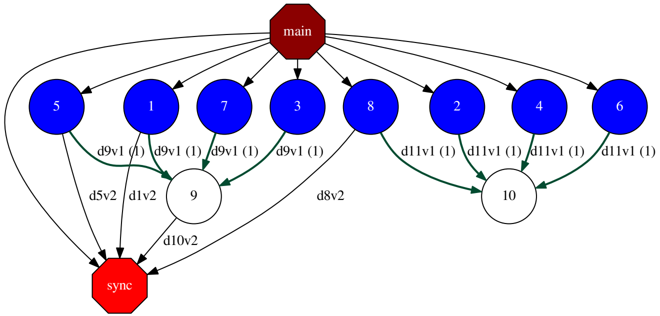
Declaring tasks

```
@task(returns=object)
def OphidiaImport(file, measure, op):
    src_path='2000_cam6-nemo4_025deg_tc/atm/hist/2000_cam6-nemo4_025deg_tc.cam.h1.' + file + '-*-*-00000.nc'
    Year = cube.Cube.importncs(src_path= src_path,
    measure=measure,
    imp_dim='time',
    description='6-Hours Temps'
    )
    movingAvg = Year.apply(
    query="oph_shift(oph_moving_avg(oph_reduce2(measure," + op + ",4), 5, 'OPH_SMA'), -2, 0)",nthreads=2)
    return movingAvg

@task(returns=object, movingavgcubes=COLLECTION_IN)
def OphidiaClimatologicalAvg(movingavgcubes, filename):
    baseline=cube.Cube.intercube2(cubes=movingavgcubes)
    baseline.exportnc2(output_path="/work/asc/ss18121/CMCC-CM3/HeatWaveNotebooks",output_name=filename)
    return baseline
```

Invoking tasks

```
num_files=8
maxdatacubes = [0 for i in range(num_files)]
mindatacubes = [0 for i in range(num_files)]
for i in range(num_files):
#Import of all historical files
#computation of the max or min on the 6-hours values and of the moving average on a 5-day window
    maxdatacubes[i] = OphidiaImport(str(i).zfill(4), 'TSMX', "'OPH_MAX'")
    mindatacubes[i] = OphidiaImport(str(i).zfill(4), 'TSMN', "'OPH_MIN'")
#Computation of the baseline and export in NetCDF files
climatological_avg_tasmax = OphidiaClimatologicalAvg(maxdatacubes, "climatological_avg_tasmax")
climatological_avg_tasmin = OphidiaClimatologicalAvg(mindatacubes, "climatological_avg_tasmin")
```

Based on Python: PyOphidia and PyCOMPSs



PyCOMPSs execution graph

Datacubes with the baseline values
(for max and min temperature variables)

Output

Analytics block: indicators computation

eFlows4HPC

Computation of 6 indicators on each year produced by the model

Input

Baseline: datacubes from the first stage
Files: daily CMCC-CM3 NetCDF files (x365)

Computation of Heat Wave/Cold Wave Indicators:

Declaring tasks

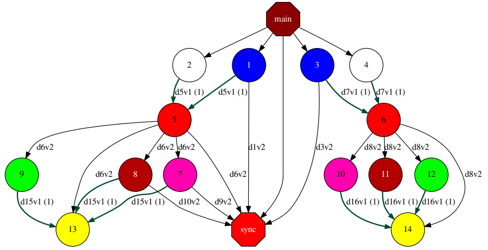
```
@task(returns=object, cubes=COLLECTION_IN)
def OphidiaCompare(cubes, firstvalues, mask):
    #Computation of the difference between a year and the baseline;
    Diff = cubes[0].intercube(cube2=cubes[1].pid, operation='sub')
    Duration = Diff.apply(
    query="oph_predicate(oph_sequence(oph_predicate("+firstvalues+",'x-5',"+mask+",'1','0')),'x-5','>0','x','0')")
    return Duration

@task(returns=object)
def IndexDurationNumber(duration, filename):
    #Computation of the number of durations in a year (HWN or CWN)
    DurationMask = duration.apply(query="oph_predicate('OPH_INT','OPH_INT',measure,'x','>0','1','0')")
    DurationCount = DurationMask.reduce(operation='sum', ncores=1, description="Number of durations cube")
    return DurationCount
```

• • •

Invoking tasks

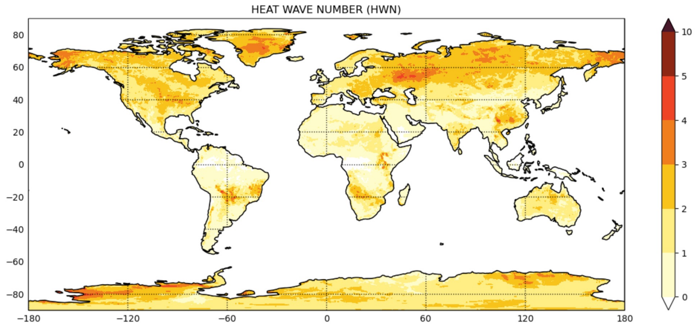
```
hwdi = OphidiaCompare(maxcubes, "oph_predicate('OPH_FLOAT','OPH_FLOAT',measure,'x-100','>0','0','x')", "'>0'")
cwdi = OphidiaCompare(mincubes, "measure", "'<0'")
hwn = IndexDurationNumber(hwdi, 'HWN')
cwn = IndexDurationNumber(cwdi, 'CWN')
```

Based on Python: PyOphidia and PyCOMPSs, Cartopy



PyCOMPSs execution graph

Number of heat waves (HWN) in a year for each (lat,lon) point on map

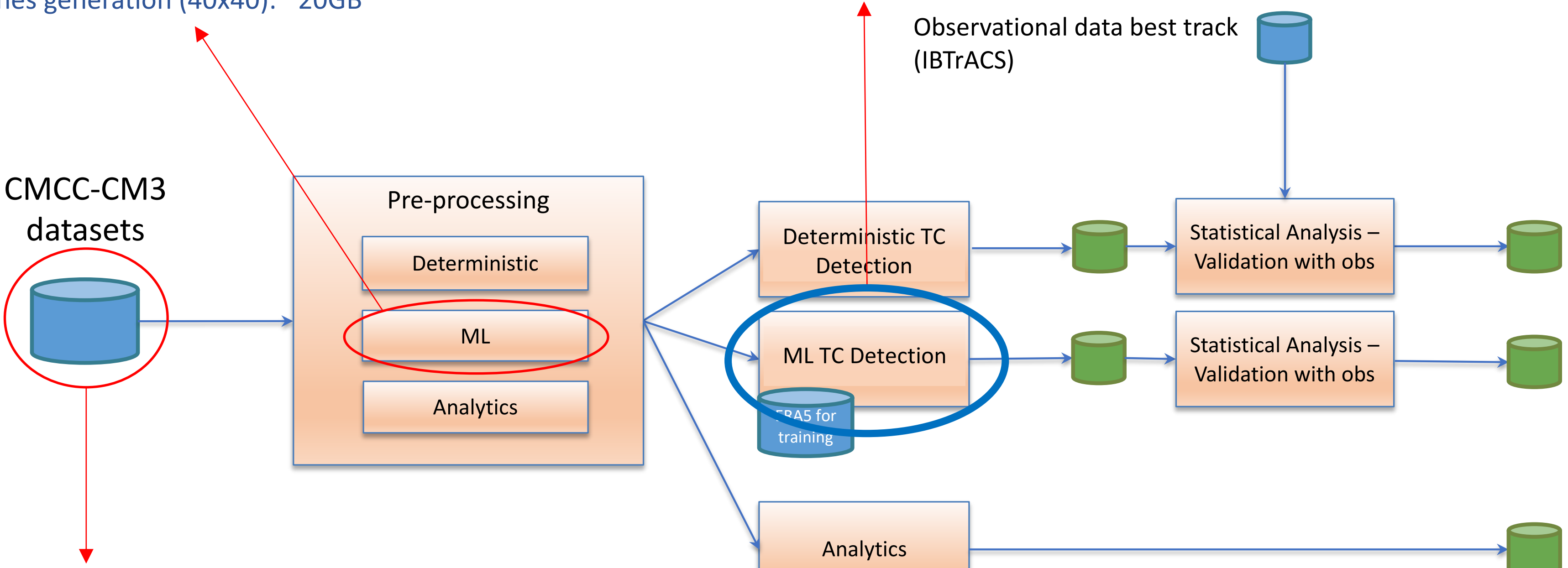


Output

Statistical analysis and feature extraction workflow

- Input data for NN training
  ERA5 NetCDF (North Pacific region): ~240GB
- Patches generation (40x40): ~20GB

- Inference on CMCC-CM3 datasets



- NetCDF: 2000_cam6-nemo4_025deg_tc01.cam.h1.0001-01-01-00000.nc
  ~200MB
  4 timesteps (6 hourly)
  Global domain

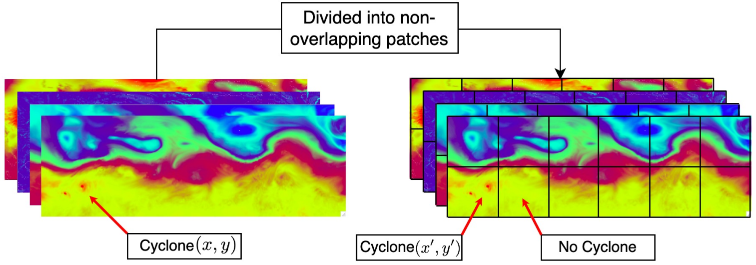
Tropical Cyclones Detection ML Workflow: Training Phase

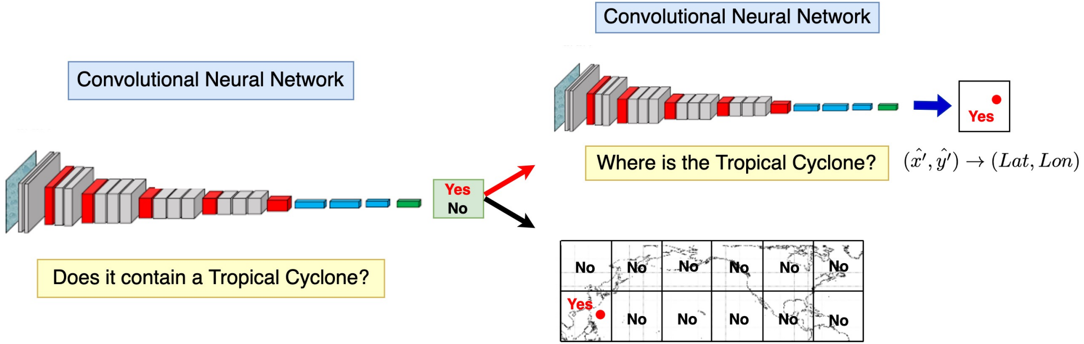
eFlows4HPC

Stage 1:
ERA5 Data Collection + Labeling (IBTrACS)

Stage 2:
Classification

Stage 3:
Localization





ERA5 Climate drivers:

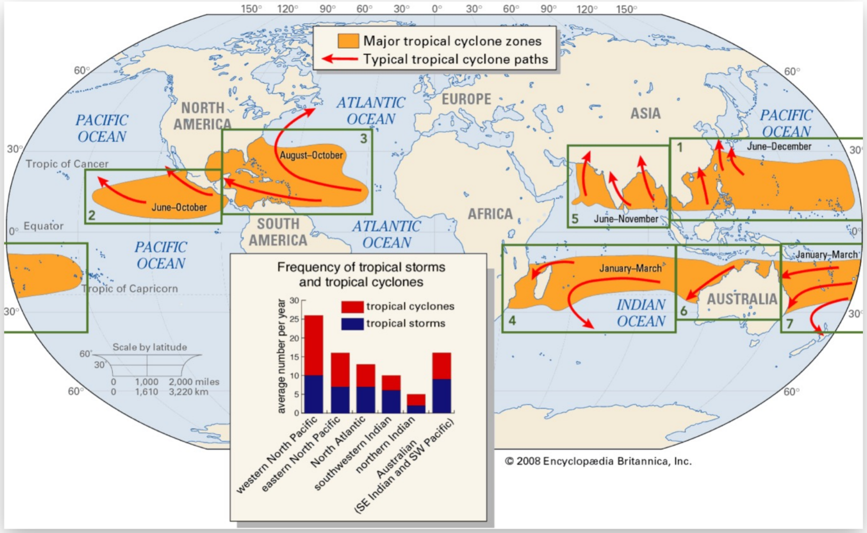
- 10 m wind gust
- Temperature at 500 hPa
- Temperature at 300 hPa
- Mean sea level pressure

- Three main stages are involved:
  - **STAGE 1:** Generation of patches containing at most 1 tropical Cyclone (TC) each starting from ERA5 data
  - **STAGE 2:** Classification of TC presence/absence inside the patch
  - **STAGE 3:** Localization of TC center coordinates in the patches in which the TC was previously classified as present

TC ML Detection: Training phase – Experimental Setup

Data sources

- **ERA5 hourly data on single levels from 1979 to present (0.25° x 0.25°, 3 hourly)**
  - Mean Sea Level pressure
  - 10m wind gust
- **ERA5 hourly data on pressure levels from 1979 to present (0.25° x 0.25°, 3 hourly)**
  - Temperature at 300 mb
  - Temperature at 500 mb
- **International Best Track Archive for Climate Stewardship (IBTrACS)**
  - Lat/Lon of TC center



➢ TC FORMATION BASINS:

1. **Northwest Pacific**
2. **Northeast Pacific**
3. North Atlantic
4. Southwest Indian Ocean
5. North Indian Ocean
6. Southeast Indian Ocean
7. Southwest Pacific

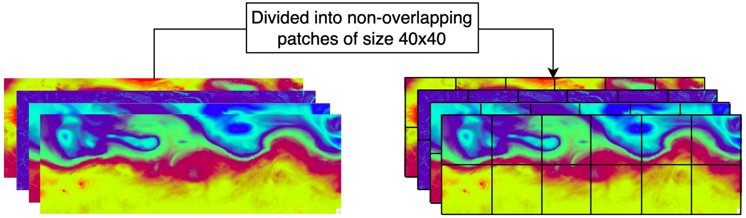
→ 100 – 320 °E, 0 – 70 °N

TC ML Detection: Inference Phase on CMCC-CM3 data

CMCC-CM3 Data Collection

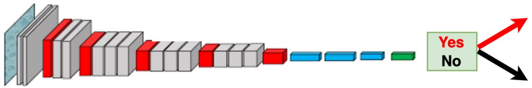
Classification
through the NN pre-trained on ERA5

Localization
through the NN pre-trained on ERA5



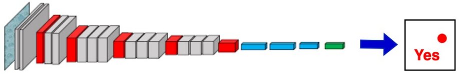
A single map is divided into 532 (19x28) patches of size **40x40**

Convolutional Neural Network



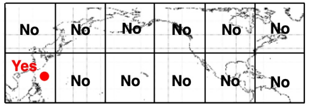
Does it contain a Tropical Cyclone?

Convolutional Neural Network



Where is the Tropical Cyclone?

$(\hat{x}', \hat{y}') \rightarrow (Lat, Lon)$



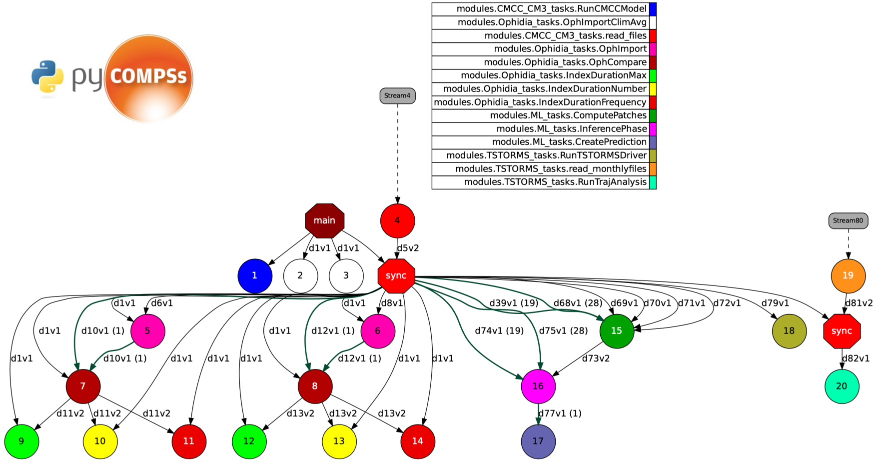
CMCC-CM3 Climatic maps:

- Horizontal Resolution: ~ 0.25° x 0.25°
- Temporal Resolution: 6 hourly
- Variables: the same involved in the training phase, i.e.:
  - 10m wind gust (WSPDSRFMX)
  - Temperature at 500 hPa (T500)
  - Temperature at 300 hPa (T300)
  - Mean sea level pressure (PSL)

Feature extraction workflow

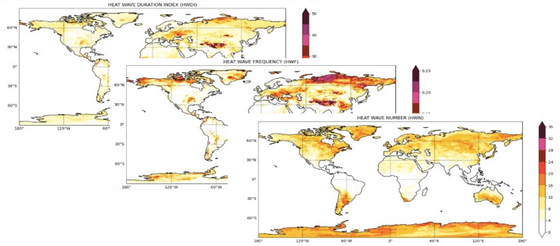
Based on PyCOMPSs

```
enqueue_compss --
sc_cfg=/pycompss/Runtime/scripts/queues/super
computers/zeus.cfg --num_nodes=2 -d --
pythonpath=$PWD --project_name=0459 --
queue=p_short --streaming=FILES --
master_working_dir=$PWD --
worker_working_dir=$PWD --
worker_in_master_cpus=0 --graph=true --
wall_clock_limit=14400 --exec_time=120
feature_extraction.py
```

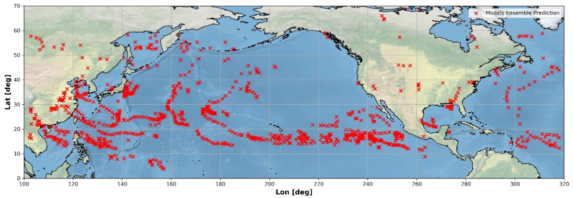


Output

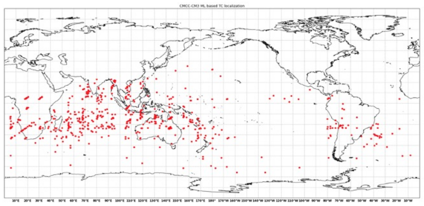
Extreme events indicators (HWD, HWF, HWN) in a year for each (lat,lon) point on map



CMCC-CM3 based Tropical Cyclone localization on map

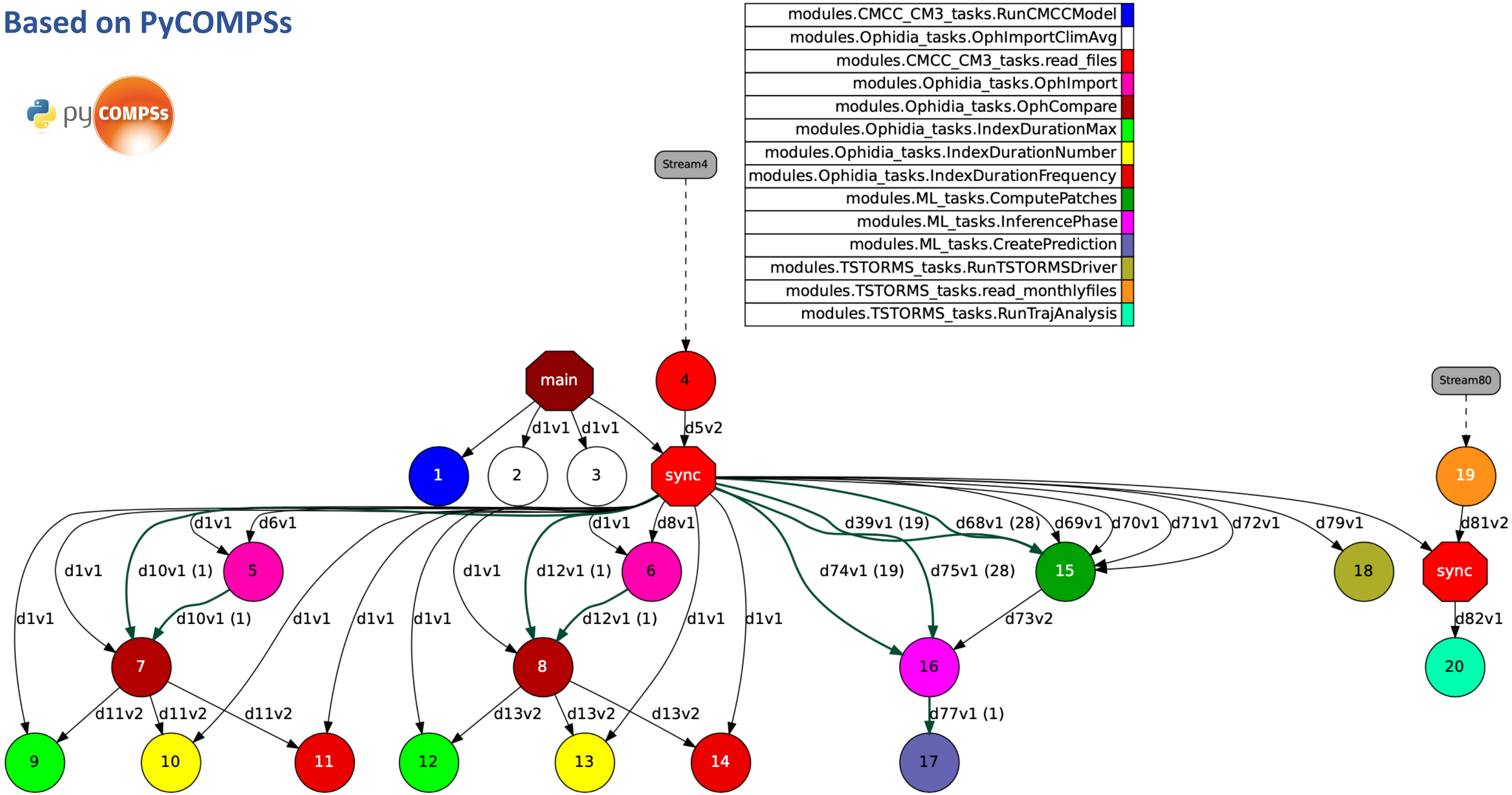


TSTORMS based Tropical Cyclone localization on map



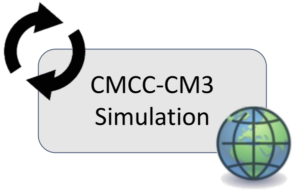
Feature extraction workflow

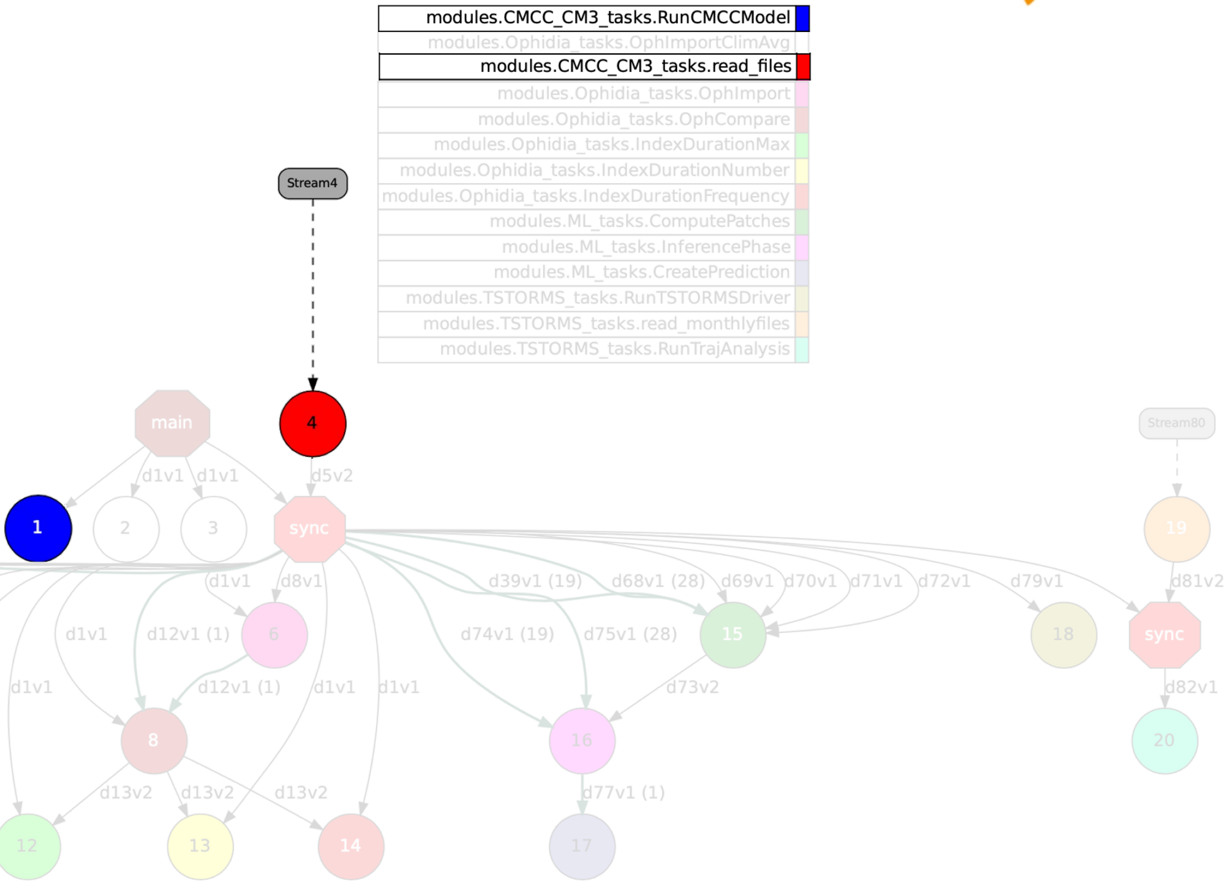
Based on PyCOMPSs



Feature extraction workflow

Based on PyCOMPSs

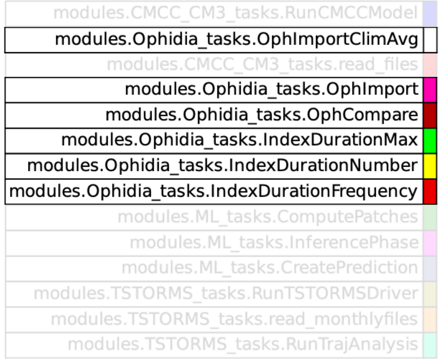


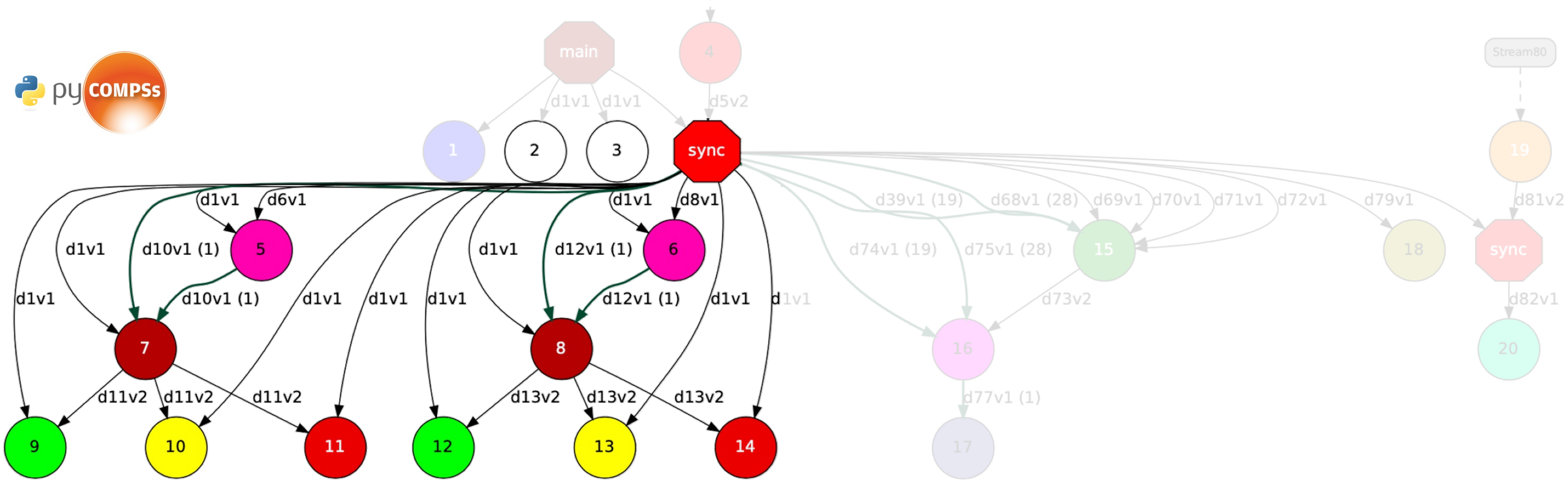


Feature extraction workflow

Based on PyCOMPSs

Analytics

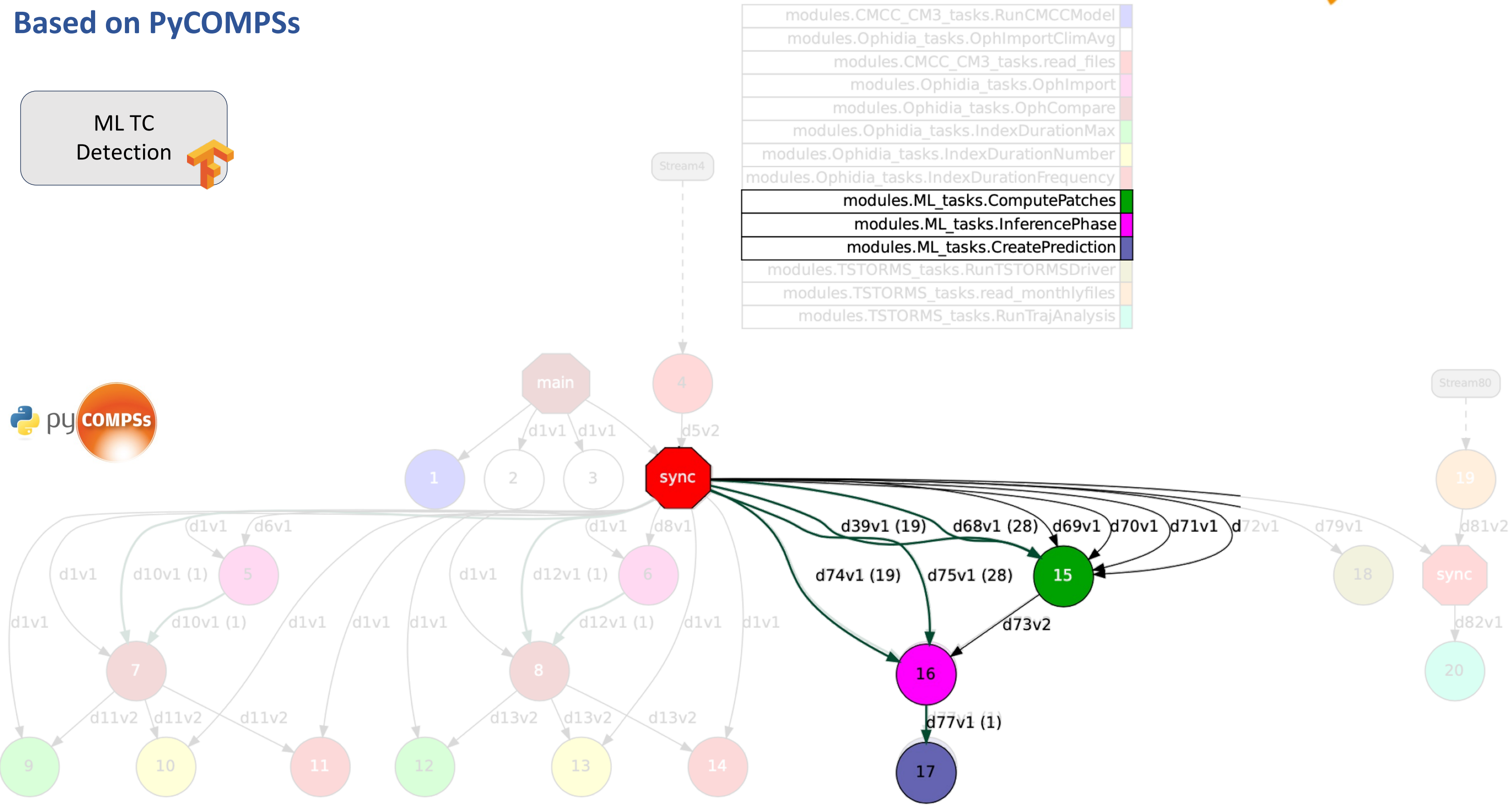




Feature extraction workflow

Based on PyCOMPSs

ML TC
Detection

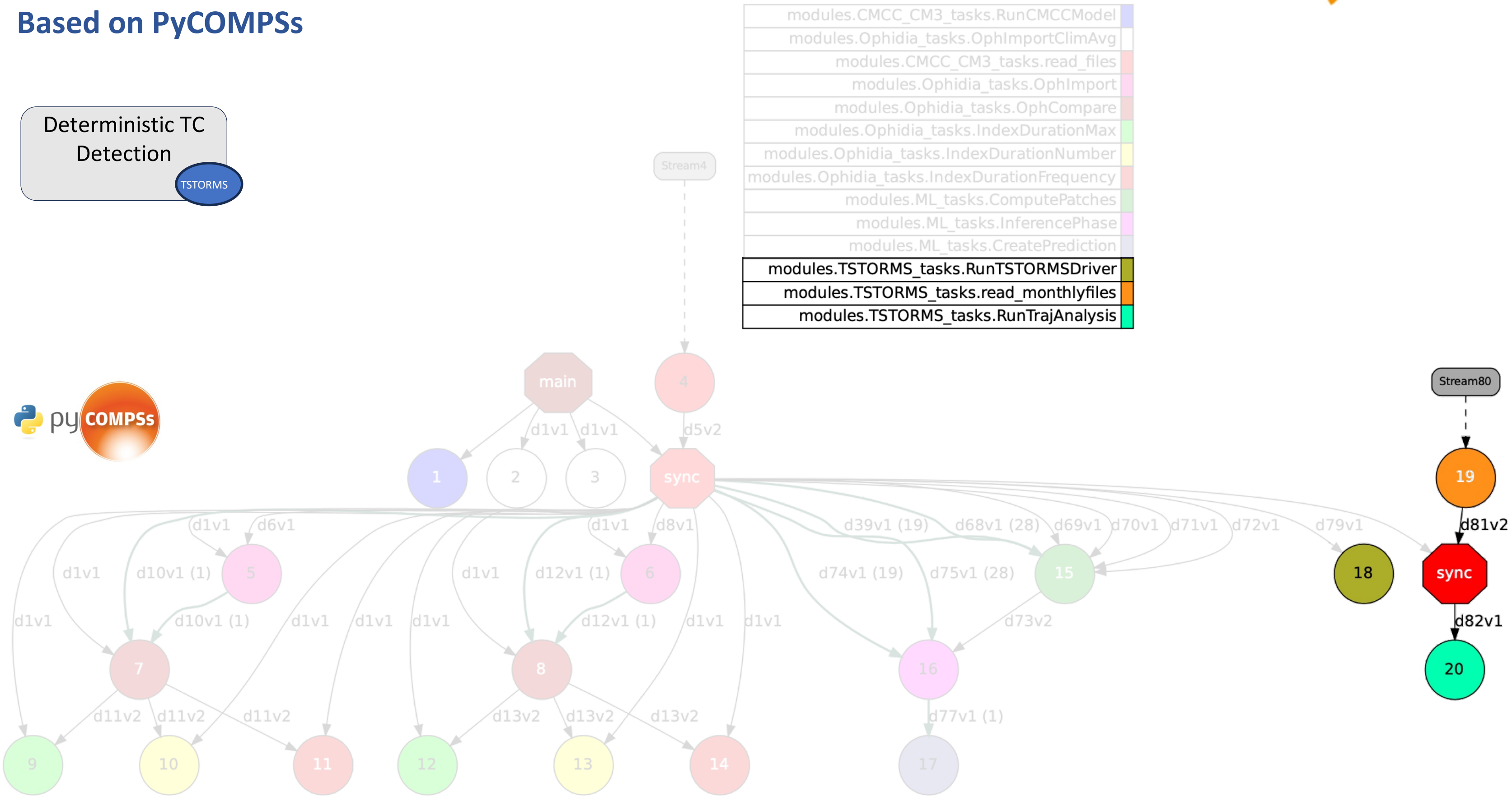


Feature extraction workflow

Based on PyCOMPSs

Deterministic TC Detection

TSTORMS



Conclusions

We exploited the **eflows4HPC software stack** (A4C, YORC, PyCOMPSs, Ophidia, DLS) for the management and run of the entire workflow.

A4C + YORC allows the remote execution of the workflow on the remote HPC cluster Zeus at CMCC.

PyCOMPSs manages the entire tasks flow.

Ophidia allows HPDA features executing data analytics tasks on HPC architectures.

A novel **ML** approach has been developed for the detection of TC eyes.

DLS allows data stage-in and stage-out.

Final refinements should be still carried out.

THANKS

eFlows4HPC

Enabling dynamic and Intelligent workflows in the future EuroHPC ecosystem

www.eFlows4HPC.eu

@eFlows4HPC eFlows4HPC Project

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955558. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Germany, France, Italy, Poland, Switzerland, Norway.