uc3m | Universidad Carlos III de Madrid

ARCOS

Implementación de aplicaciones de eSalud con el modelo de programación DCEx

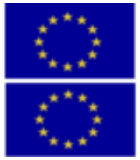
Javier García-Blas, Javier Fernández y Jesús Carretero

Universidad Carlos III de Madrid

fjblas@inf.uc3m.es

Comunidad de Madrid

UNIÓN EUROPEA
Fondos Estructurales
Invertimos en su futuro
UNIÓN EUROPEA
Fondo Social Europeo
El Fondo Social Europeo invierte en tu futuro

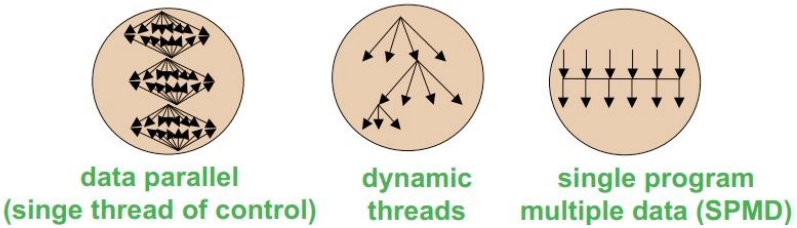


CABAHLA-CM: ConvergenciA Big dAta-Hpc: de Los sensores a las Aplicaciones

DCEx Programming Model

- In the DCEx model **data is the fundamental artifact** and parallelism is based on concurrent data processing:
  - **Data Parallelism**
  - **Task parallelism (data–driven)**
  - **SPMD parallelism**
- **Data-parallel blocks**
  - Data objects for in-memory managing of

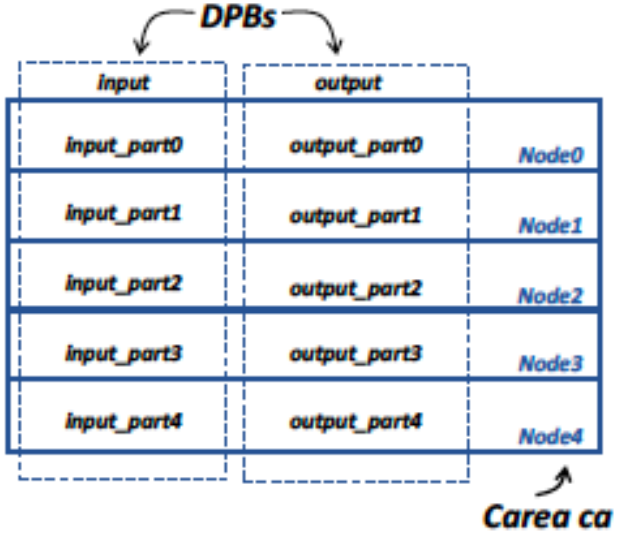
    files, records, databases, streams, tables, text lines, chunks, …



DCEx Programming Model

Data block on computing areas

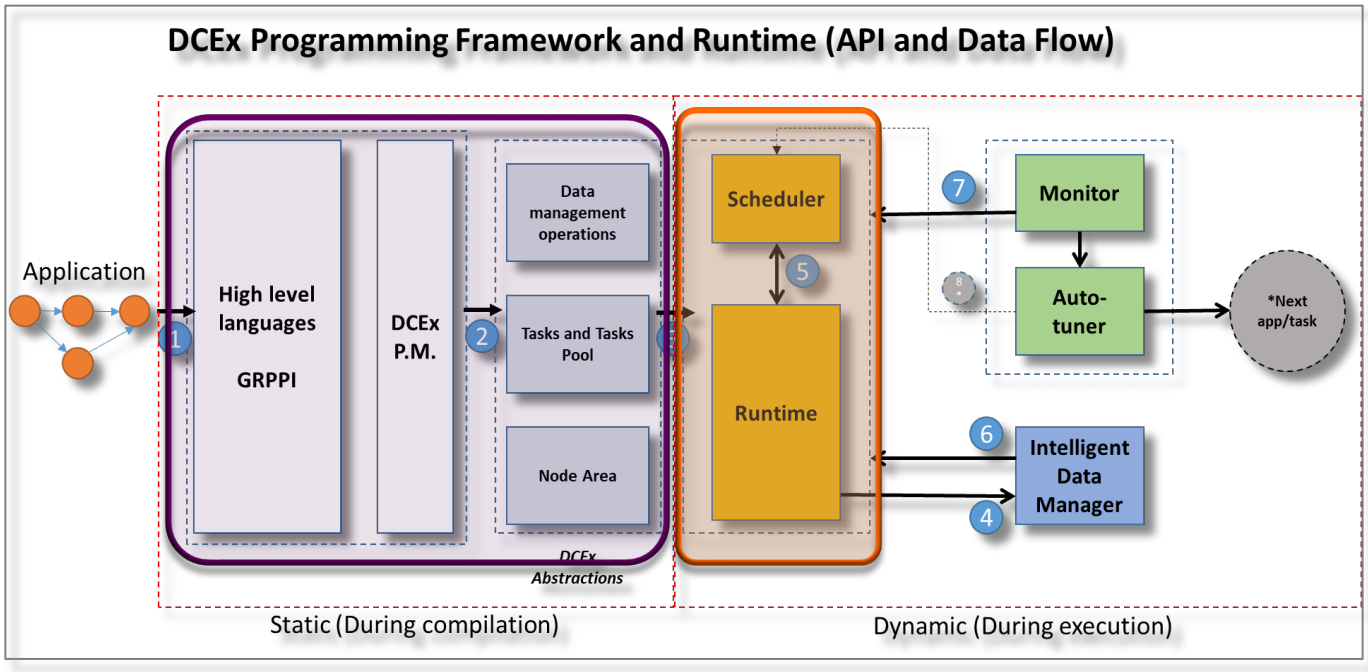
- Basic **data-parallel blocks** operations (**`data.get`**, **`data.set`**) defined for input and output operations from/to secondary storage may use **`Cnode`** and **`Carea`** concepts to map data to nodes.



Example:

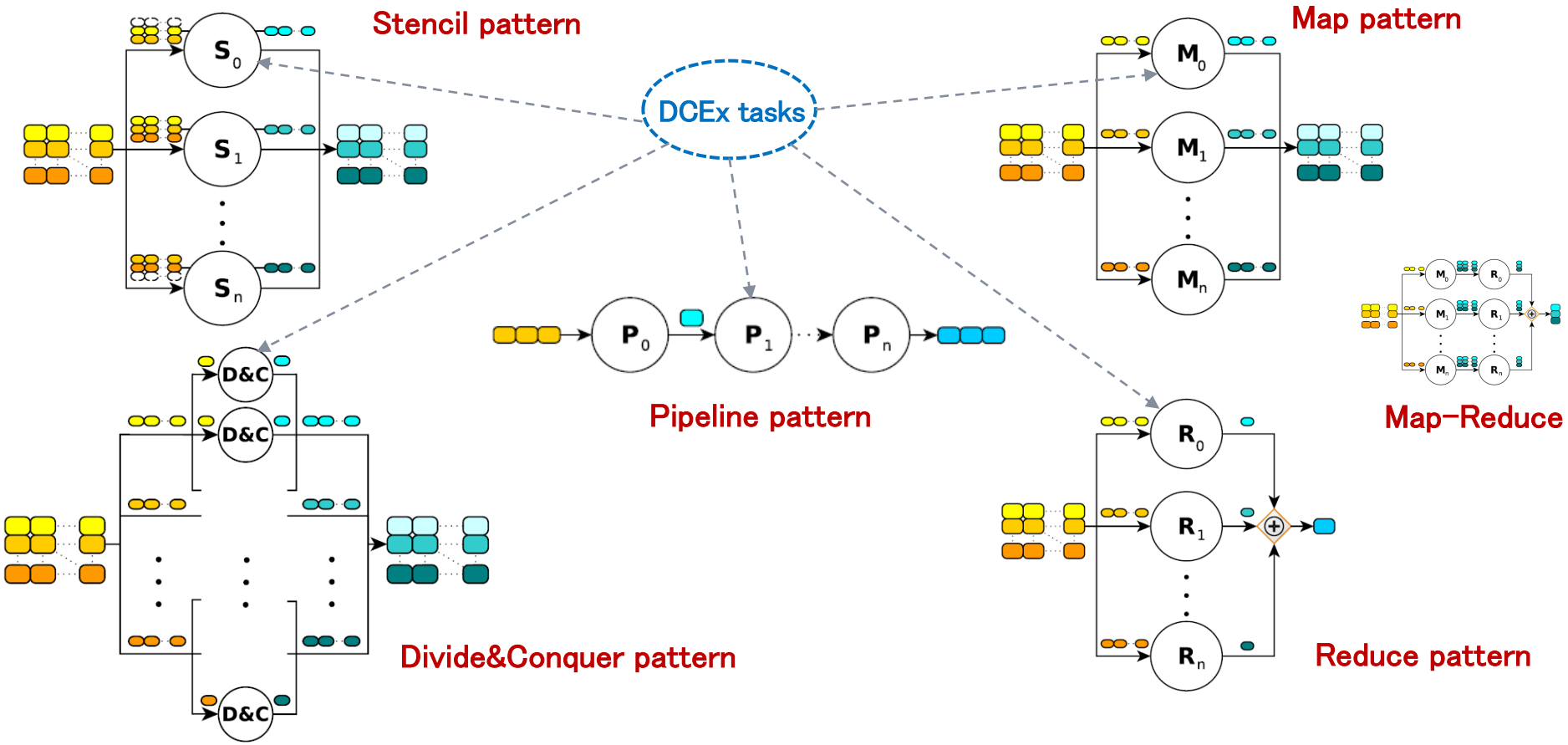
```
node_group = Carea(5,1);
d = data.get(input,[format], … ) at node_group;
```

DCEx Programming Model

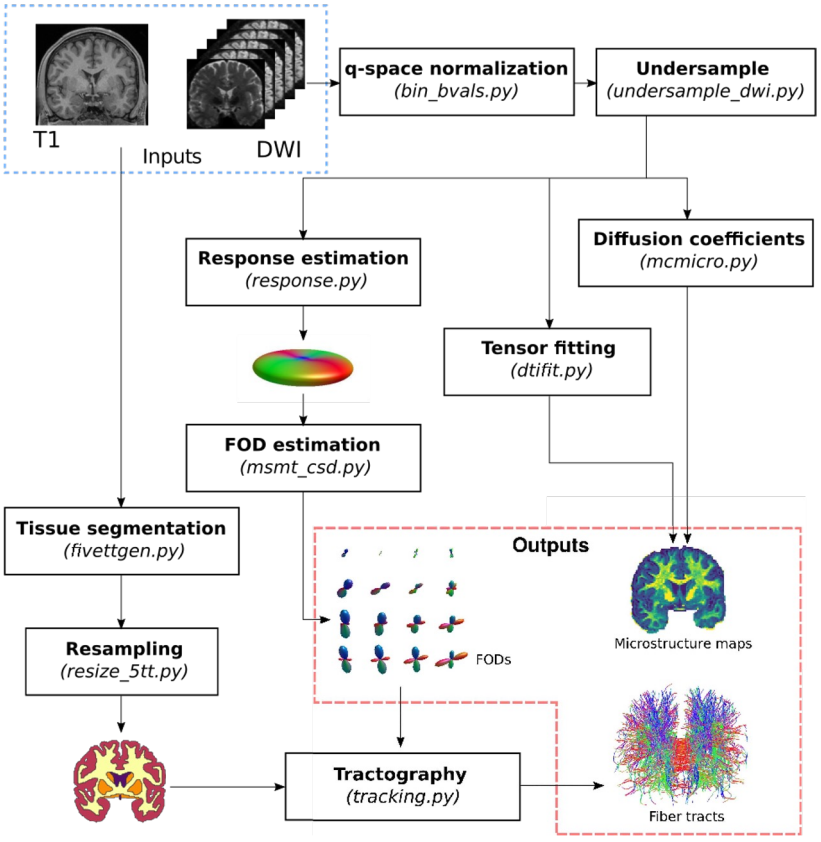


Implementing Big Data applications with DCEx programming model. *Javier Garcia-Blas, Javier Fernandez, Jesus Carretero, Fabrizio Marozzo, Domenico Talia, Alberto Fernandez-Pena, Daniel Martín de Blas. IEEE 34th International Symposium on Computer Architecture and High Performance Computing. 2022. Sent for submission.*

Contribution: GrPPI + DCEx

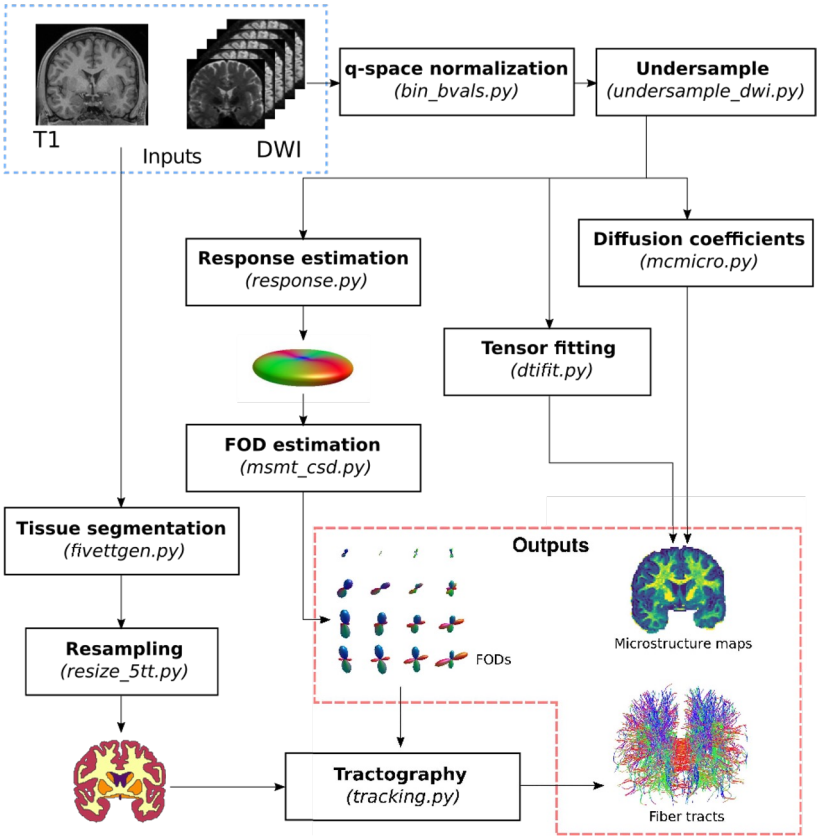


DWI processing workflow



- Processing workflow to study white-matter microstructure and structural connectivity
- Heterogeneous workflow
  - Processing step from different neuroimaging toolboxes
  - Some processes allow for multithreading
- Parallelization
  - Per subject parallelization
  - Some processes can be executed concurrently
  - Some processes allow for per voxel parallelization

DWI processing workflow



- Input data: from HCP (~1000 subjects)

Input file	File format	Dimensions	Size
T1w structural image	NIfTI-1	260 × 311 × 260	84.1 MB
T2w structural image	NIfTI-1	260 × 311 × 260	84.1 MB
Diffusion weighted image	NIfTI-1	145 × 174 × 145 × 288	4.2 GB
Mask image	NIfTI-1	145 × 174 × 145	14.4 MB
b-vectors file	Text file	3 × 288	9.5 kB
b-values file	Text file	1 × 288	1.3 kB

- Output data:

Data avaliable in:
https://db.humanconnectome.org/

  - ~ 1.0 Gb per subject
    - Tissue segmentation (NIfTI)
    - White-matter microstructure maps (NIfTI)
    - Fiber orientation distribution maps (NIfTI)
    - Tractography streamlines (binary file)

Example of output data available in:
https://doi.org/10.5281/zenodo.3662302

DCEx integration

- Baseline application implemented using **nipype** python package:
  - Execution plugins: Multiproc, SGE/SLURM,...
- Workflow implemented using the ASPIDE ***DCEx***
  - task: execution of external code (python scripts)
  - farm, split_join, and pipeline patterns
- ***Containers***

```
auto resize_5tt = [](std::string path) {
        auto in  = path + "/5tt.nii.gz";
        auto out = path + "/resized_5tt.nii.gz";
        return "python2.7 resize_5tt.py " + in + " " + out;
};
```

Example of declaration of the call to the Python script.

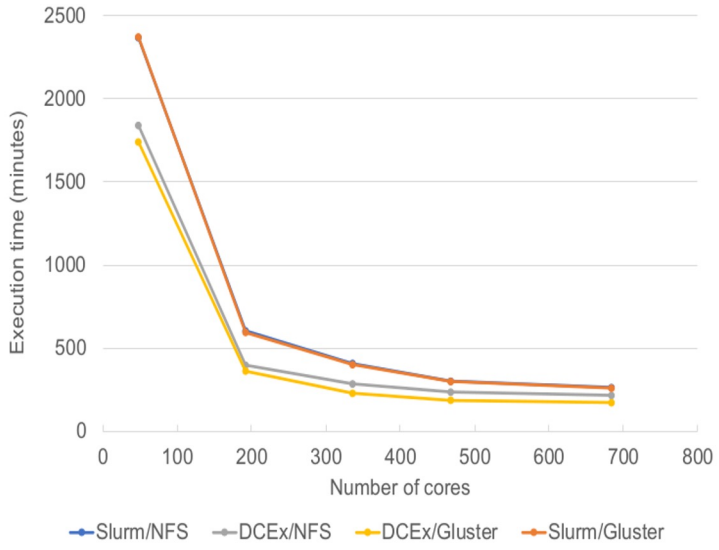
```
auto inner_split = split_join(duplicate{},
    farm(task(response)),
    farm(task(true,dtifit)),
    farm(task(true,mcmicro))
);

pipeline(exec,
    all(folders),
    task([](string path){
        return "mkdir -p " + path + WORKING;
    }),
    split_join(duplicate{},
        pipeline(
            farm(task(true,fivettgen)),
            farm(task(resize_5tt))
        ),
        pipeline(
            farm(task(bin_bvals)),
            farm(task(undersample_dwi)),
            inner_split,
            farm(task(true,msmt_csd))
        )
    ),
    farm(task(true,tracking)),
    [](string & path)  {
        cerr <<  "Finished " + path  << endl;
    }
);
```

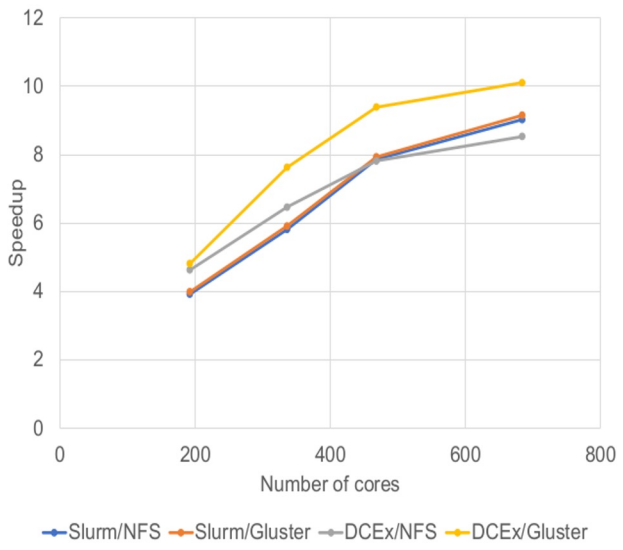
Code avaliable in:
https://gitlab.arcos.inf.uc3m.es/aspide/dcex/tree/master/usecases/sermas

Evaluation of results

- Tests:
  - Cluster of 28 nodes (total 684 cores)
  - NFS and Gluster file systems



(a) Overall execution.



(b) Speedup.

Comparison of Nipype and DCEx execution for the two file systems. We used 32 random subjects from the HCP-YA database

uc3m | Universidad Carlos III de Madrid

ARCOS

Implementación de aplicaciones de eSalud con el modelo de programación DCEx

Javier García-Blas, Javier Fernández y Jesús Carretero

Universidad Carlos III de Madrid

fjblas@inf.uc3m.es

CABAHLA

CABAHLA-CM: ConvergenciA Big dAta-Hpc: de Los sensores a las Aplicaciones