Distributed BioDynaMo

Nam Nguyen

Supervisors: Lukas Breitwieser
Ahmad Hesam
Fons Rademakers

European Organization for Nuclear Research (CERN)

August 15, 2018

At a glance ↑

Context

Project

Status

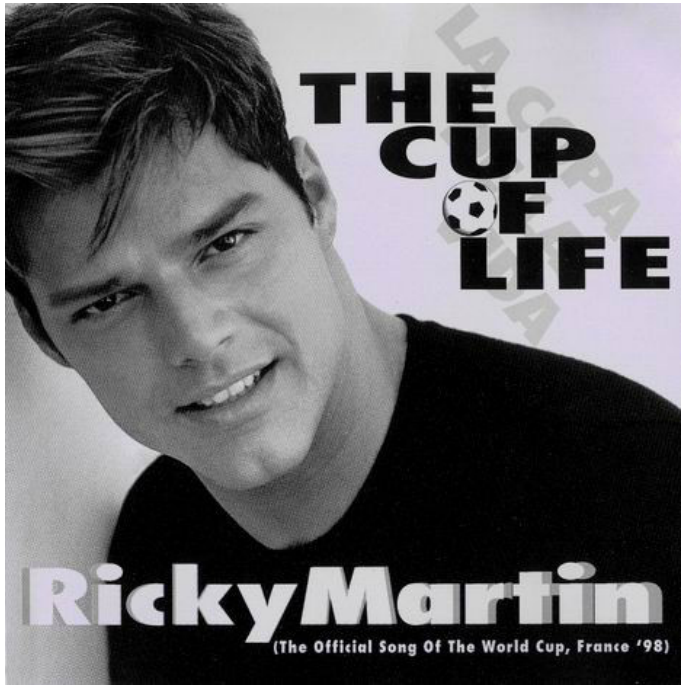
Future work

About myself

- From San Francisco bay area, California, USA
- Recently graduated from GATech MSc in CompSci in May, 2018

About myself

- From San Francisco bay area, California, USA
- Recently graduated from GATech MSc in CompSci in May, 2018
- Old enough to celebrate another French championship (in 20 years) of The Cup of Life ale ale ale!



© Sony Music Entertainment Inc.

BioDynaMo overview

- *Large-scale* platform for biological simulation
- Started in 2015 as a CERN openlab summer student project
- International collaboration

CERN openlab

Newcastle University

intel

ETH zürich

INNOPOLIS UNIVERSITY

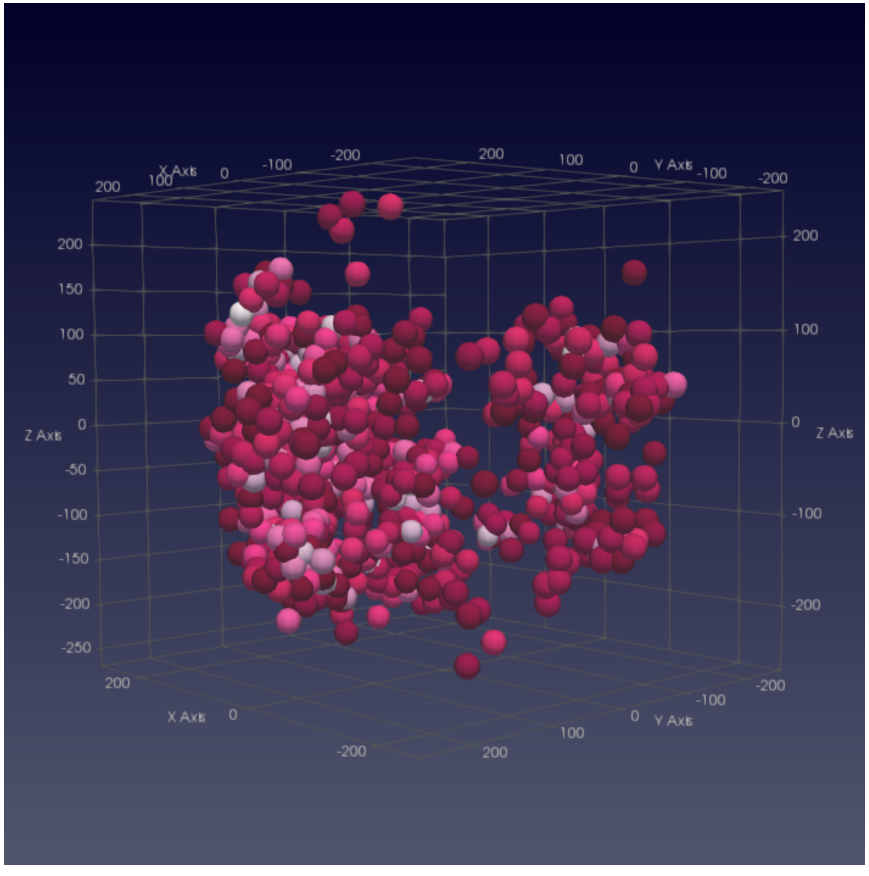
KAZAN FEDERAL UNIVERSITY

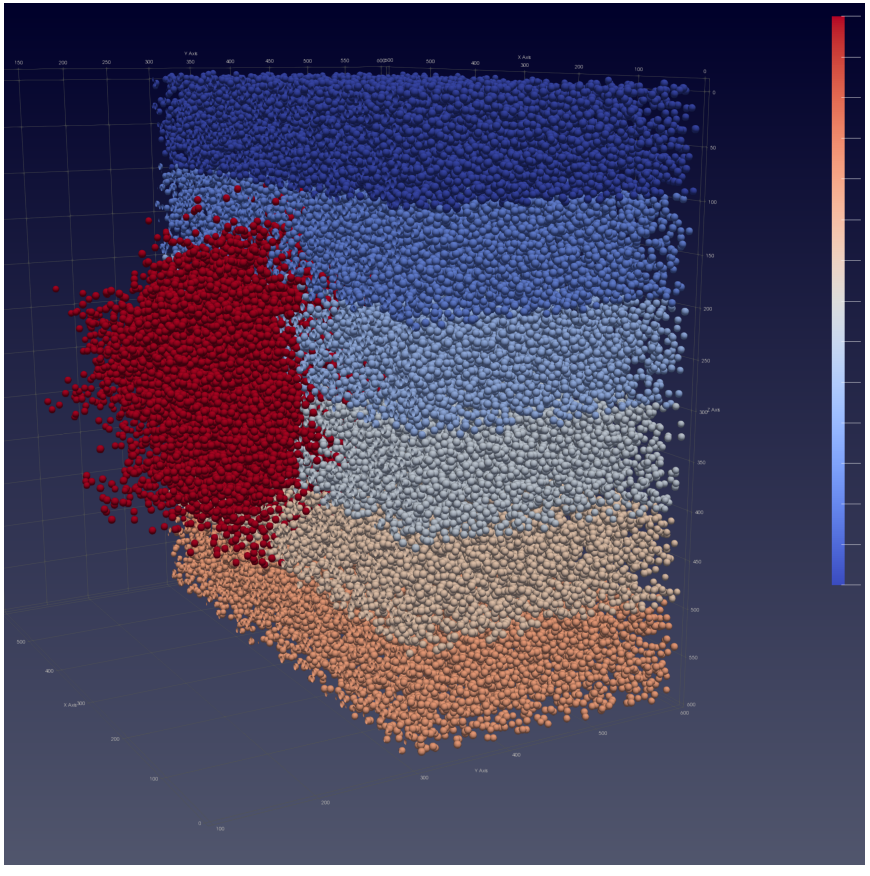
BioDynaMo overview

Video of some simulations (Adobe Reader required)

BioDynaMo overview

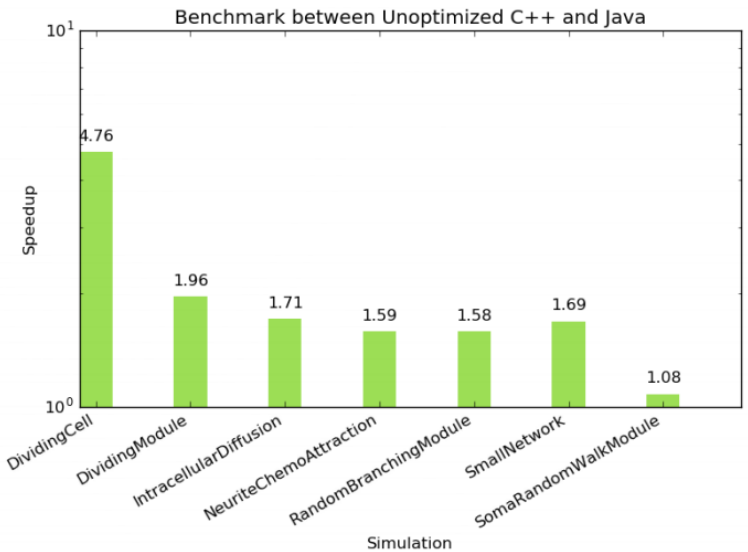
Backup photos

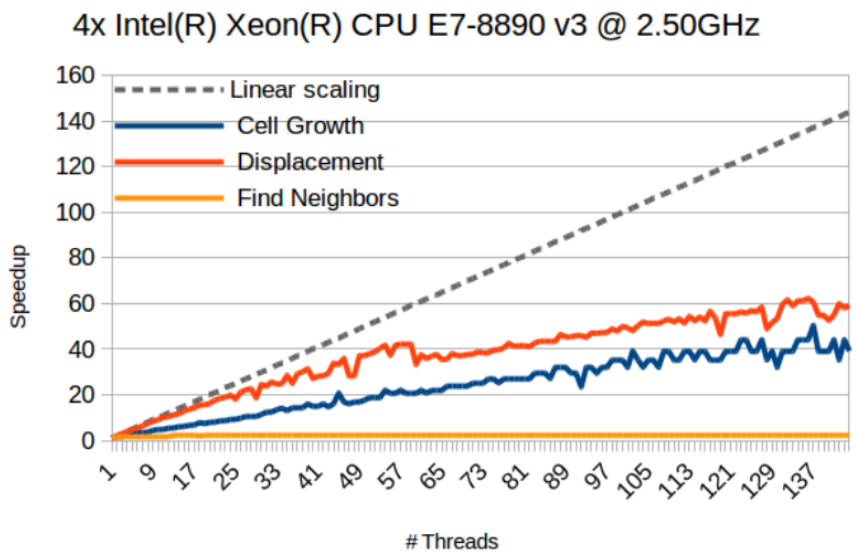




BioDynaMo status

- Support different specialities
- Scale *up* nicely (see status report in 2017)
- Does not scale *out*, yet
  - Konstantinos Kanellis prototyped a message passing layer last summer





My work

1. Small fry
   - Touch up last summer code
   - Reorganize demos and tests
2. Big fish
   - Implement a proof-of-concept to scale *out* with distributed computing

Project status

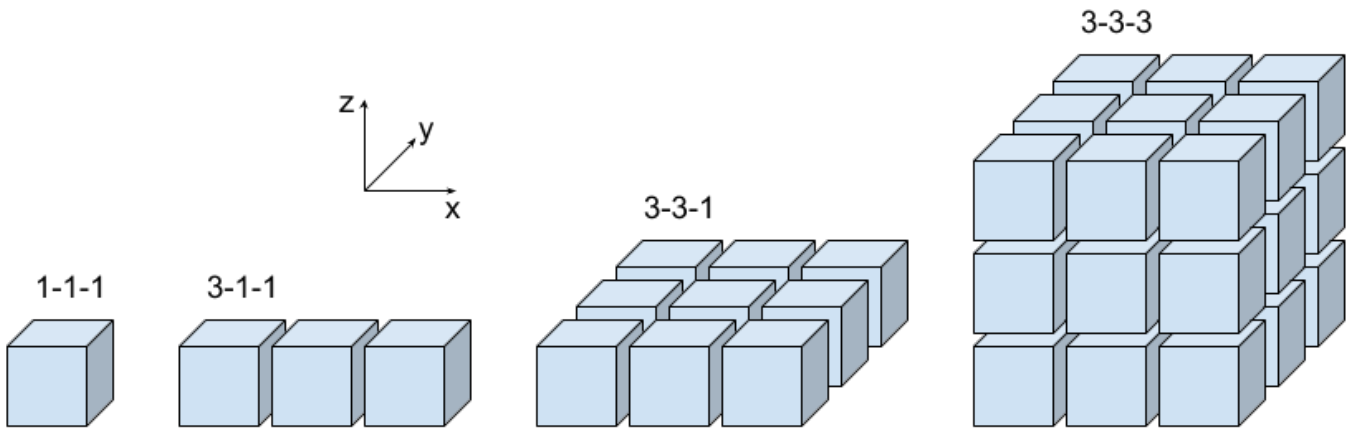
Small fry

1. Last summer code
   - 1.1 had third party dependencies (ZeroMQ libraries) built and packaged up
   - 1.2 is now buildable from fresh checkout
   - 1.3 is in `distribution` branch
2. Demos and tests are
   - 2.1 now organized into a more standard structure
   - 2.2 more conveniently copied out with `biodynamo demo` command
   - 2.3 automatically tested by Travis CI in every commit
   - 2.4 merged in `master`

Project status

Big fish (the work)

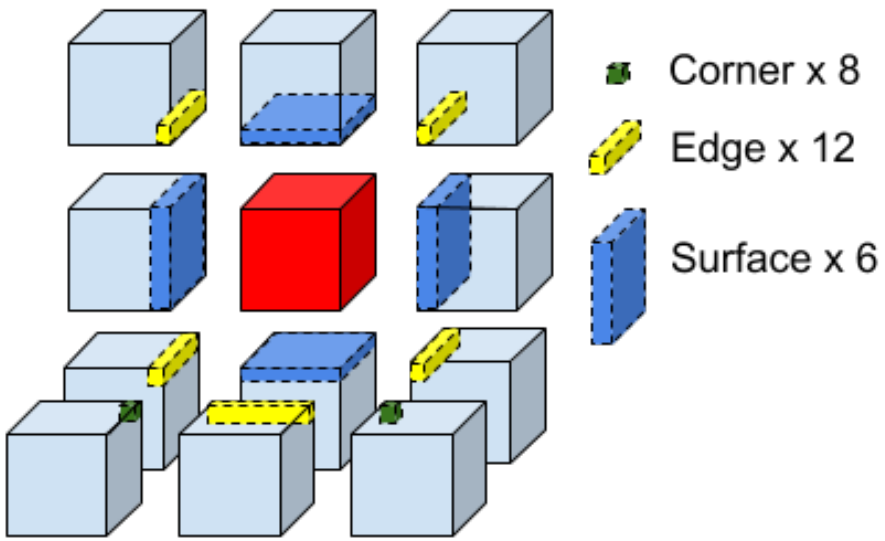
- Proposed to use Berkeley RISELab's Ray distributed execution engine
  - Stand on giant's shoulder
- Implemented a general axis-aligned 3D cube partitioning / work distribution scheme



Project status (cont.)

Big fish (the work)

- The partitioner knows about "halo" (border) regions.



- There are maximumly 8 corner regions, 12 edge regions, and 6 surface regions neighboring any particular main cube.

Project status (cont.)

Big fish (the work)

- Cubes are logical view of tasks.
- Each task comprises of 3 steps:
  - Reassembling all 26 neighboring regions and this cube from last time-step into 1 bigger cube
  - Executing one time-step simulation in this merged cube
  - Disassembling the cube back into 27 overlapping regions for the next time-step
- Python driver builds task dependency tree, and Ray manages/schedules the tasks.
- Python driver uses `ctypes` to load and invoke functions in C++ shared library.

Project status (cont.)

Big fish (the results)

- Sample demo `distributed` can be built from fresh checkout with all prebuilt dependencies packaged up.
- The driver can execute the demo in a Ray cluster, or as a local process.
- Distributed execution is transparent to the simulation code. Virtually no change to existing C++ code is required.
- There is large overhead in disassembling (serialization) and reassembling (deserialization) (up to 90% of execution time).
- Simulation objects are assumed to not move across regions.
- All code is in `experimental-ray` branch.

Future work

- Reconcile the movement of simulation objects
  - From a remote region moving into a main region
  - From a remote region moving into another remote region
  - From a main region moving outside
  - Farther than the halo distance
- Alleviate serialization / deserialization
- Ensure that tasks are given to nodes owning largest blobs
- Ensure deterministic PRNGs on different workers
- Support resuming simulation when needed
- Most importantly, validate the correctness of distributed execution

Fini

Questions and suggestions are much appreciated!

(But please don't ask if I would cover Ricky.
Do you really want it?
Do you really want it?)