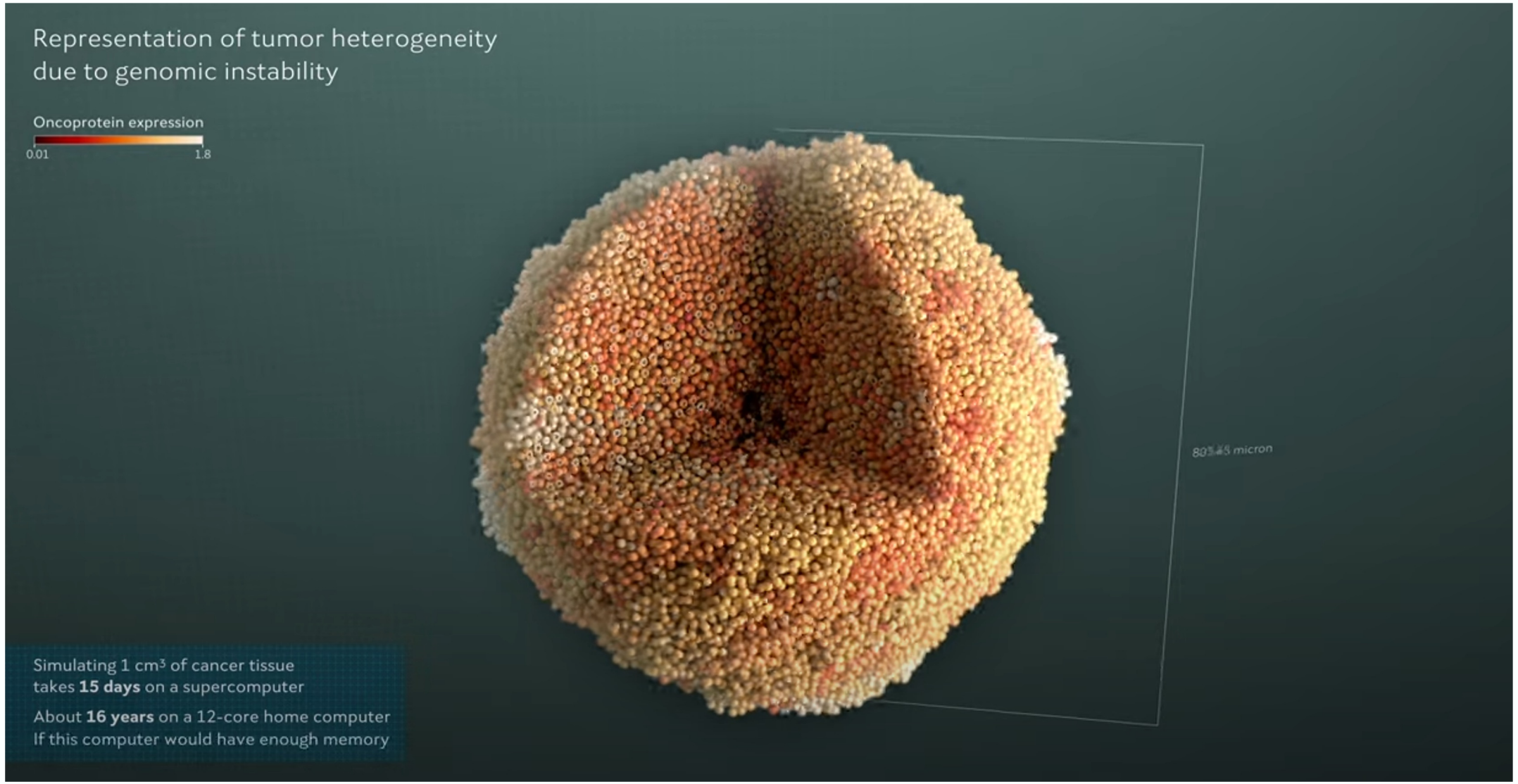
HPC/Exascale Centre of Excellence in Personalised Medicine

Practical session with PhysiCell

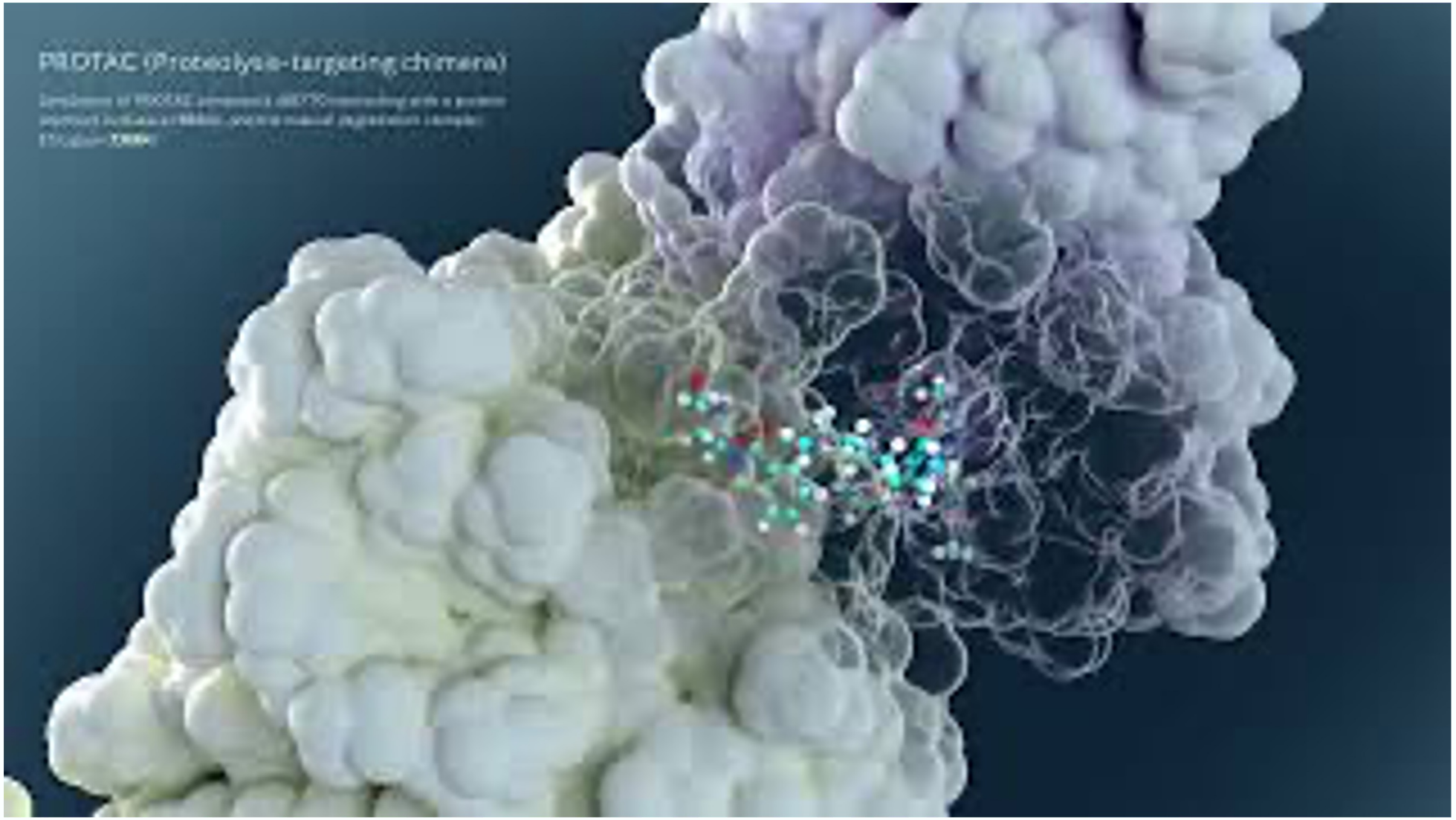
José Carbonell Caballero
Thalia Diniaco

The PerMedCoE project has received funding from the European Union's Horizon 2020 research and innovation programme under the grant agreement N°951773

Simulating a cellular microenvironment

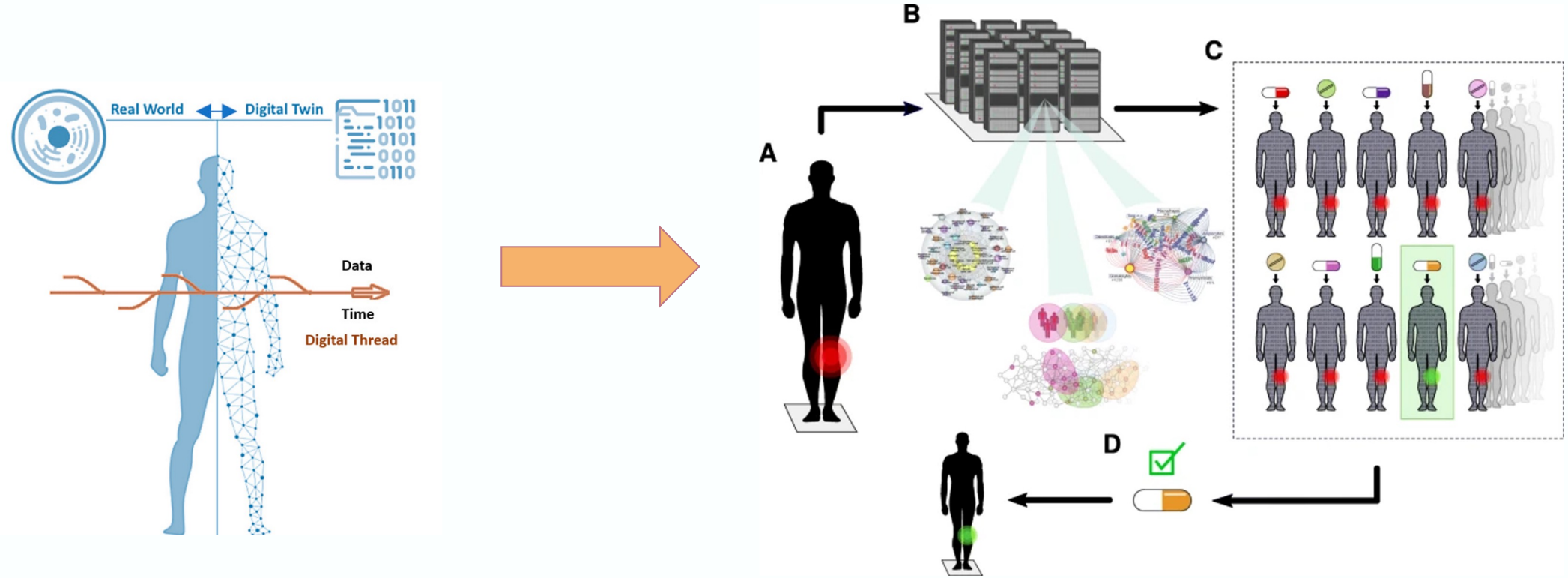


Simulating a cellular microenvironment



Personalised Medicine and Digital Twins

Towards patient-specific treatments

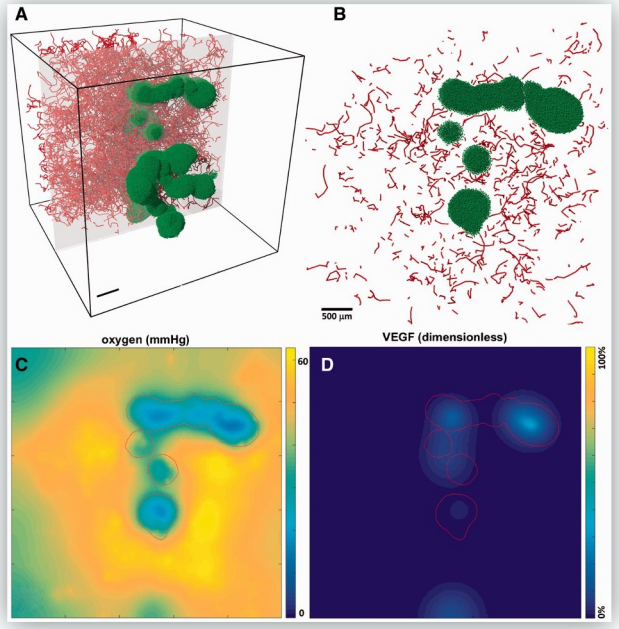


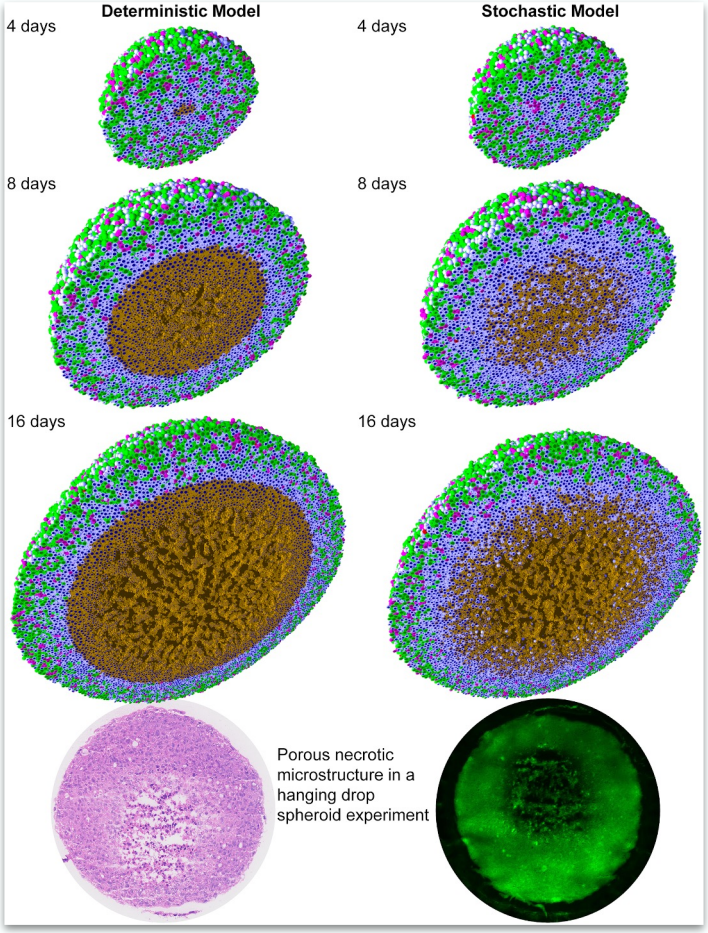
Björnsson, B., Borrebaeck, C., Elander, N. et al. Digital twins to personalize medicine. Genome Med 12, 4 (2020).

Core tools

PhysiCell

- Physics-based virtual microscope
- Implements Agent-Based programming
- Cell phenotype and cell-cell interactions
- Diffusion transport solver

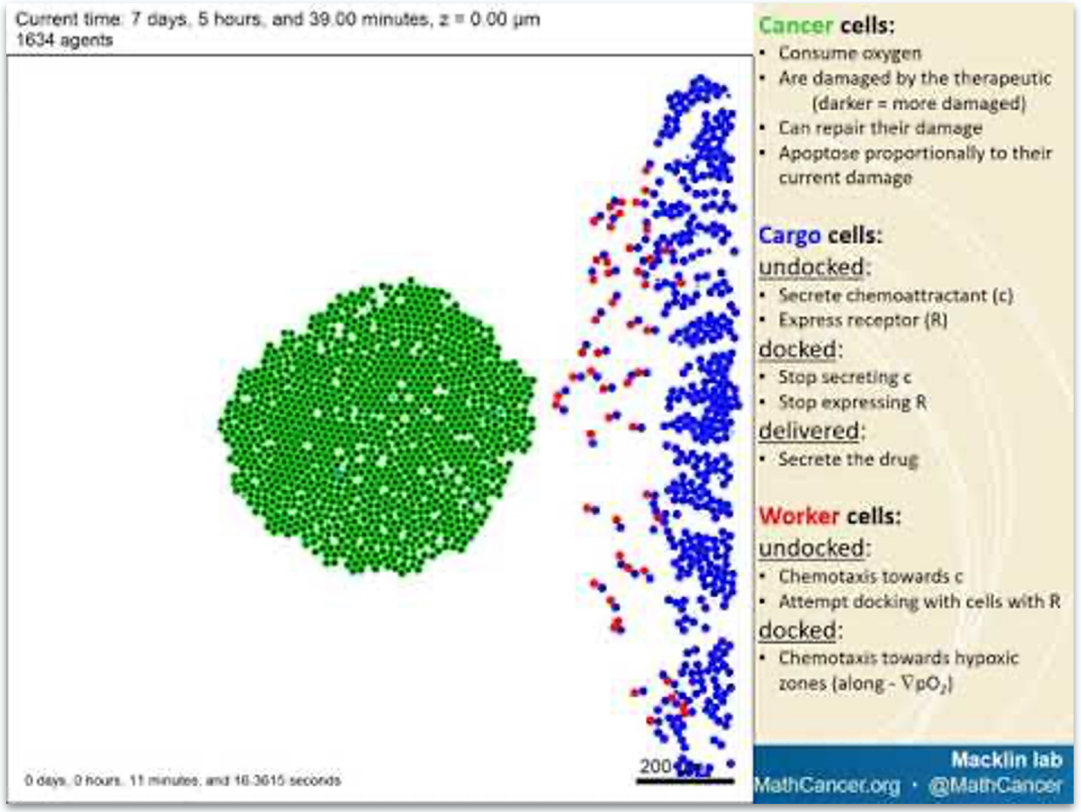




Ghaffarizadeh, A. et al (2018). https://doi.org/10.1371/journal.pcbi.1005991

PhysiCell

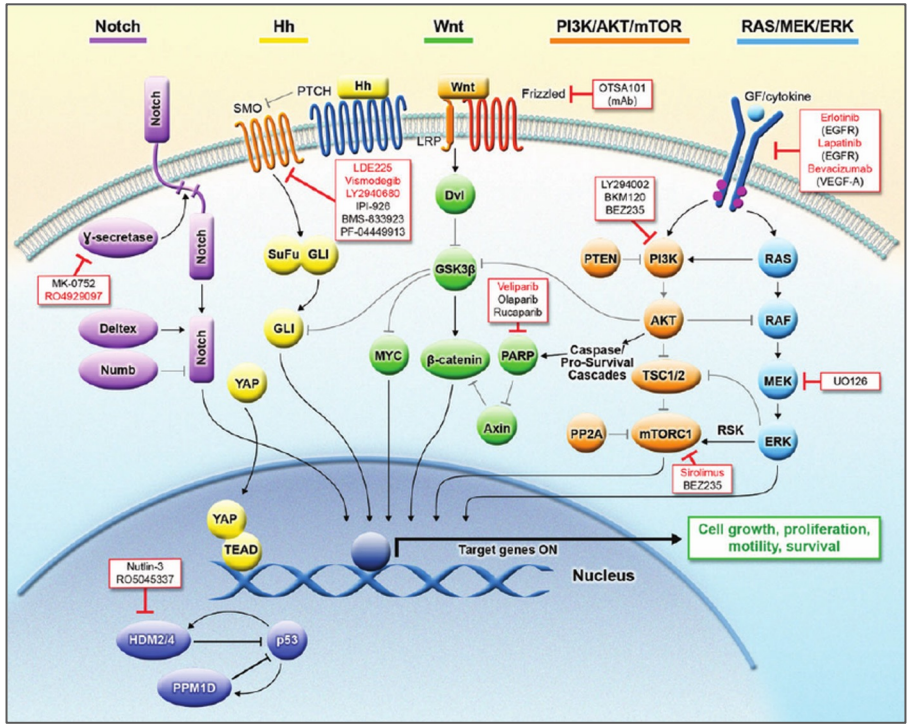
The framework recapitulates main cellular features



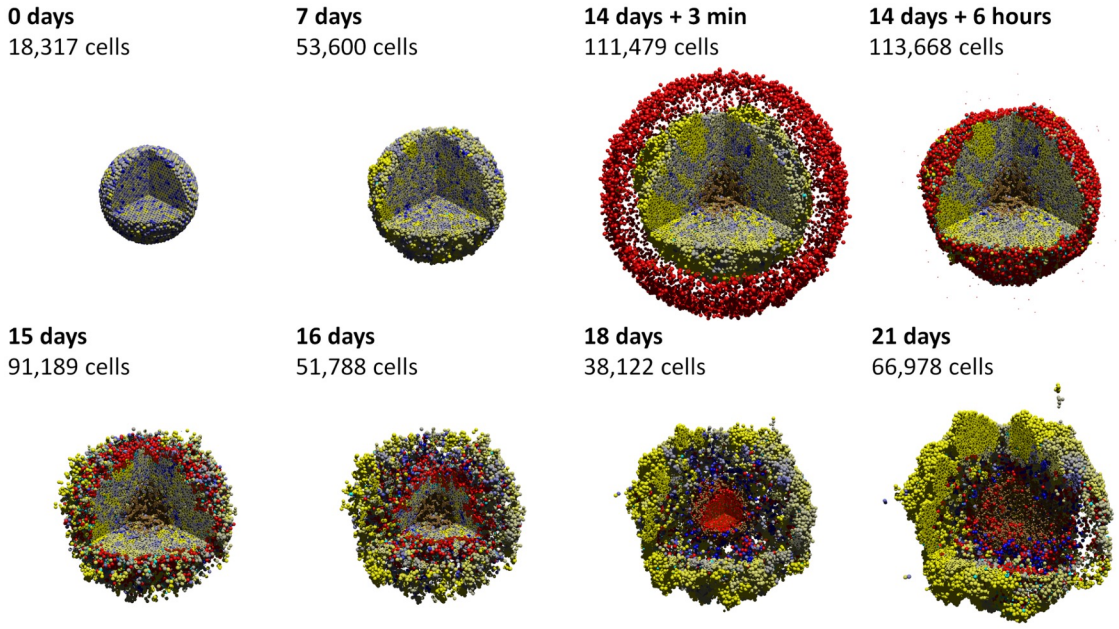
- Cell motility (and chemotaxis)
- Cell cycle phases (GO/G1, S, G2, M)
- Cell death (apoptosis and necrosis)
- Cell volume (nucleus and cytoplasm)
- Cell growth
- Substrate production/consumption
- Physical interaction between cells
- Phagocytosis

Extending PhysiCell

Introducing molecular pathways into multiscale agents



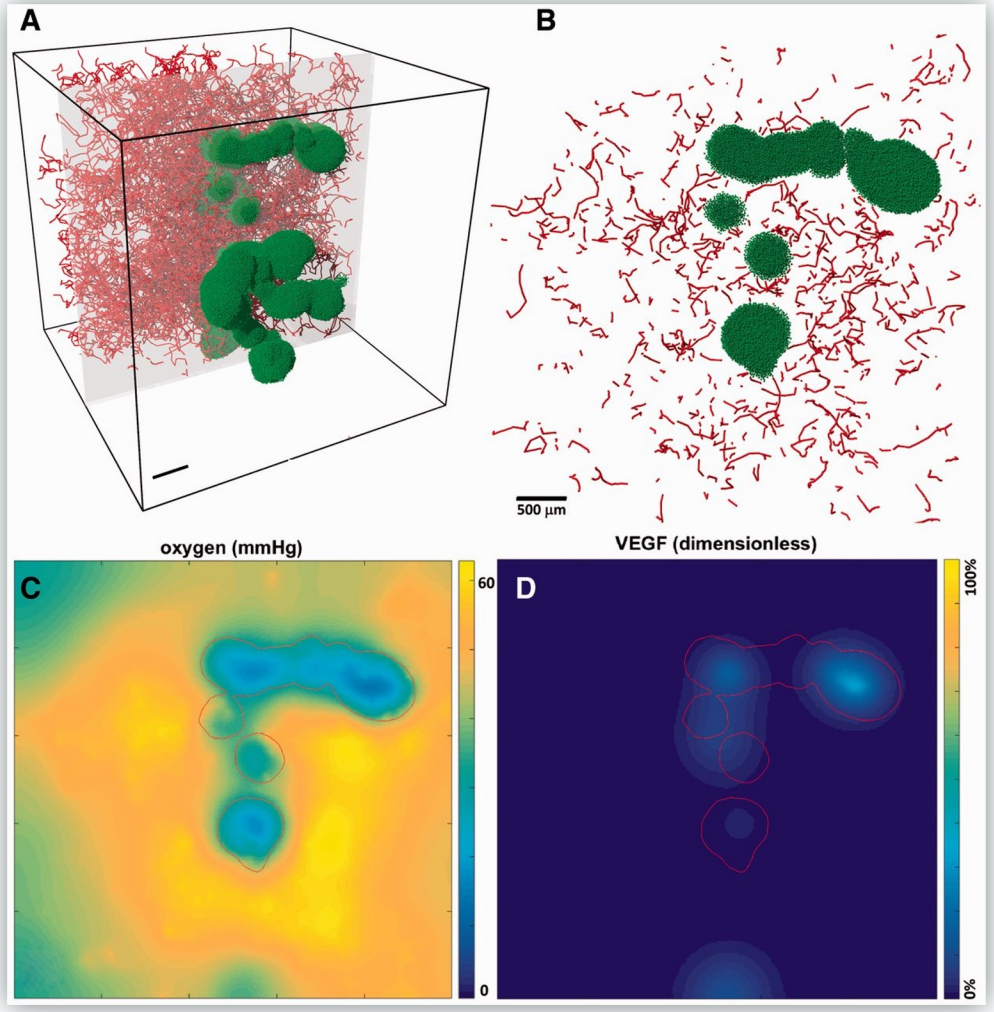
Tobey J et al..Neuro-Oncology, Volume 16, Issue 1, (2013)



Ghaffarizadeh A et al. PLoS Comput Biol. 2018

PhysiCell

BioFVM (diffusive transport solver)



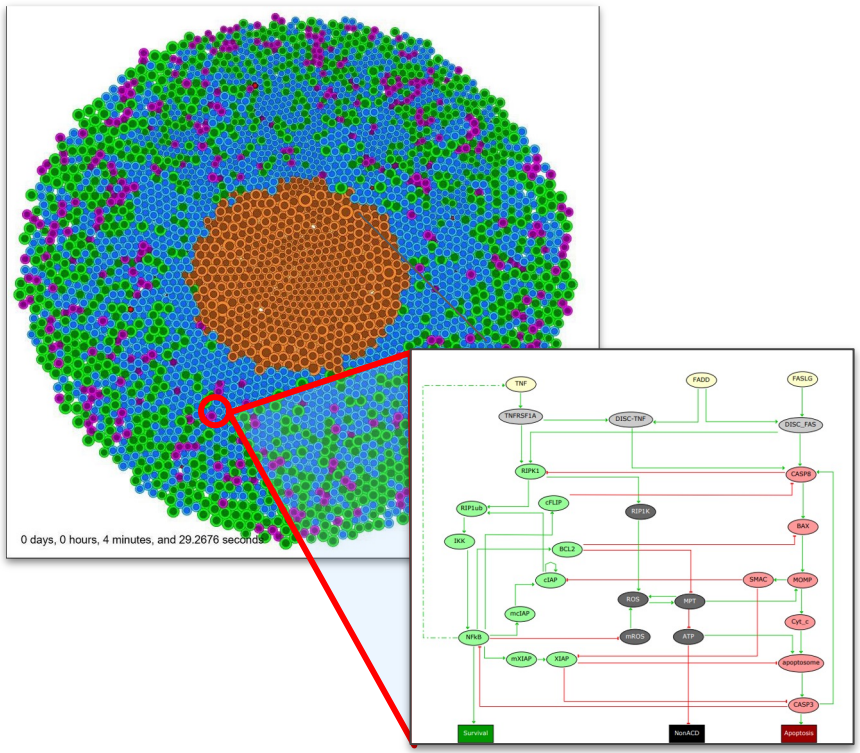
$$\frac{\partial \vec{\rho}}{\partial t} = \overbrace{\vec{D}\nabla^2\vec{\rho}}^{\text{diffusion}} - \overbrace{\vec{\lambda}\vec{\rho}}^{\text{decay}} + \overbrace{\vec{S}(\vec{\rho}^* - \vec{\rho})}^{\text{bulk source}} - \overbrace{\vec{U}\vec{\rho}}^{\text{bulk uptake}}$$

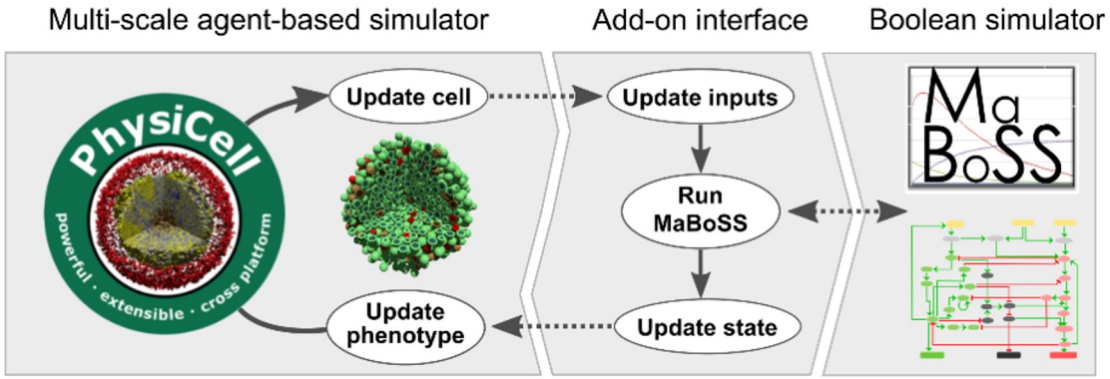
$$+ \overbrace{\sum_{\text{cells}k} 1_k(\vec{x}) \left[\vec{S}_k(\vec{\rho}_k^* - \vec{\rho}) - \vec{U}_k\vec{\rho}\right]}^{\text{sources and uptake by cells}} \text{ in } \Omega$$

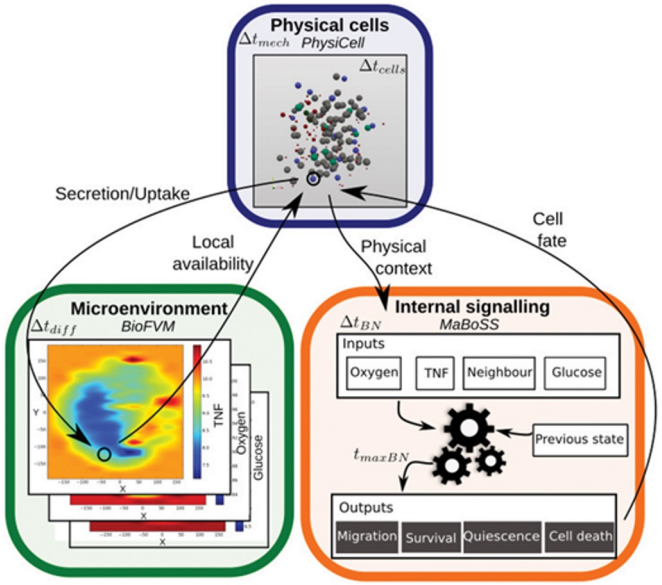
Extending PhysiCell

PhysiBoss (PhysiCell + MaBoSS)

Cell state and decisions depend on embedded boolean models





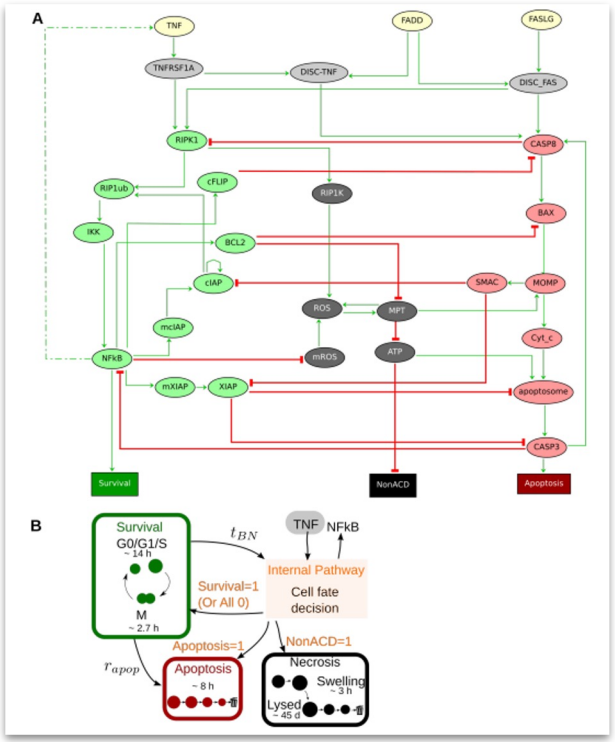


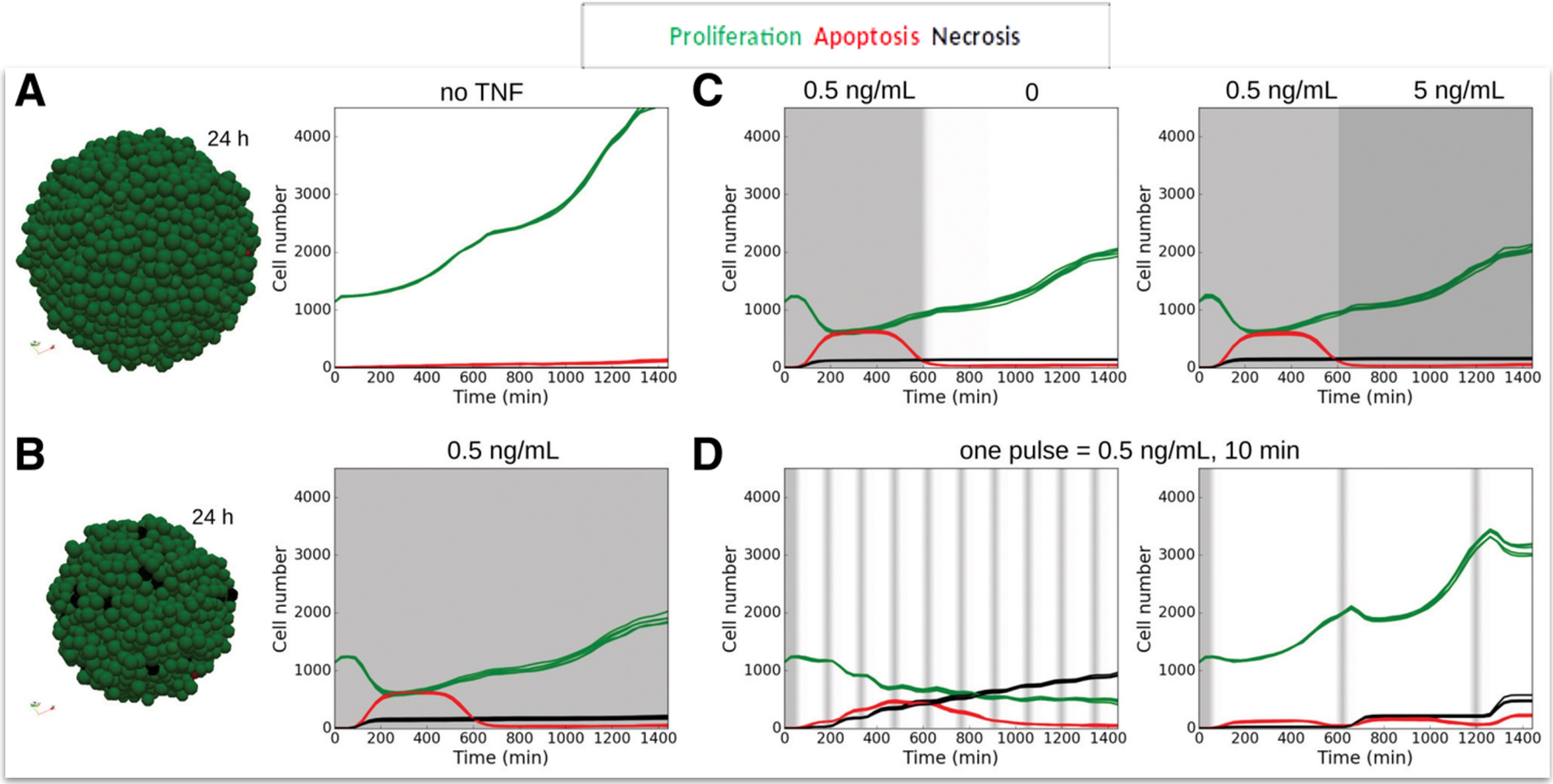
Letort, G., Montagud, A. et al (2019). Bioinformatics, 35(7), 1188–1196. https://doi.org/10.1093/bioinformatics/bty766

Extending PhysiCell

Agent-based + probabilistic logical models (PhysiCell + MaBoSS)

PhysiBoSS

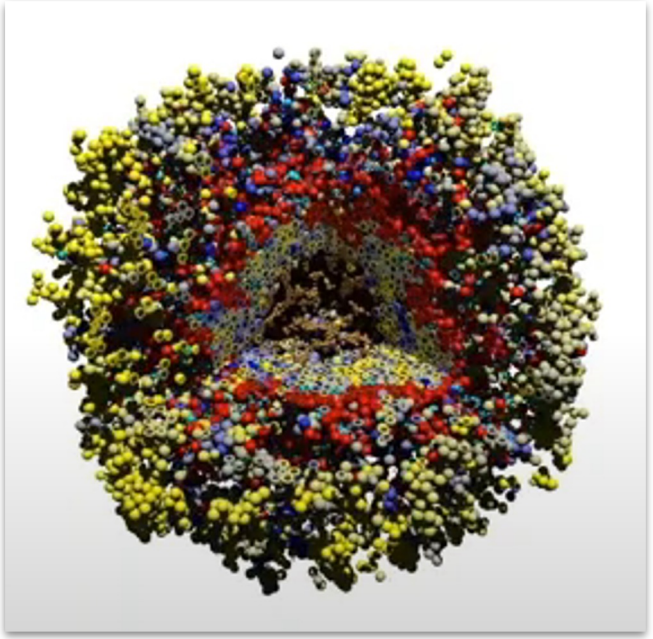




Per Med CoE

Letort, G., Montagud, A., et al. (2019). Bioinformatics, 35(7), 1188–1196

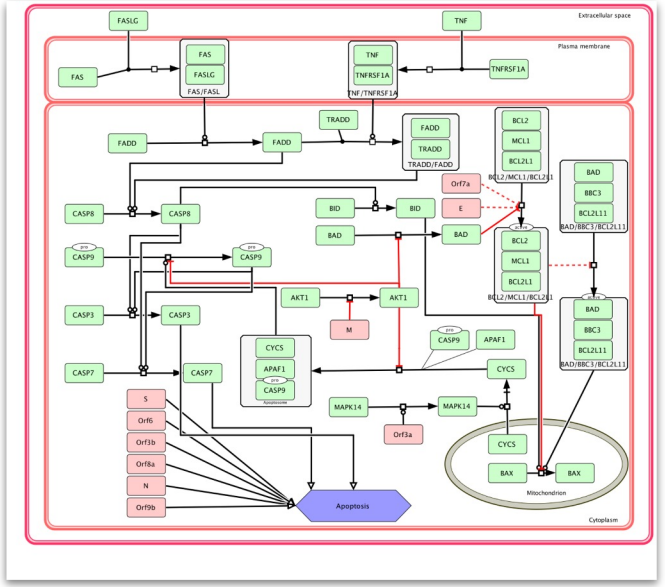
Cell level simulations



Per Med CoE

HPC

Molecular Pathways

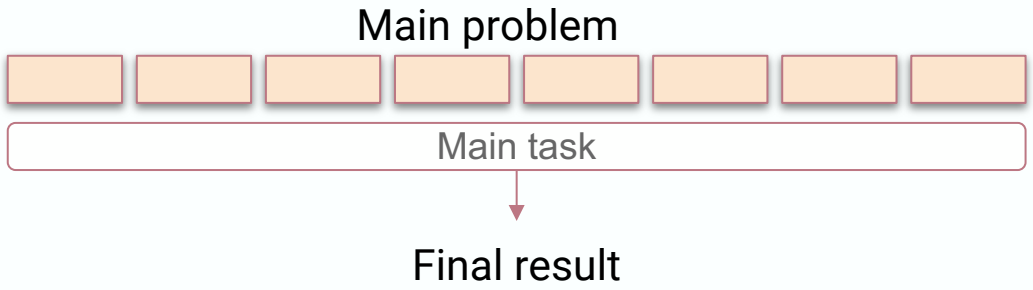


Parallel computing

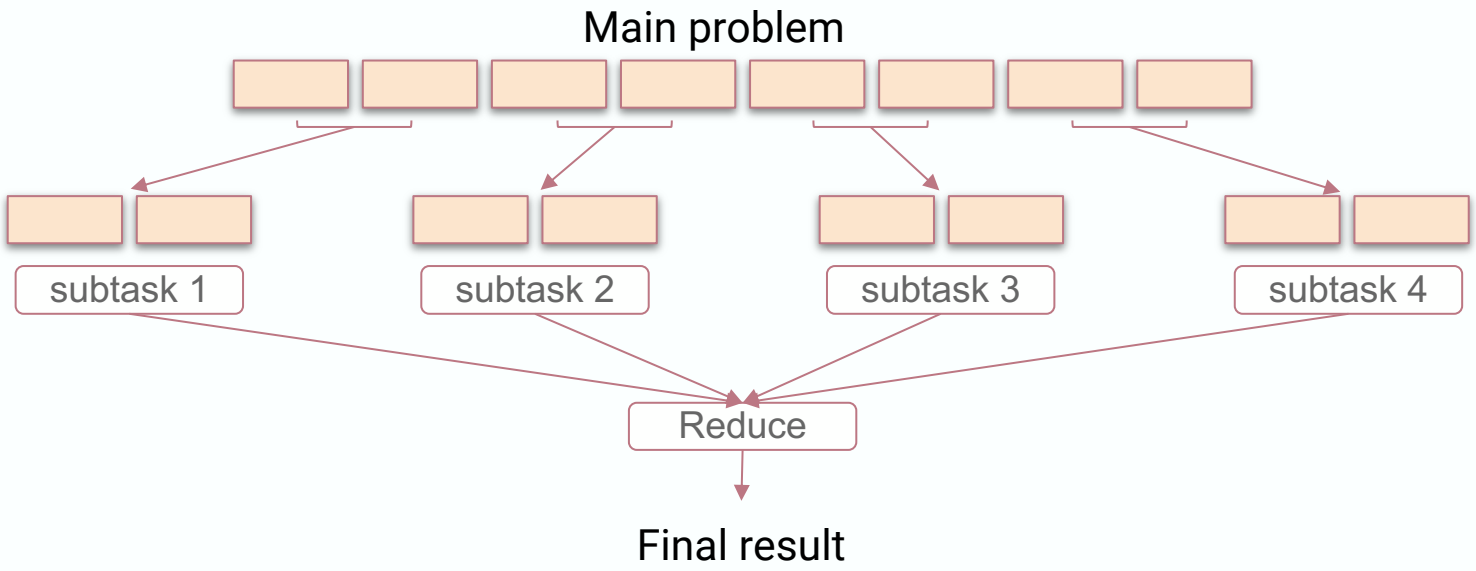
Traditional approaches

- Native support
- Based on logical threads
- POSIX threads (IEEE)

Serial computing

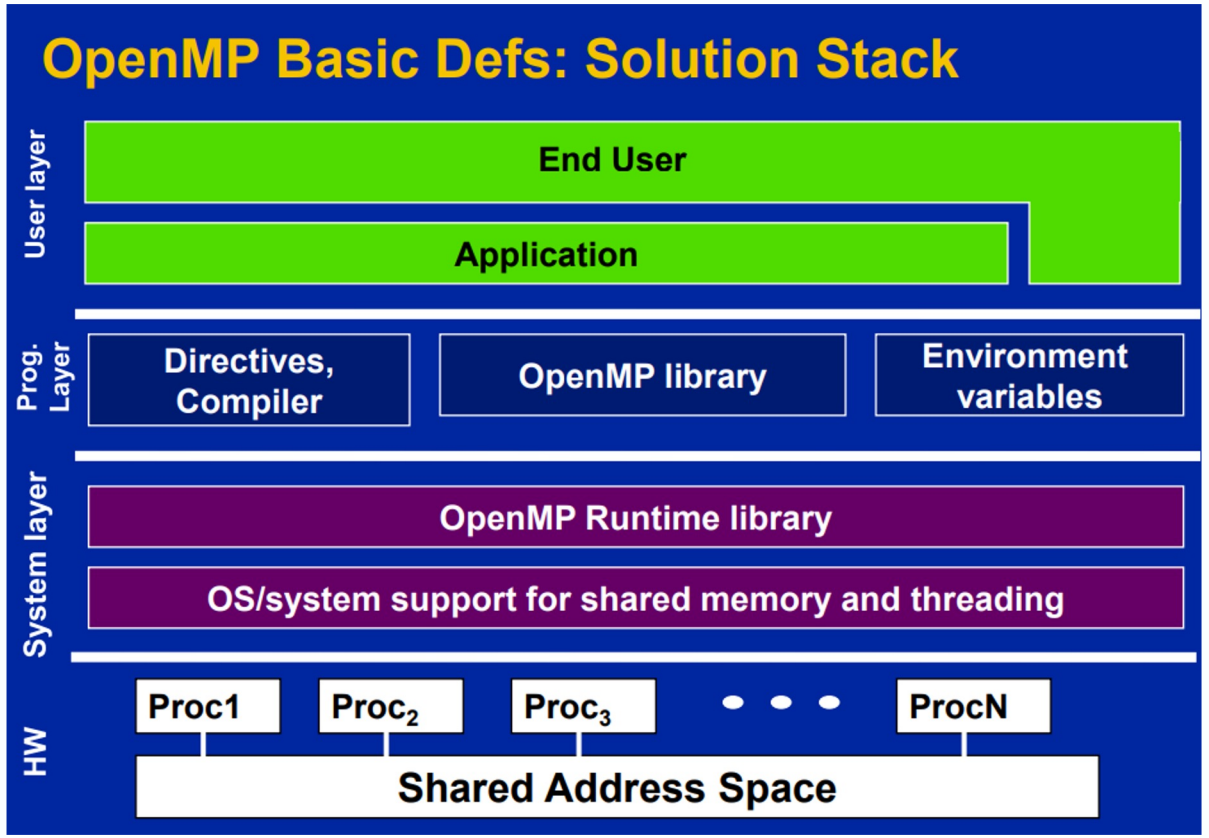


Parallel computing



OpenMP

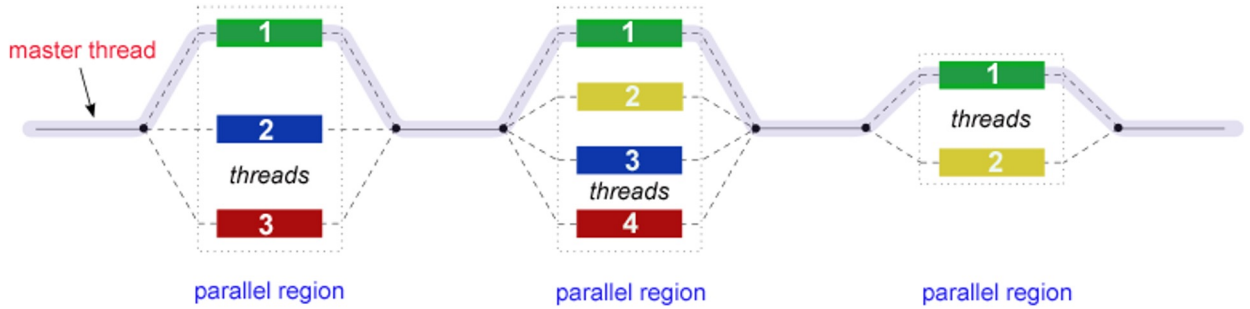
- Directive-based
- Simple and flexible interface for developing parallel application
- API for direct multi-threaded, shared memory parallelism
- Comprised of three primary API components:
  - Compiler Directives
  - Runtime Library Routines
  - Environment Variables



Tim Mattson

Parallel computing

```
#include <stdio.h>
#include <omp.h>

int main(int argc, char** argv){
    int partial_Sum, total_Sum;

    #pragma omp parallel private(partial_Sum) shared(total_Sum)
    {
        partial_Sum = 0;
        total_Sum = 0;

        #pragma omp for
        {
            for(int i = 1; i <= 1000; i++){
                partial_Sum += i;
            }
        }

        //Create thread safe region.
        #pragma omp critical
        {
                //add each threads partial sum to the total sum
                total_Sum += partial_Sum;
        }
    }
    printf("Total Sum: %d\n", total_Sum);
    return 0;
}
```



Example ploop.1.c

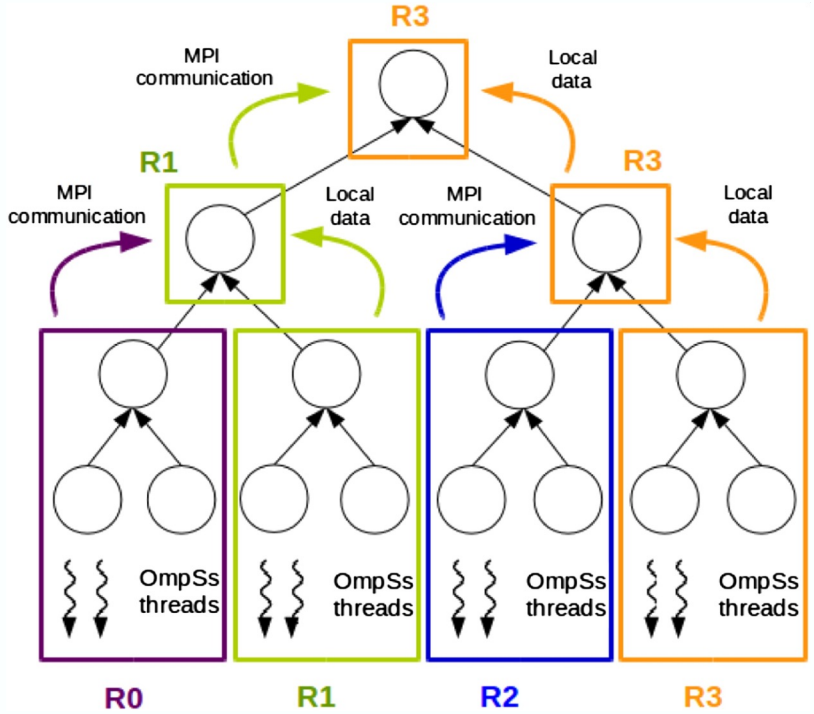
```
S-1    void simple(int n, float *a, float *b)
S-2    {
S-3        int i;
S-4
S-5    #pragma omp parallel for
S-6        for (i=1; i<n; i++) /* i is private by default */
S-7            b[i] = (a[i] + a[i-1]) / 2.0;
S-8    }
```

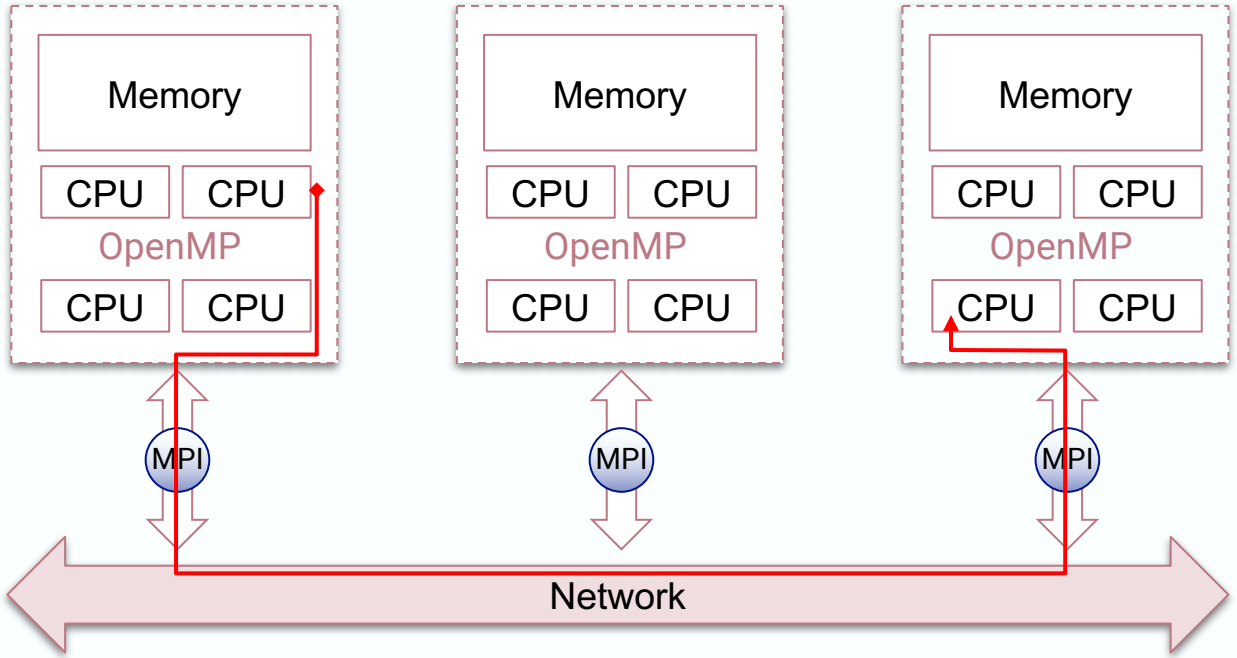
credit by HPC at Lawrence Livermore National Lab

Distributed computing

Message Passing Interface (MPI)

- API for coordinating different nodes to jointly perform a given task
- Communication via the local network (latency is an issue)
- Several nodes work as a single machine
- Different implementations (OpenMPI, Intel MPI)





Distributed computing

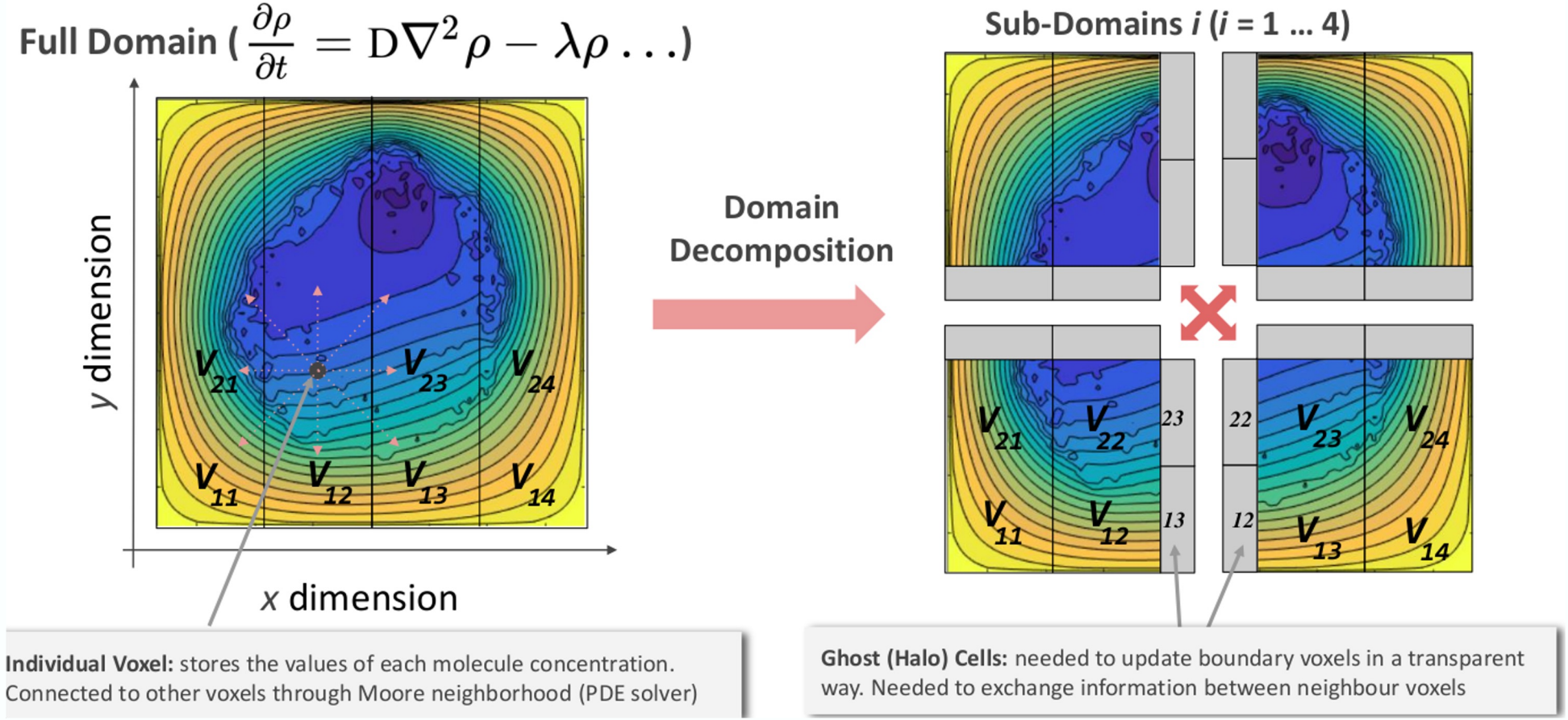
Message Passing Interface (MPI)

```c
#include <stdio.h>
#include <mpi.h>

int main(int argc, char** argv){
    int process_Rank, size_Of_Cluster;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size_Of_Cluster);
    MPI_Comm_rank(MPI_COMM_WORLD, &process_Rank);

    for(int i = 0, i < size_Of_Cluster, i++){
        if(i == process_Rank){
            printf("Hello World from process %d of %d\n", process_Rank, size_Of_Cluster);
        }
        MPI_Barrier(MPI_COMM_WORLD);
    }

    MPI_Finalize();
    return 0;
}
```

https://curc.readthedocs.io/en/latest/index.html

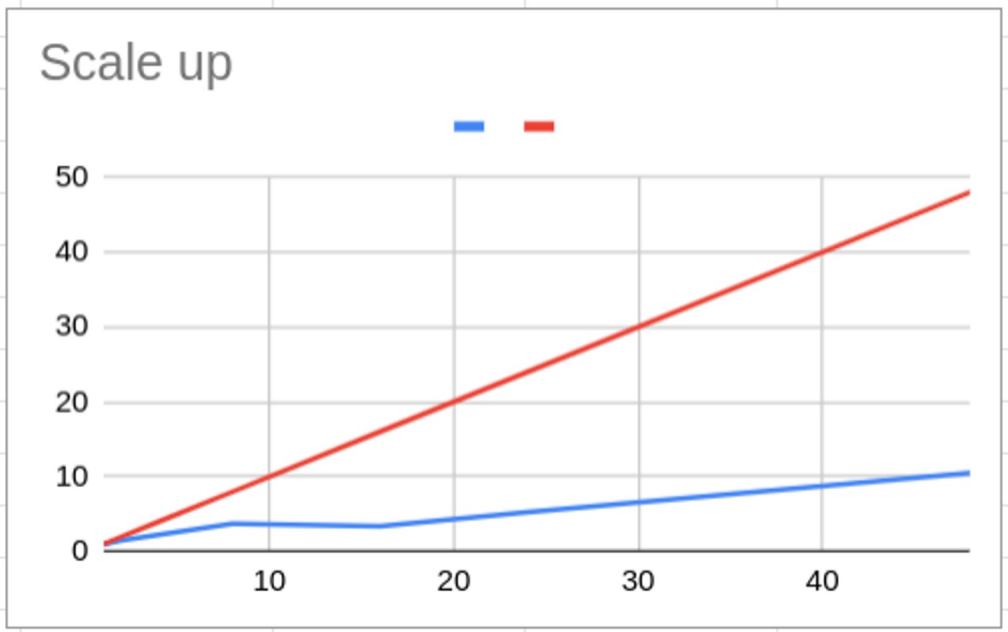
Refactoring strategies in PhysiCell

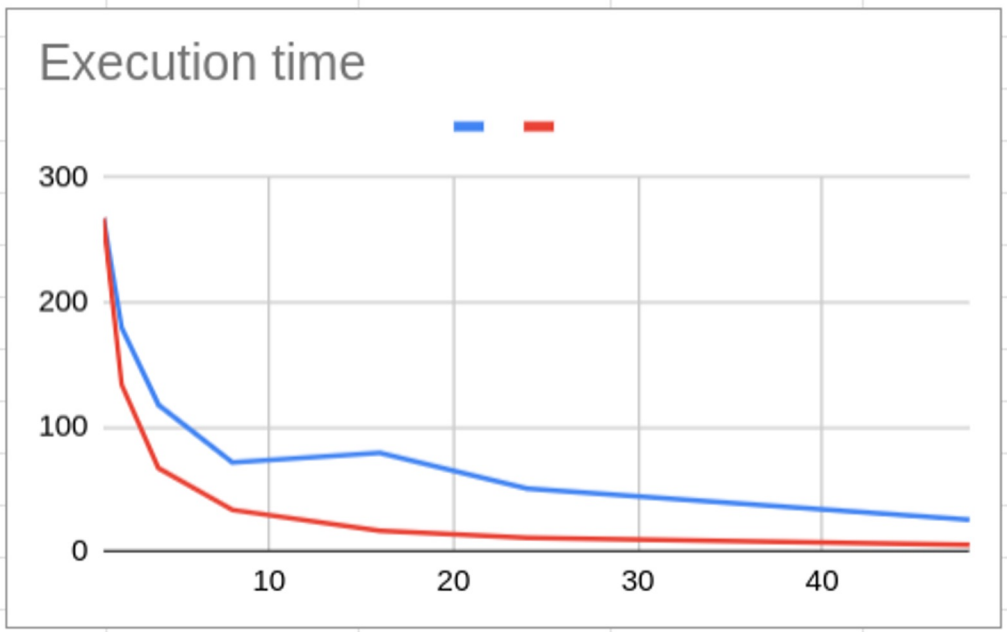
PhysiCell-MPI



Hands-on

		CORES						
		1	2	4	8	16	24	48
NODES	1	266.942	179.2124	117.6434	71.4096	79.2203	50.5011	25.387
	2	123.5726	85.8754	60.0800271	39.0463			
	4	99.2707	77.2388	56.7724				





THANK YOU

HPC/Exascale Centre of Excellence in Personalised Medicine

www.permedcoe.eu

Follow us in social media:

www.linkedin.com/company/permedcoe

@permedcoe

The PerMedCoE project has received funding from the European Union's Horizon 2020 research and innovation programme under the grant agreement N°951773