



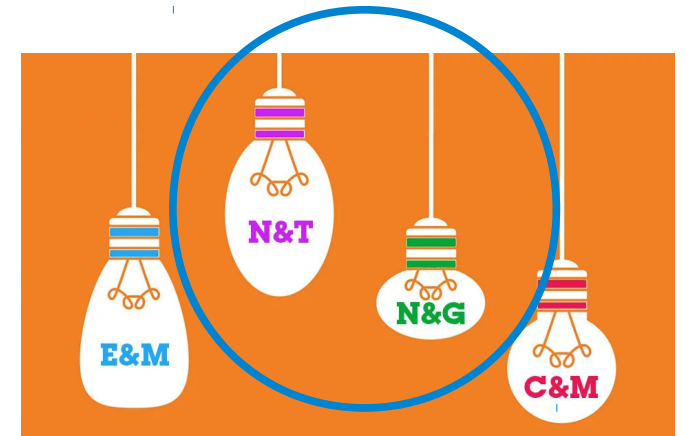
Biomedische Simulaties op CERN

Dutch Teacher's Programme

Ahmad Hesam
26 September 2019

Introductie

- Tweetalig VWO gestudeerd in Apeldoorn
- Natuur & Techniek + Natuur & Gezondheid
- Technische Natuurkunde (Applied Physics)
- 3e jaar: hmm... wil ik dit blijven doen?
- Interesse in Quantum Computers heeft me overtuigd om een Minor in Computer Science / Engineering te doen



Universiteit: Master

- Minor was leuk: ik doe de master!
- High-Performance Computing is de (nabije) toekomst
- Delftse Bedrijvendagen: CERN kwam langs

└─▶ CERN biedt zomerstages aan voor studenten!

CERN openlab summer programme

- 9 weken in de zomer
- Betaalde stage
- Lezingen over CERN technologieën
- Mogelijkheden om te blijven
- Gemiddeld 40 studenten uit 20 verschillende landen



CERN openlab

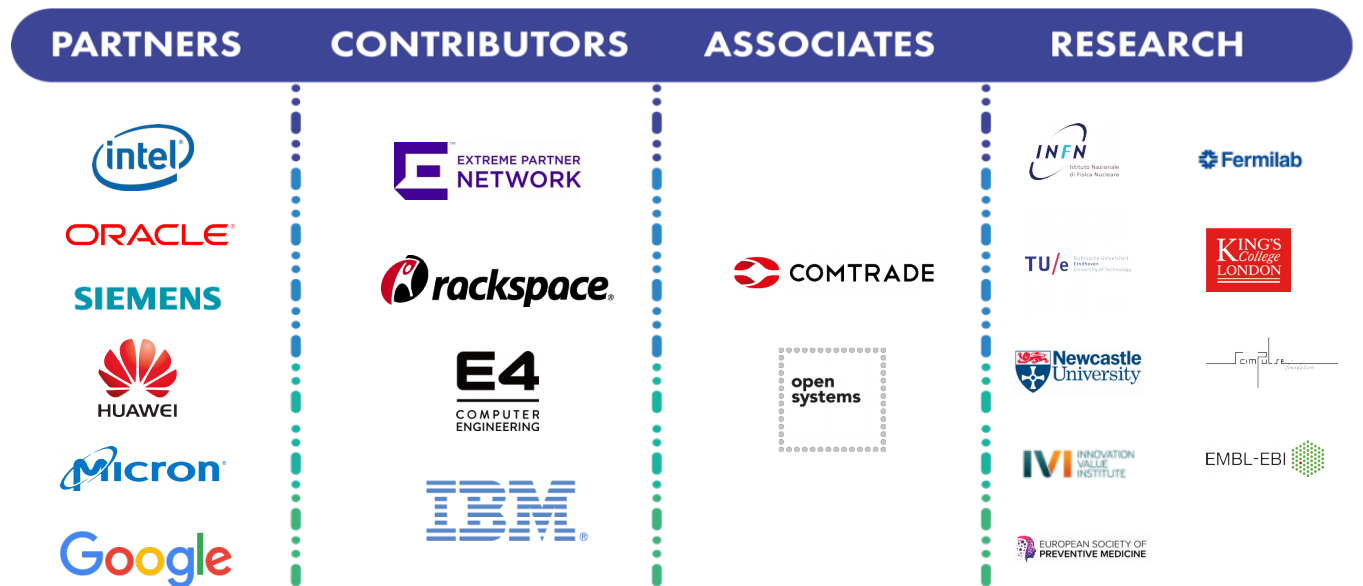
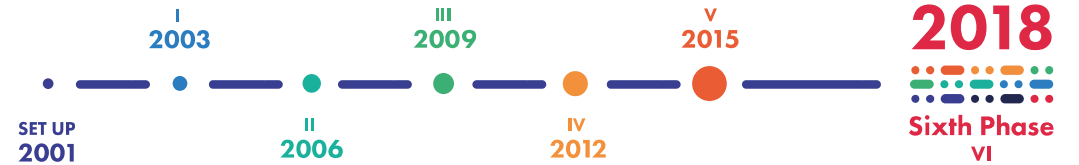
A science – industry partnership to drive R&D and innovation

Evaluate **state-of-the-art technologies** in a challenging environment and improve them

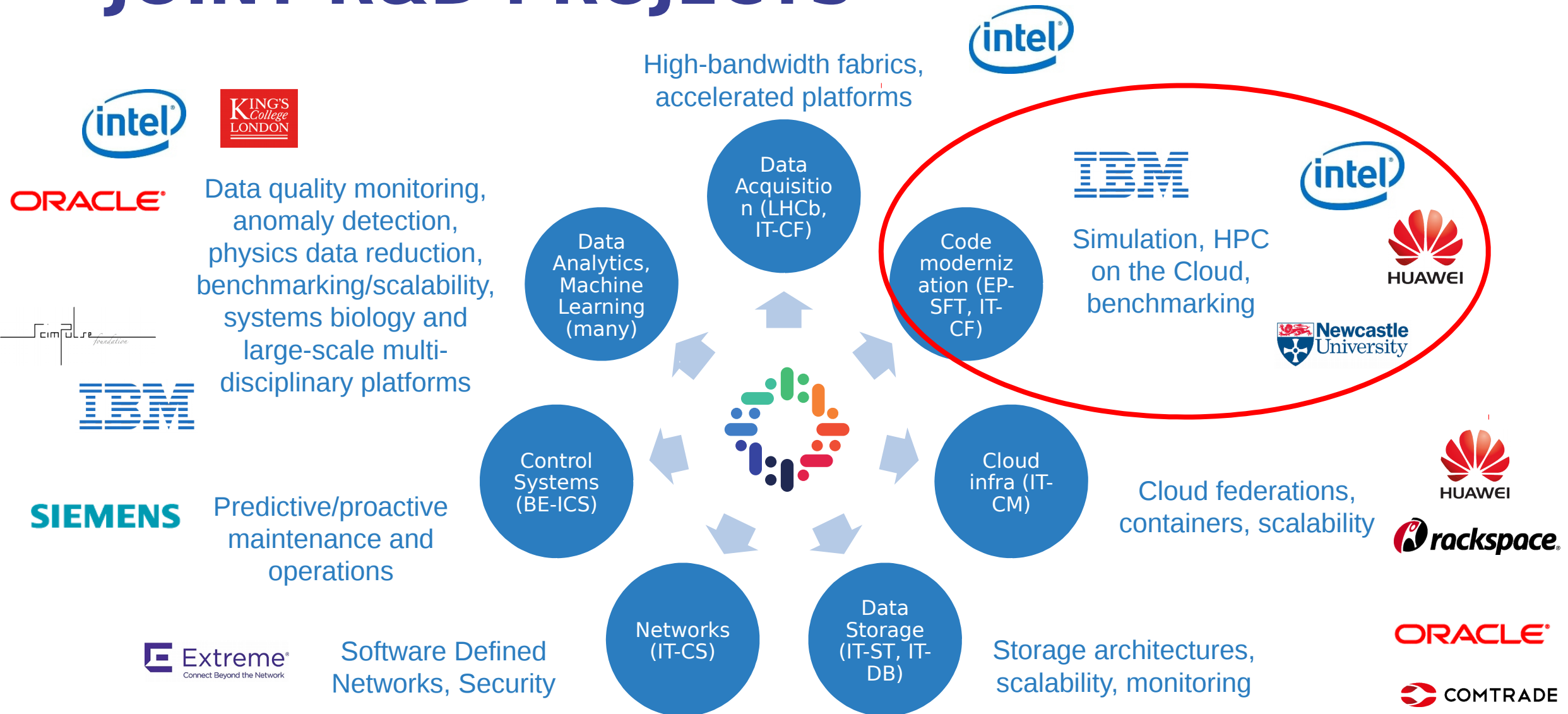
Test in a **research environment** today what will be used in many business sectors tomorrow

Training

Dissemination and outreach



JOINT R&D PROJECTS



CERN openlab Technical Student

- Master eindproject bij CERN openlab
- Doorgaan met zomerstage project: BioDynaMo

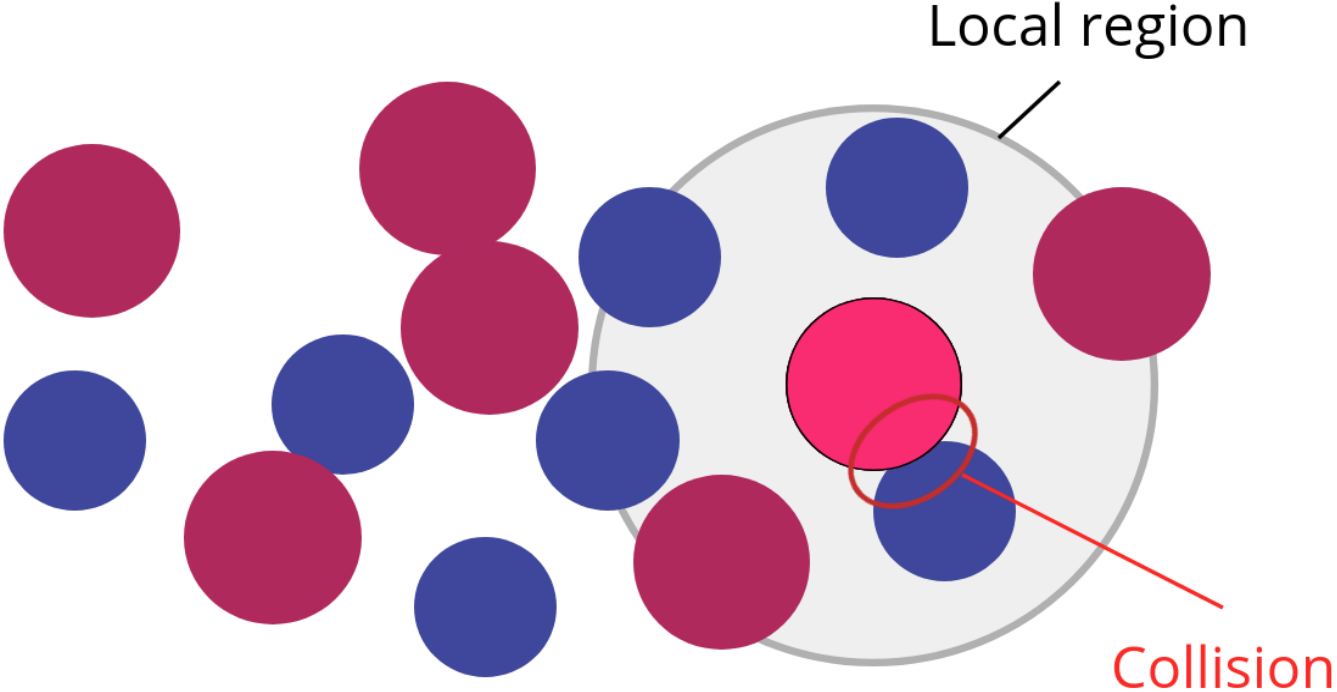


BIODYNAMO
BIOLOGY DYNAMICS MODELLER

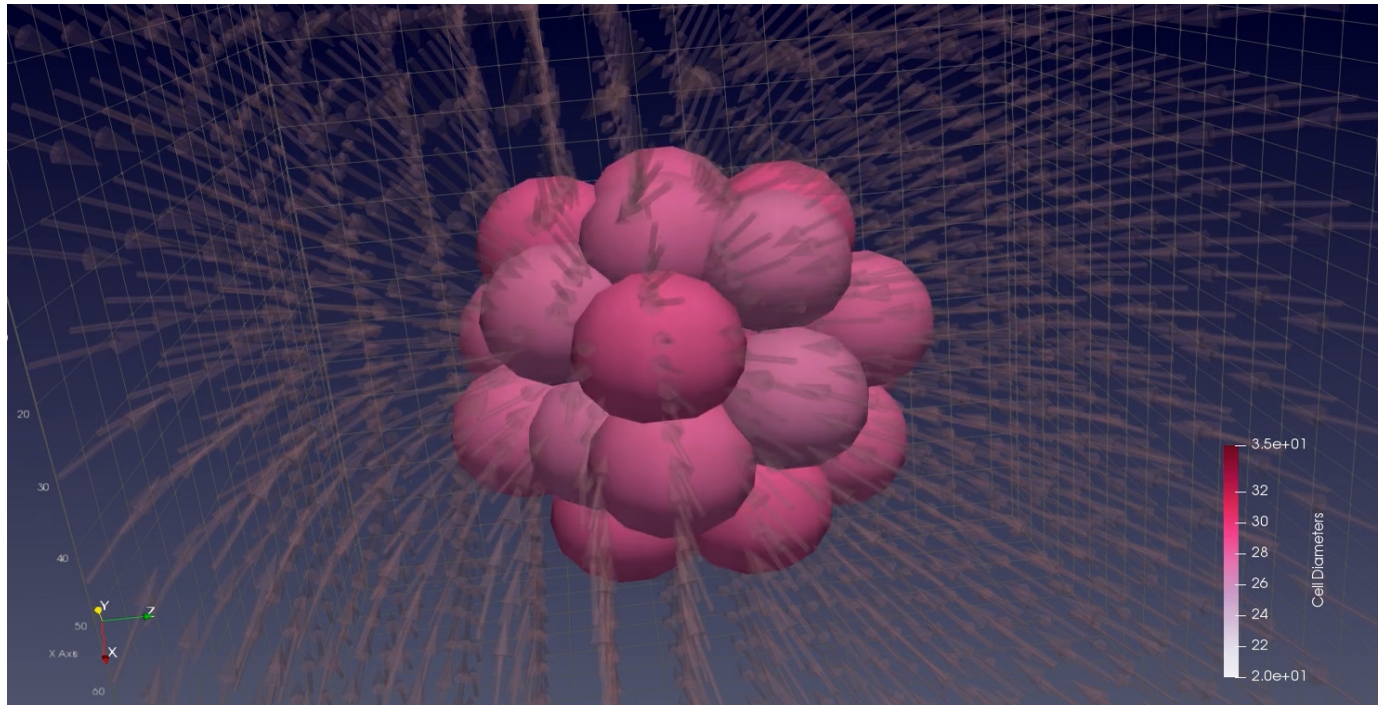
Een “Knowledge Transfer” project: deel de kennis van CERN met andere wetenschappen

Agent-based simulation

Simulation object = *Agent*



Cellgroei



- Elke cell is geprogrammeerd
- Maak je eigen regels
- BioDynaMo zorgt ervoor dat het snel gesimuleerd wordt

Code

- Simulaties worden geschreven in C++ (zoals veel HEP software)
- C++ is snel, maar lastig voor beginners
- Natuurkundigen en bio-wetenschappers zijn er vaak niet bekend mee
- Hoe pakken we dit aan?

```
#ifndef DEMO_CELL_DIVISION_MODULE_H
#define DEMO_CELL_DIVISION_MODULE_H

#include "biodynamo.h"

namespace bds {

// .....
// This model creates a grid of 128x128x128 cells. Each cell grows until a
// specific volume, after which it proliferates (i.e. divides).
// .....

inline int Simulate(int argc, const char** argv) {
    // Create new simulation
    Simulation simulation(argc, argv);
    // Since sim objects in this simulation won't modify neighbors, we can
    // safely disable neighbor guards to improve performance.
    simulation.GetExecutionContext()->DisableNeighborGuard();

    // Define initial model - in this example: 3D grid of cells
    size_t cells_per_dim = 128;
    auto construct = [](const std::array<double, 3>& position) {
        Cell* cell = new Cell(position);
        cell->SetDiameter(30);
        cell->SetAdherence(0.4);
        cell->SetMass(1.0);
        cell->AddBiologyModule(new GrowDivide());
        return cell;
    };
    ModelInitializer::Grid3D(cells_per_dim, 20, construct);

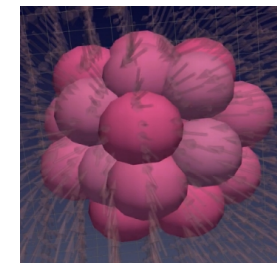
    // Run simulation for one timestep
    simulation.GetScheduler()->Simulate(1);

    std::cout << "Simulation completed successfully!\n";
    return 0;
} // namespace bds

#endif // DEMO_CELL_DIVISION_MODULE_H
```



Compilieren en uitvoeren



Notebooks

- ROOT is een software ontwikkeld door CERN
- Ontwikkeld om physici te helpen bij analyses
- ROOT Notebooks is een van de aspecten die het makkelijker maakt om analyse te doen
- Wij gebruiken het om biomedische wetenschappers te helpen met hun simulaties



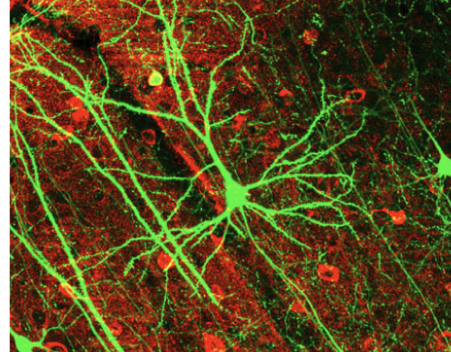
ROOT single-neuron-mode Last Checkpoint: één uur geleden (unsaved changes) Logout Terminal

File Edit View Insert Cell Kernel Widgets Help Trusted | ROOT C++

Pyramidal Cell Demo

Hello! In this demonstration we shall be simulating a so-called "pyramidal cell" with the simulation platform BioDynaMo. A pyramidal cell is one of the many types of neurons that can be found in the human brain. Pyramidal cells are mostly present in the motor cortex that is responsible for the execution and programming of voluntary movements.

Pyramidal cells typically look like the following in real life:



As you can see the cell body is cone shaped, which is characteristic for this type of neuron, which is why it earned the name "pyramidal cell". These neurons are typically about 100 micrometer in length, but some of them can grow up to a couple of centimeters.

Understanding the development of pyramidal cells is crucial in understanding how the human brain executes certain motoric tasks and can help in the fight against neuronal disorders, such as epilepsy and the disease of Alzheimer's.

Let's take a look at how we can simulate the development of a pyramidal cell.

```
In [2]: auto our_parameters = [&](Param* param) {
    param->bound_space_ = true;
    param->min_bound_ = -150;
    param->max_bound_ = 450;
    double max_length = 2.0;
    param->GetModuleParam<bdm::experimental::neuroscience::Param>()->neurite_max_length_ = max_length;

    param->detect_static_sim_objects_ = true;
    param->cache_neighbors_ = true;
};

bdm::experimental::neuroscience::InitModule();
```

Now we can initialize the simulation with an appropriate name and the parameters that we just set.

```
In [3]: Simulation simulation("piramidecell", our_parameters);
```

Info: Initialize new simulation using BioDynaMo v0.1.0-431-g47c113e

The core of a pyramidal cell is generally called a "Soma", in BioDynaMo we model a Soma as a sphere shaped object. Although this is quite a simplification of its real-life shape, for a lot of simulations it does not matter!

Demo tijd: simuleer je eigen piramidecel



1) Open je laptop en login: wachtwoord is **Acoolpassword!**

2) Dubbelklik op het Oracle VirtualBox icoon

3) Selecteer **Ubuntu_BioMasterClass** in de lijst

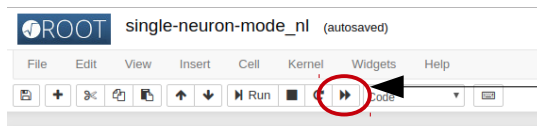
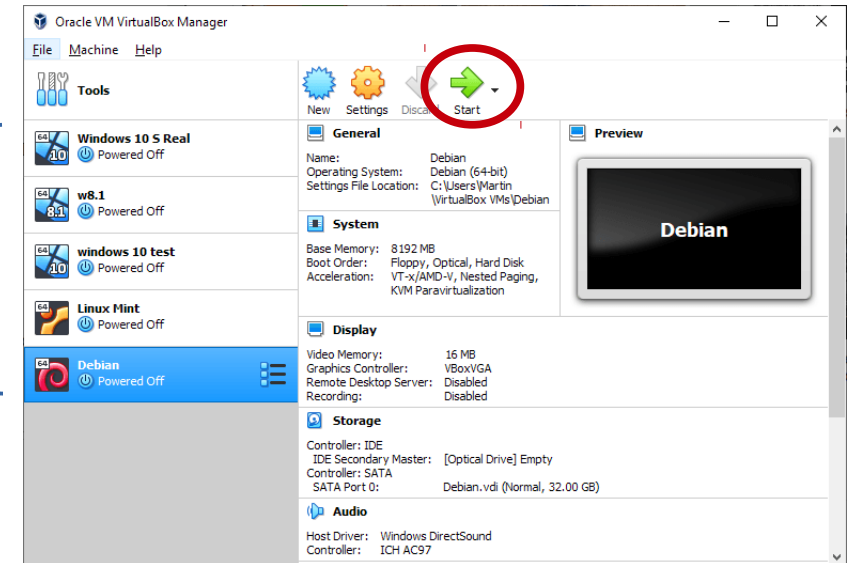
4) Klik op **Start**

5) Neem de notebook door en voor de code uit door op Shift + Enter te klikken

Probeer te begrijpen wat de code doet

6) Na het eenmaal doorgenomen te hebben: probeer wat parameters aan te passen en kijk wat het effect is.

Wie kan een tweede piramidecel maken?



Klik om te resetten en alle cellen opnieuw uit te voeren

Piramidecell Demonstratie

Hallo! In deze demonstratie gaan we een zogeheten "piramidecel" simuleren. Dit zijn soorten zenuwcellen (ookwel neuronen genoemd) die te vinden zijn in

Continuous Integration

Search all repositories

My Repositories +

- ✓ BioDynaMo/biodynamo # 2129
 - 🕒 Duration: 1 hr 45 min 31 sec
 - 📅 Finished: 3 days ago

BioDynaMo / biodynamo  build passing

Current Branches Build History Pull Requests


More options 

✓ **master** Copy libBDMGlyphFilter.so to correct location ➔ #2129 passed [Restart build](#)

➔ Commit 4ec1214 [🔗](#) 🕒 Ran for 35 min 36 sec








🔗 Compare 293d848...4ec1214 [🔗](#) 🕒 Total time 1 hr 45 min 31 sec

🔗 Branch master [🔗](#) 📅 3 days ago

 Lukas Breitwieser

Build jobs

[View config](#)

✓ # 2129.1	 </> Compiler: gcc	📦 SCRIPT="util/run-inside-docker.sh ubuntu-16.04 util/travis-ci/default-build.sh"	🕒 17 min 38 sec	🔄
✓ # 2129.2	 </> Xcode: xcode10.1	📦 SCRIPT="util/travis-ci/default-build.sh"	🕒 20 min 37 sec	🔄
✓ # 2129.3	 </> Compiler: gcc	📦 SCRIPT="util/run-inside-docker.sh ubuntu-16.04 util/travis-ci/check-code-style.sh"	🕒 7 min 31 sec	🔄
✓ # 2129.4	 </> no language set	📦 SCRIPT="util/travis-ci/installation-test.sh ubuntu-16.04"	🕒 13 min 1 sec	🔄
✓ # 2129.5	 </> no language set	📦 SCRIPT="util/travis-ci/installation-test.sh ubuntu-18.04"	🕒 14 min 14 sec	🔄
✓ # 2129.6	 </> no language set	📦 SCRIPT="util/travis-ci/installation-test.sh centos-7"	🕒 11 min 19 sec	🔄
✓ # 2129.7	 </> Xcode: xcode10.1	📦 SCRIPT="util/travis-ci/installation-test.sh osx"	🕒 21 min 11 sec	🔄

Continuous Integration

Default Branch

✓ master 🔑 275 builds	# 2129 passed 📅 3 days ago	🔗 4ec1214 👤 Lukas Breitwieser	✓	✓	✓	✓	✓
--------------------------	-------------------------------	----------------------------------	---	---	---	---	---

Active Branches

✗ root-notebook-rebase 🔑 13 builds	# 2128 failed 📅 5 days ago	🔗 47c113e 👤 Ahmad Hesam	✗	⊘	⊘	⊘	✗
✓ lukas 🔑 112 builds	# 2127 passed 📅 5 days ago	🔗 4ec1214 👤 Lukas Breitwieser	✓	✓	✓	✗	✓
! sbml-integration-osx 🔑 1 build	# 2069 errored 📅 15 days ago	🔗 37af96f 👤 Ahmad Hesam	!	⊘	⊘	⊘	⊘
...	...		✗	⊘	✗	✗	✗

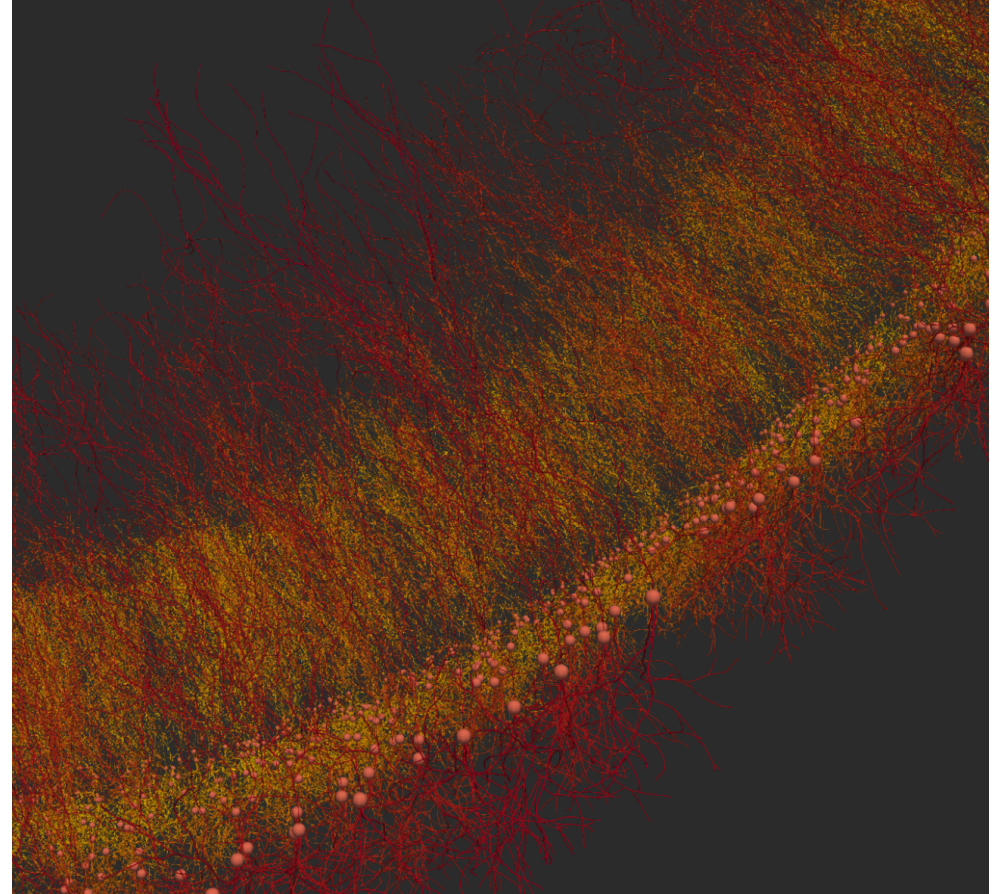
Open-Source Code

<https://github.com/BioDynaMo/biodynamo>

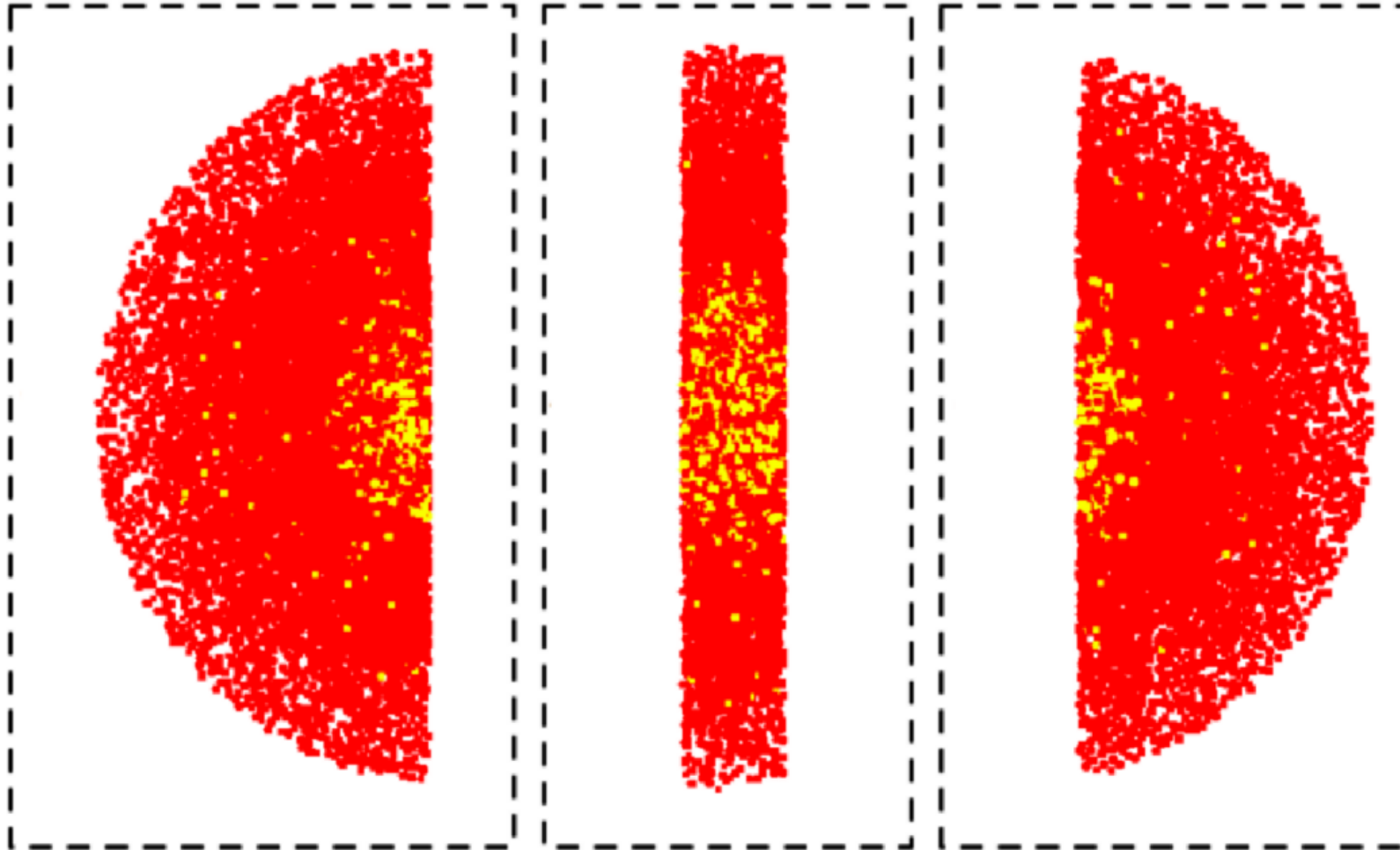
- Code is publiekelijk beschikbaar
- We moedigen contributies aan
- Het uiteindelijke doel is om BioDynaMo de defacto standaard te maken in biomedische simulaties

High-Performance Computing

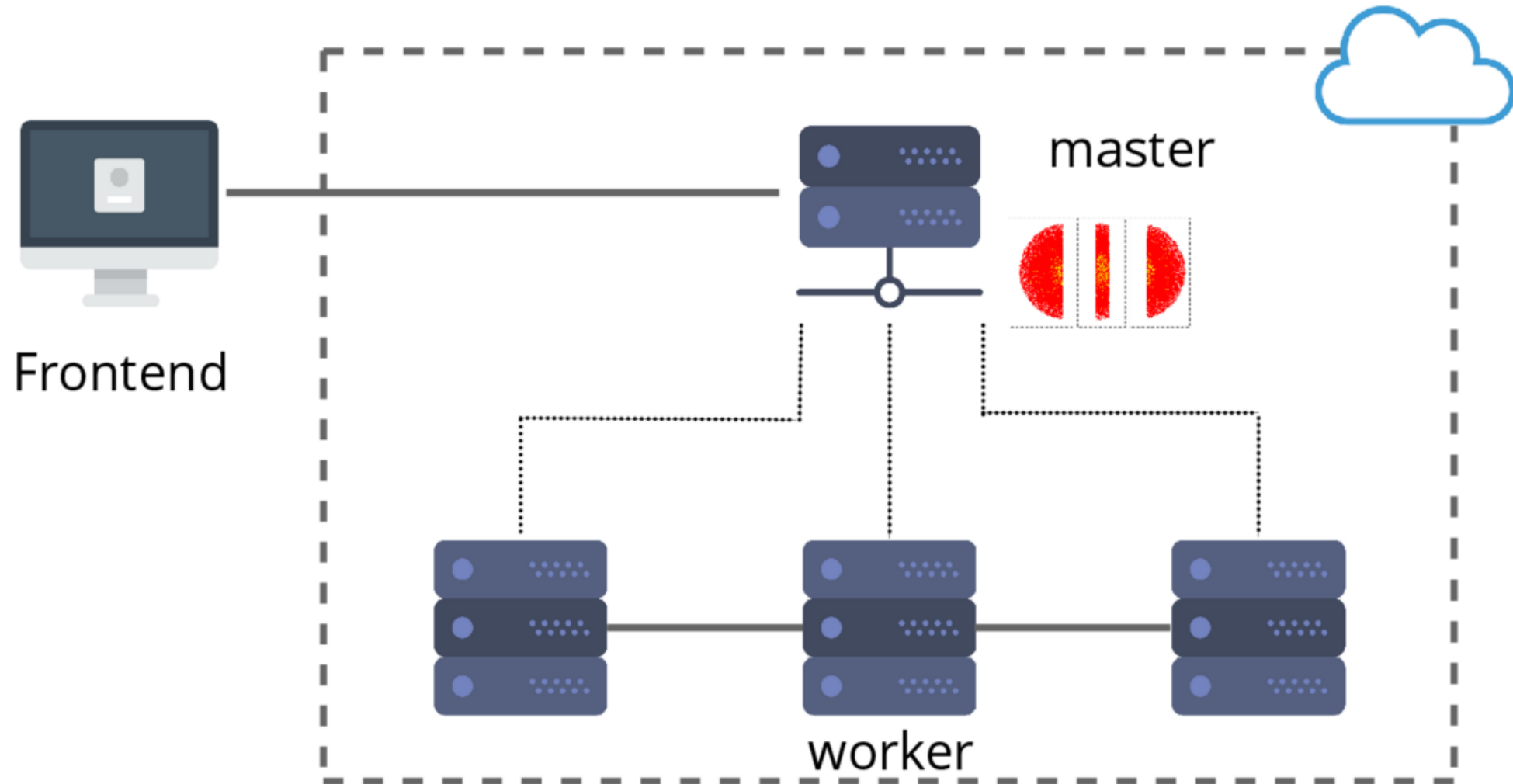
- Simulaties van een paar neuronen kan gemakkelijk op een laptop
- Maar simulaties van duizenden neuronen moet gedaan worden op een krachtigere computer
- Sommige simulaties kunnen zo complex worden dat zelfs krachtige computers het niet aankunnen
- Oplossing: High-Performance Computing
 - Gedistribueerd: verdeel de simulatie over meerdere computers die tegelijkertijd werken
 - Hardware accelerators: GPUs / FPGAs



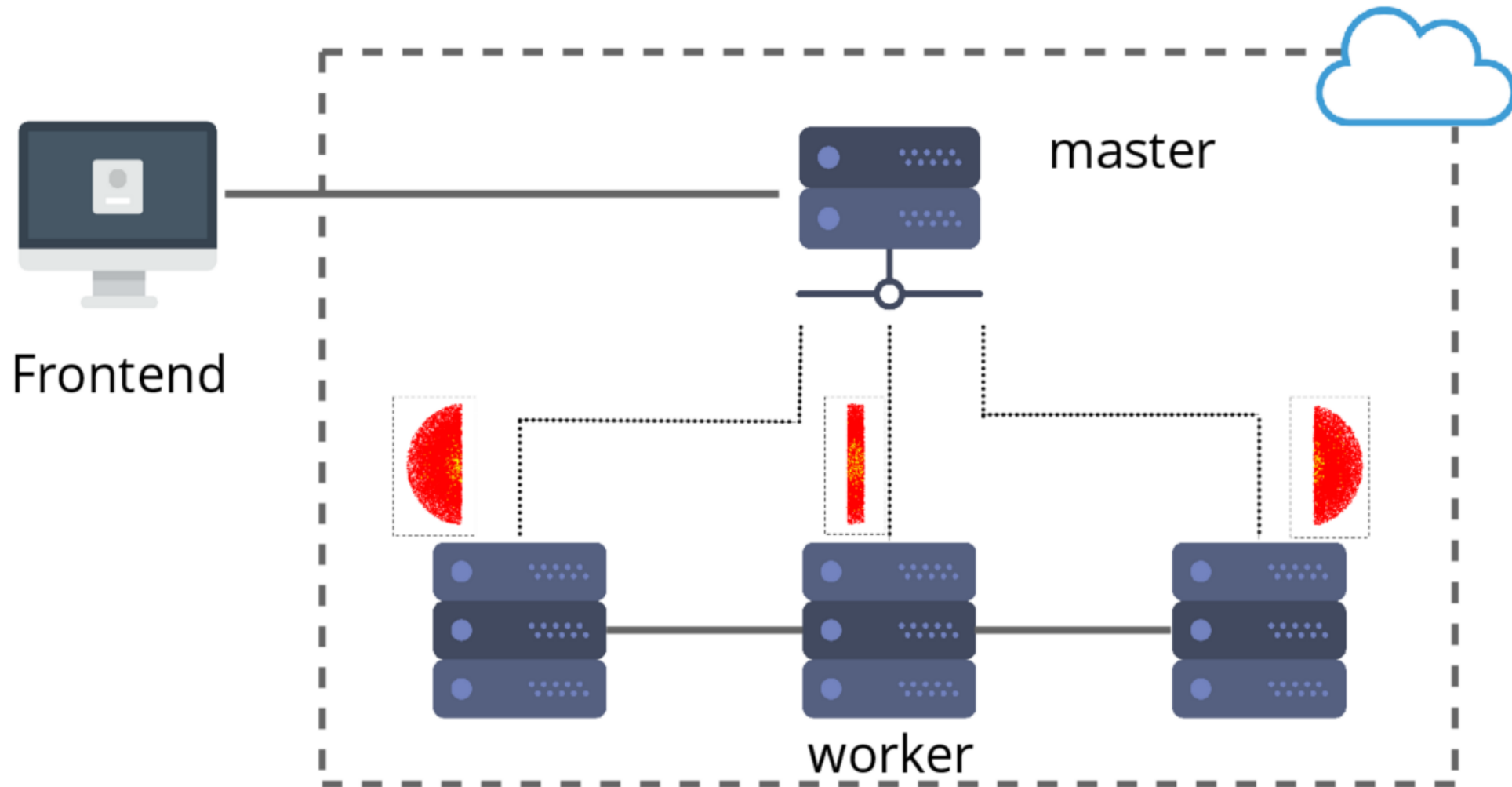
Domain-Decomposition



Distributed Runtime



Distributed Runtime



CPU vs GPU: A Demo

<https://www.youtube.com/watch?v=-P28LKWTzrI>

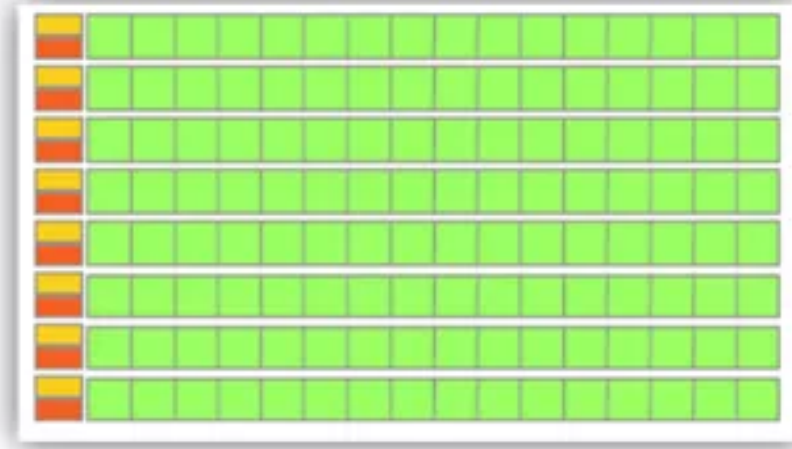
High-Performance Computing

CPU



Een paar cores...

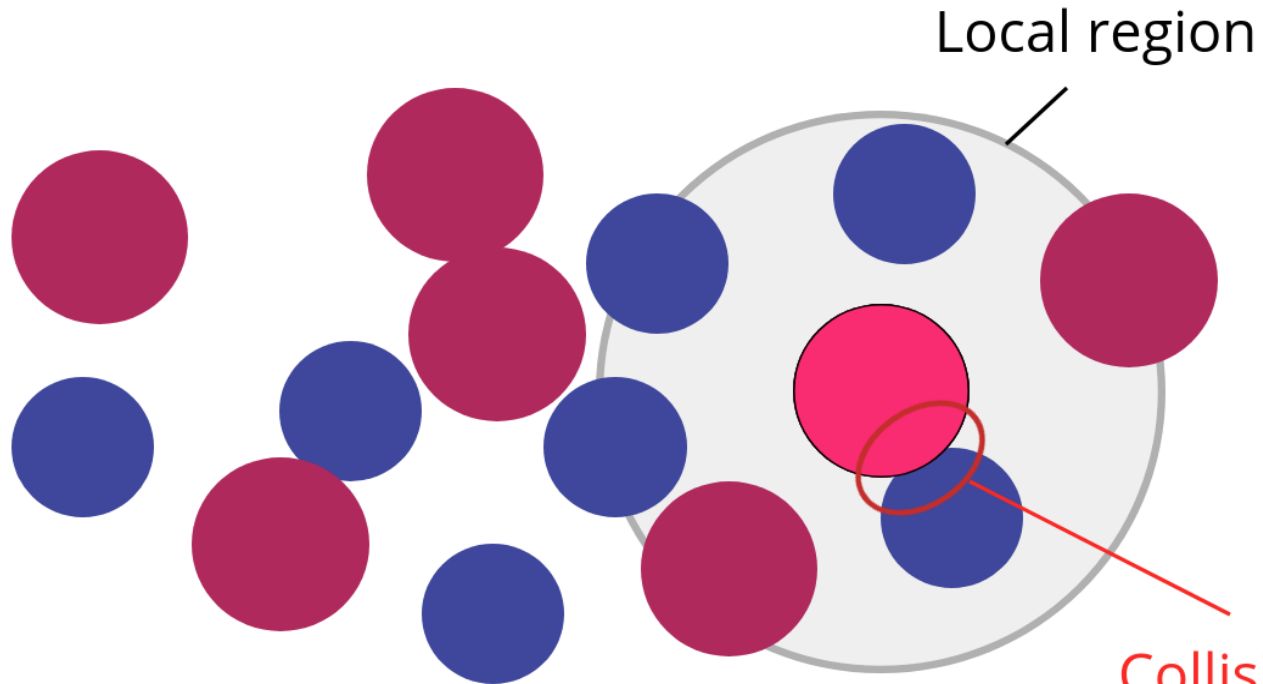
GPU



Een paar duizend cores!

Agent-based simulation

Simulation object = *Agent*



Mijn onderzoeksvraag

Kunnen we dit sneller doen?

Collision $\approx 80\%$ van de tijd!

Resultaten

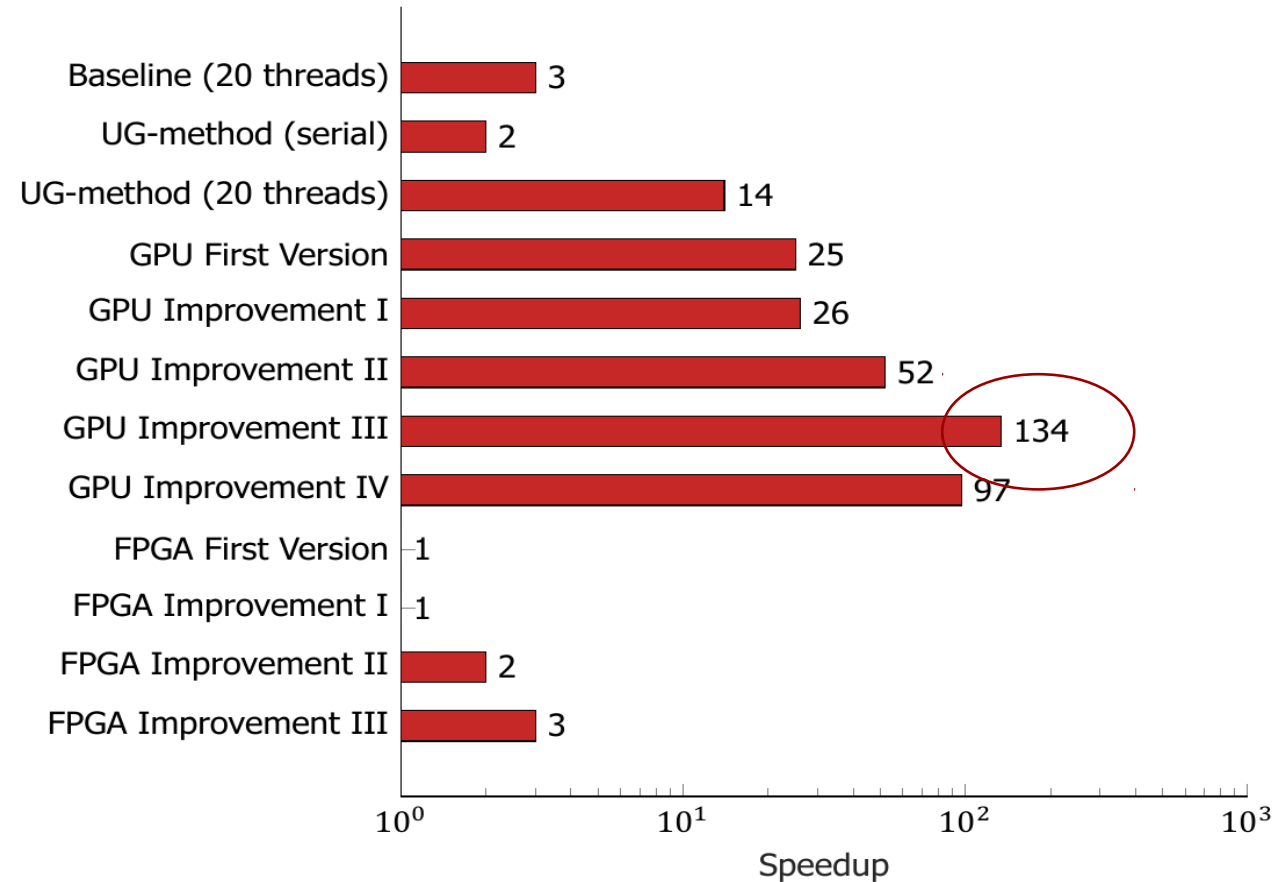


Figure 5.2: The speedup per simulation step with respect to the serial baseline version for all mechanical interaction implementations, with $num_{objects} = 2097152$.



QUESTIONS?

Ahmad Hesam

ahmad.hesam@cern.ch