Rust-AB

ABM Frameworks Review

&

Rust-AB: future plans

by Luca Postiglione

ISISLab

What we will talk about

ABM Frameworks Review

01 **ABM World**
What is an ABM, features and why we need frameworks

02 **Framework Features**
Which are the most common (and not) features offered and how frameworks are organized

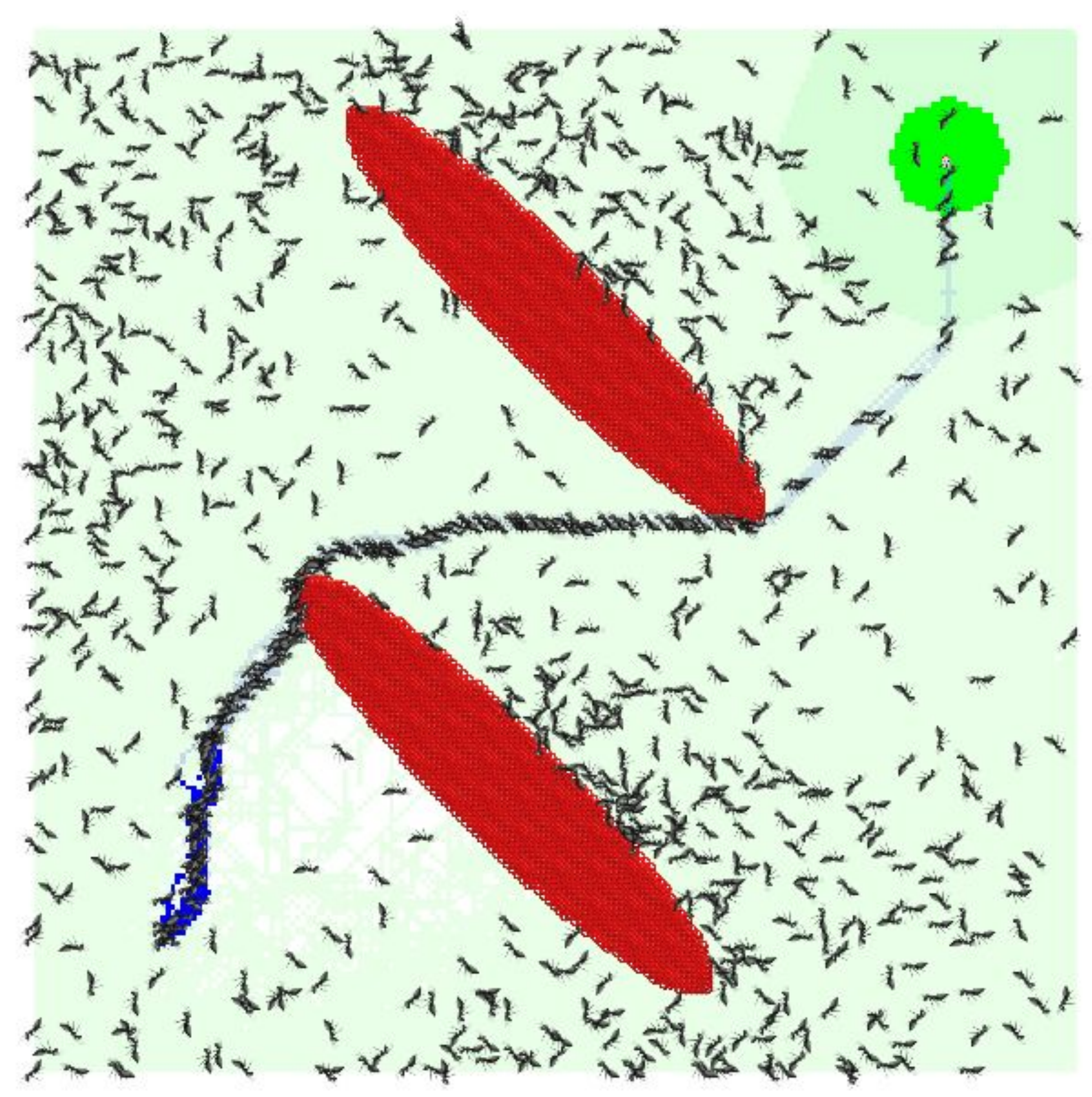
Rust-AB: future plans

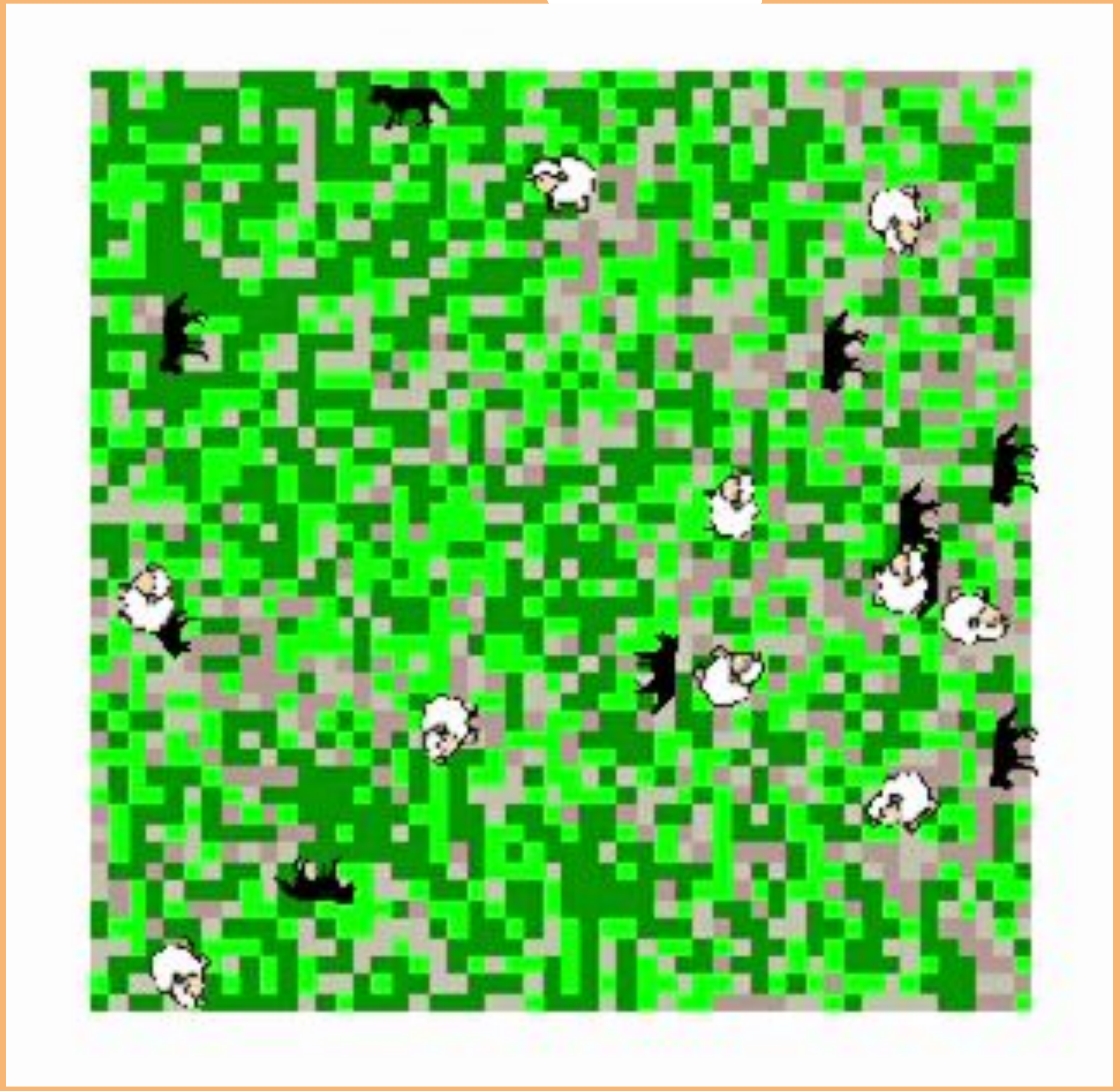
03 **Rust-AB team**
Who we are, how our work is organized and what we have done

04 **Framework evolution**
What we are developing, and my next step: **Parameter space exploration and optimization**

What is ABM

Agent Based Modelling are computer simulations used to study the interactions among entities, called **agents**, inside an environment.





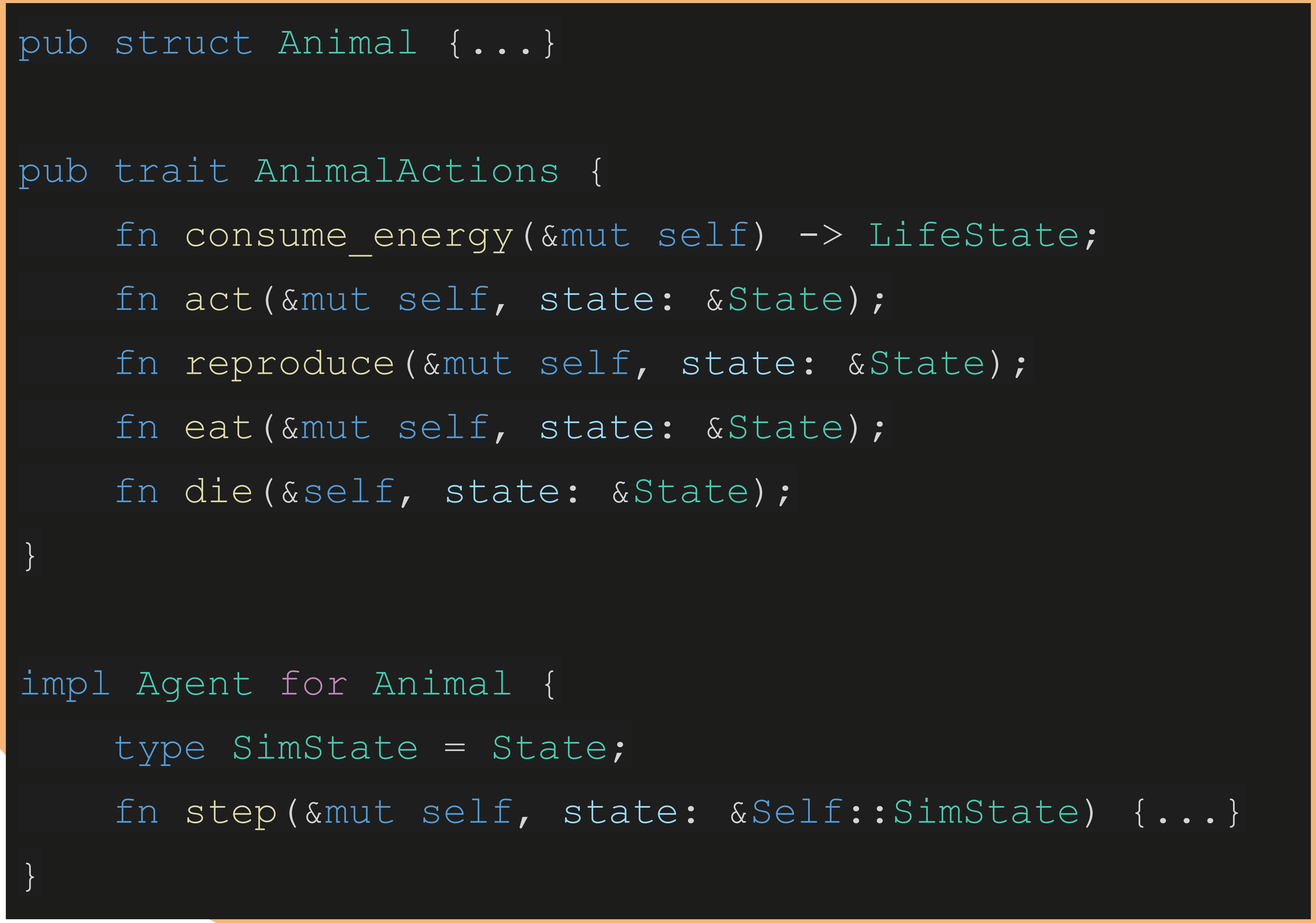
Model design, but not only...

It's not simple to define/implement, because a model is characterized of several rules/parameters.

Then you have to manage everything related to scheduling, visualization, agents and system global management, concurrency problems... and other "problems".

Does someone need help?

Focus on modelling



```
pub struct Animal {...}

pub trait AnimalActions {
    fn consume_energy(&mut self) -> LifeState;
    fn act(&mut self, state: &State);
    fn reproduce(&mut self, state: &State);
    fn eat(&mut self, state: &State);
    fn die(&self, state: &State);
}

impl Agent for Animal {
    type SimState = State;
    fn step(&mut self, state: &Self::SimState) {...}
}
```

It's **trivial...**

Model Developers want to write **model structs/behaviours**

ABM Frameworks

MASON

Completely based on **Java**

NetLogo

It's a **programming language** written in Java/Scala

Repast

Provides also an **HPC** version

Agents.jl

Based on Julia. It's part of **JuliaDynamics**

FLAME GPU

Extension of another framework.

Mesa

A **Python** framework

AnyLogic

Simulation software for **business**

Main features

01 Agents

Space

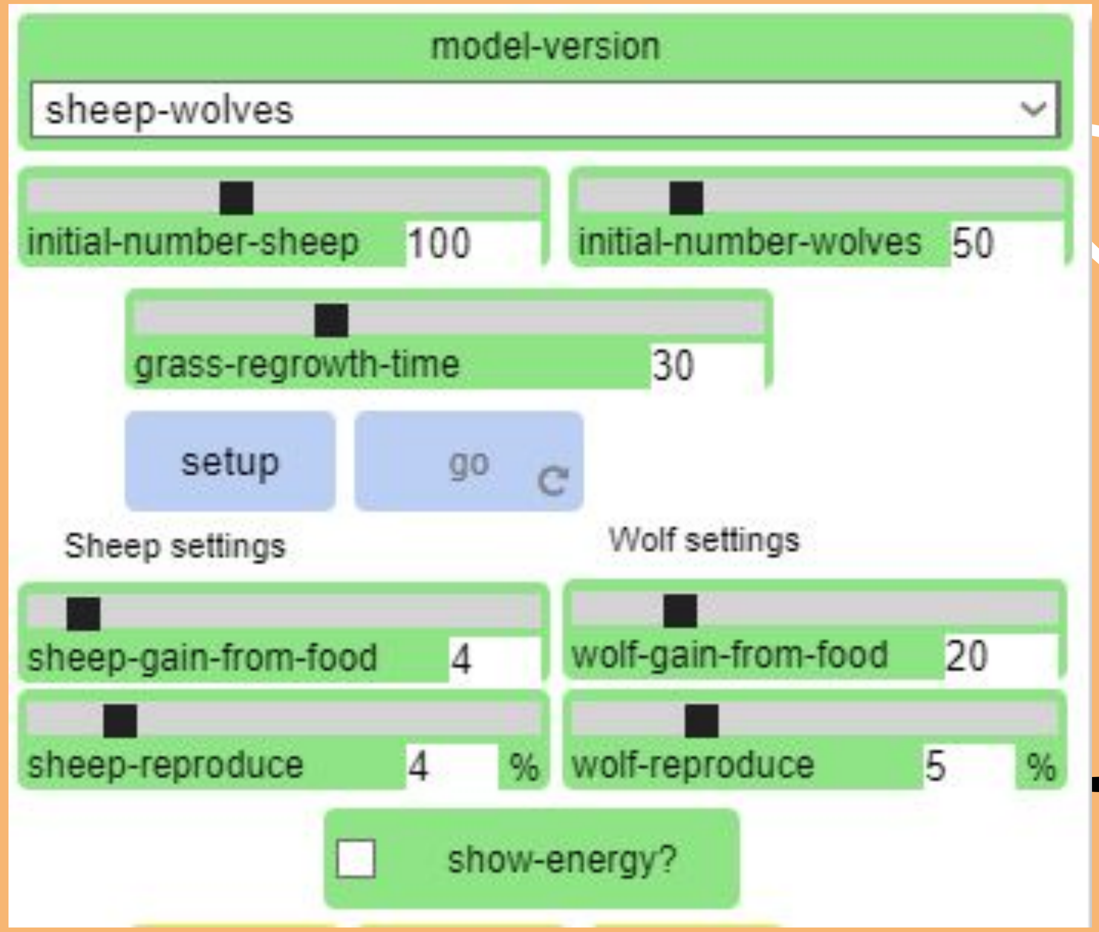
1. Grid Space
2. Continous Space
3. Graph Space

02 Visualization

2D or 3D, taking advantage of some graphic engine

03 GUI

To setup easily simulation parameter values


model-version: sheep-wolves

initial-number-sheep 100

initial-number-wolves 50

grass-regrowth-time 30

Sheep settings

sheep-gain-from-food 4

sheep-reproduce 4 %

Wolf settings

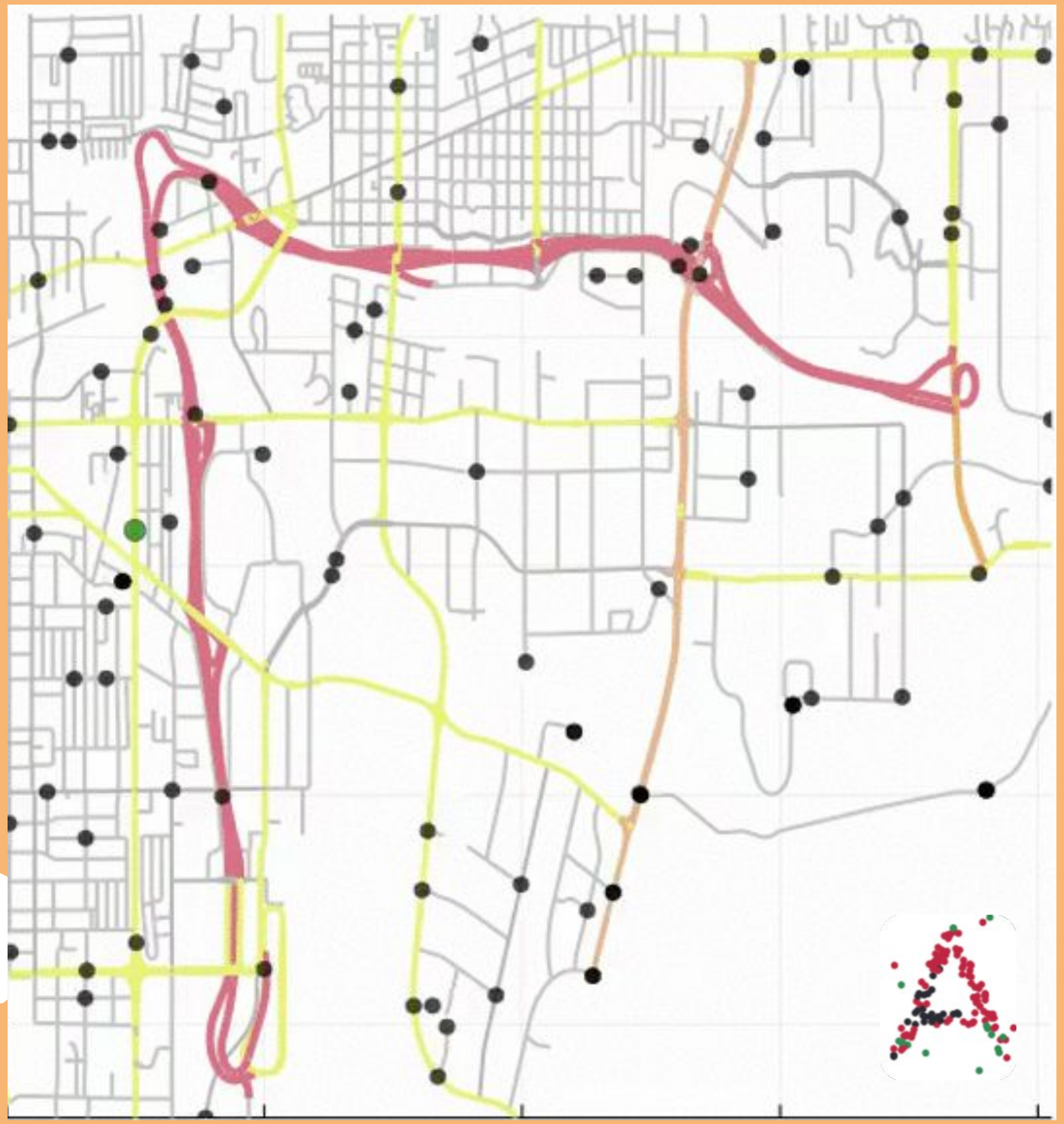
wolf-gain-from-food 20

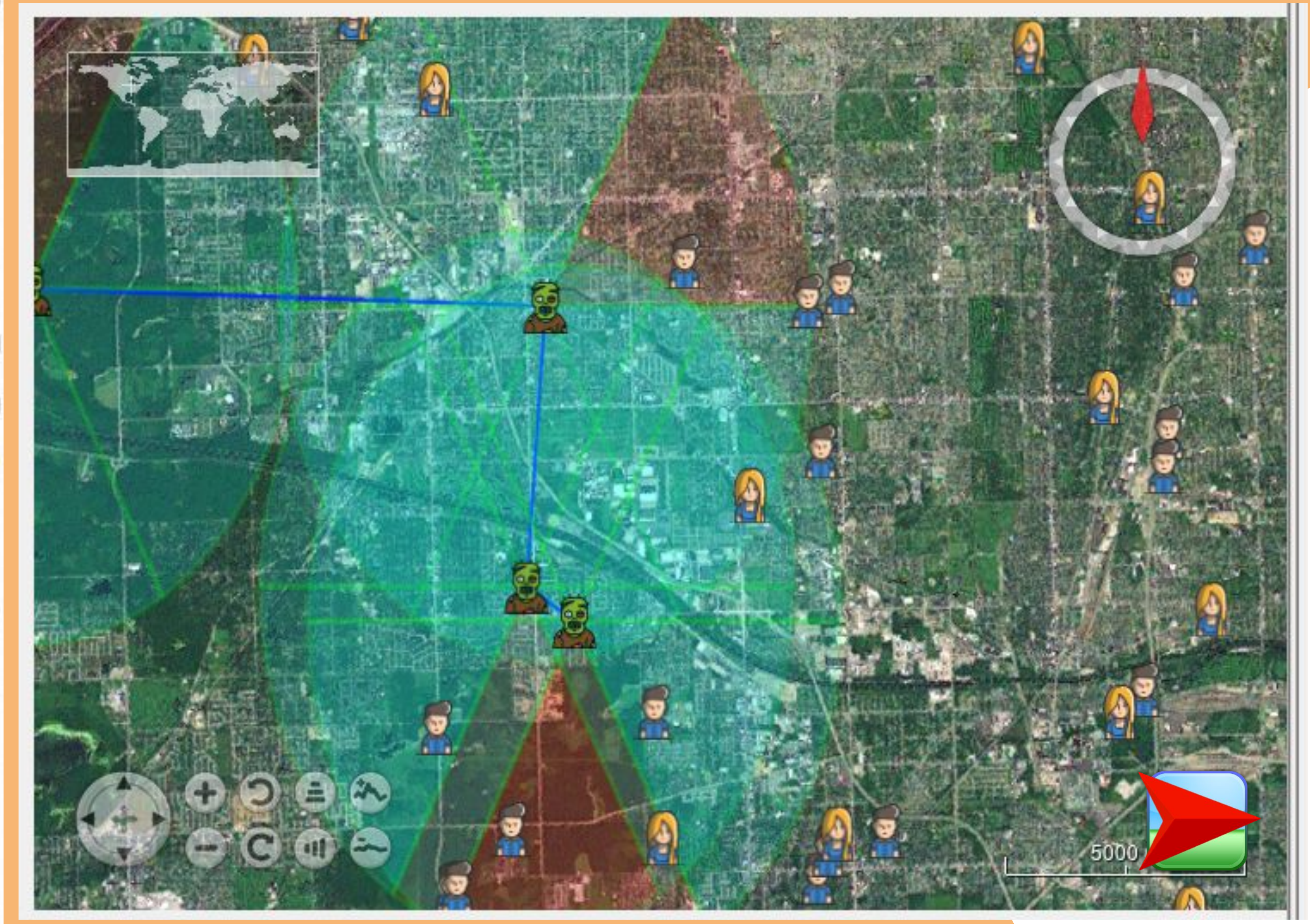
wolf-reproduce 5 %

□ show-energy?

OpenStreetMap & GIS Data

- Detailed space for agents & geographical informations;
- More realistic simulations;

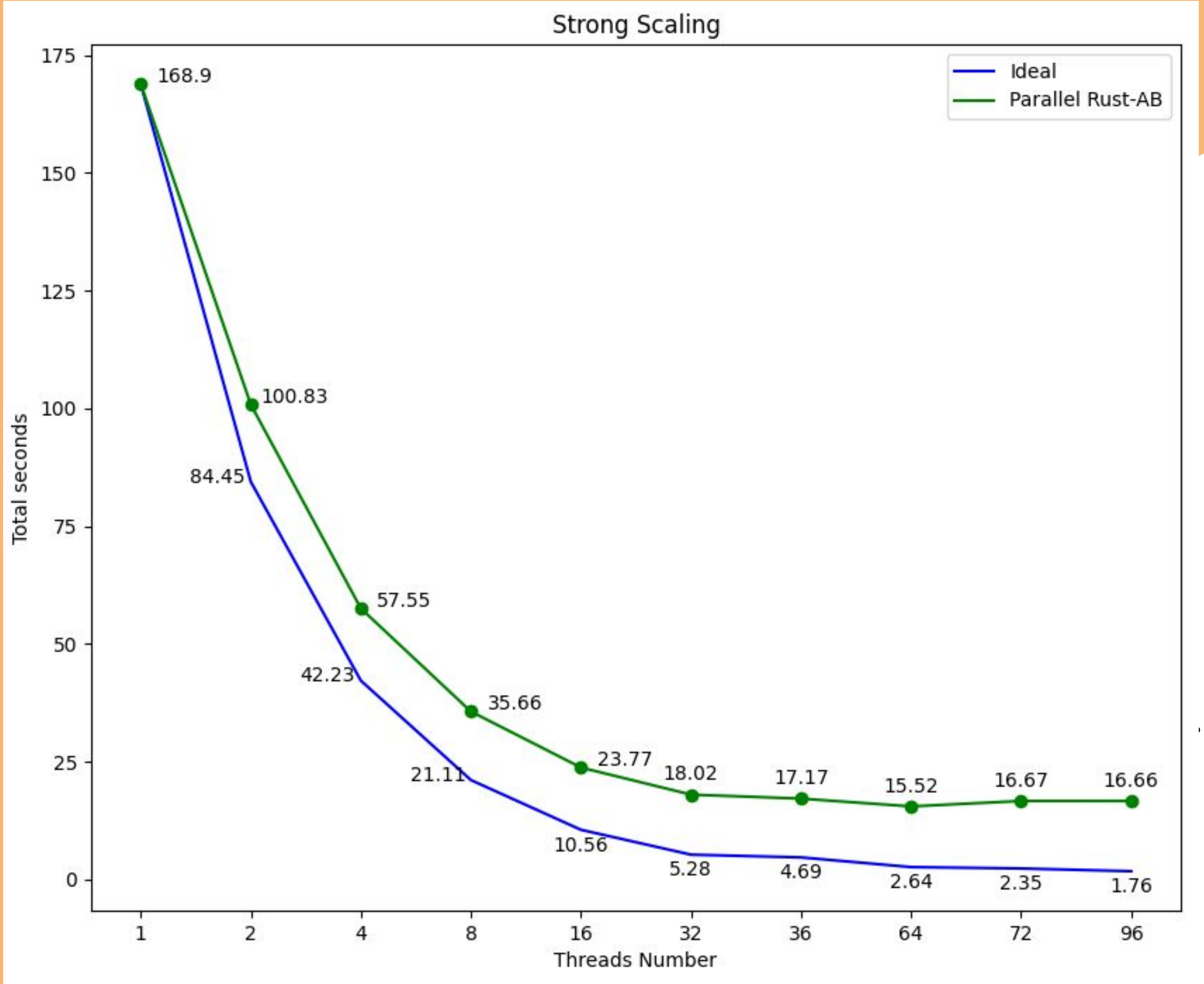




Performance are important (enabling)

- Some simulation requires many hour to be executed;
- Parallel & Distributed Cloud computing are the answer

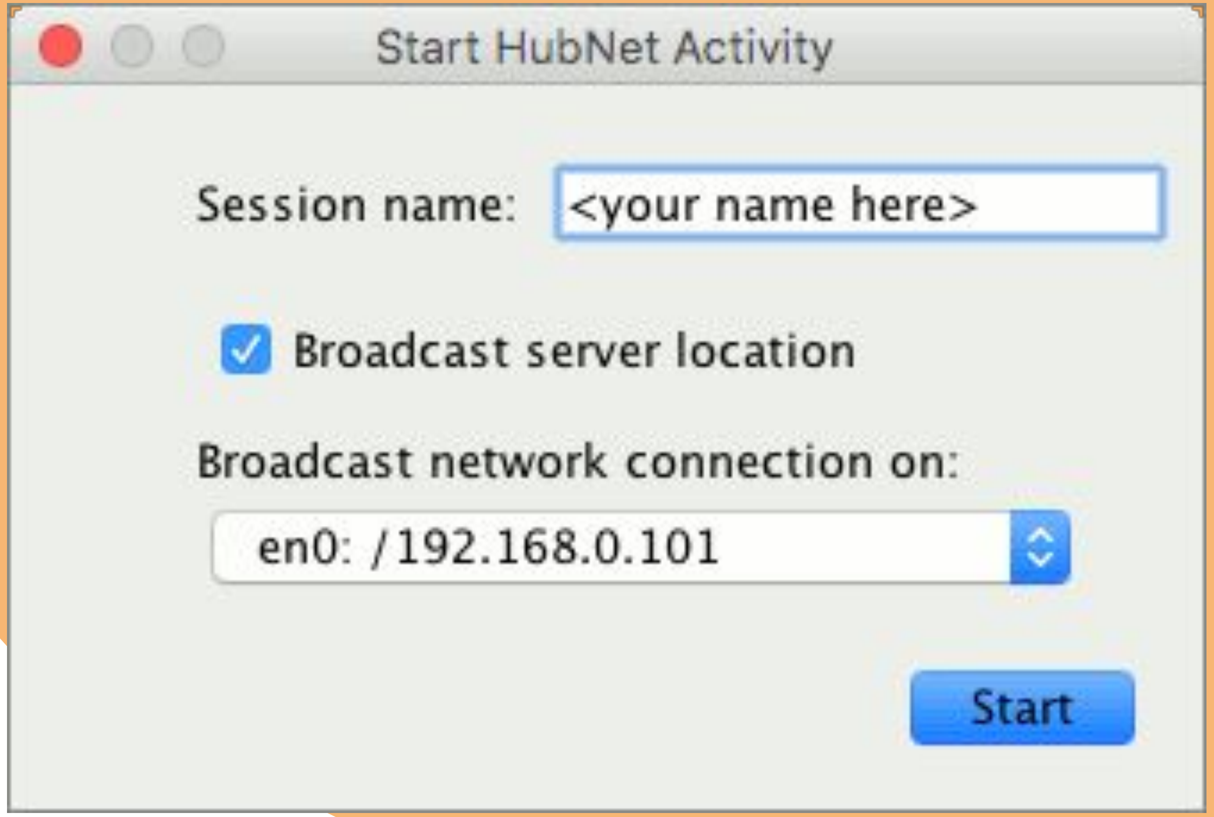
Test on 500.000 agents (Boids example)



Web Based Simulation (WBS)

Exploit Web Based Technologies

- No platform dependencies;
- Test Framework without installing something;
- E-Learning, example: **HubNet**



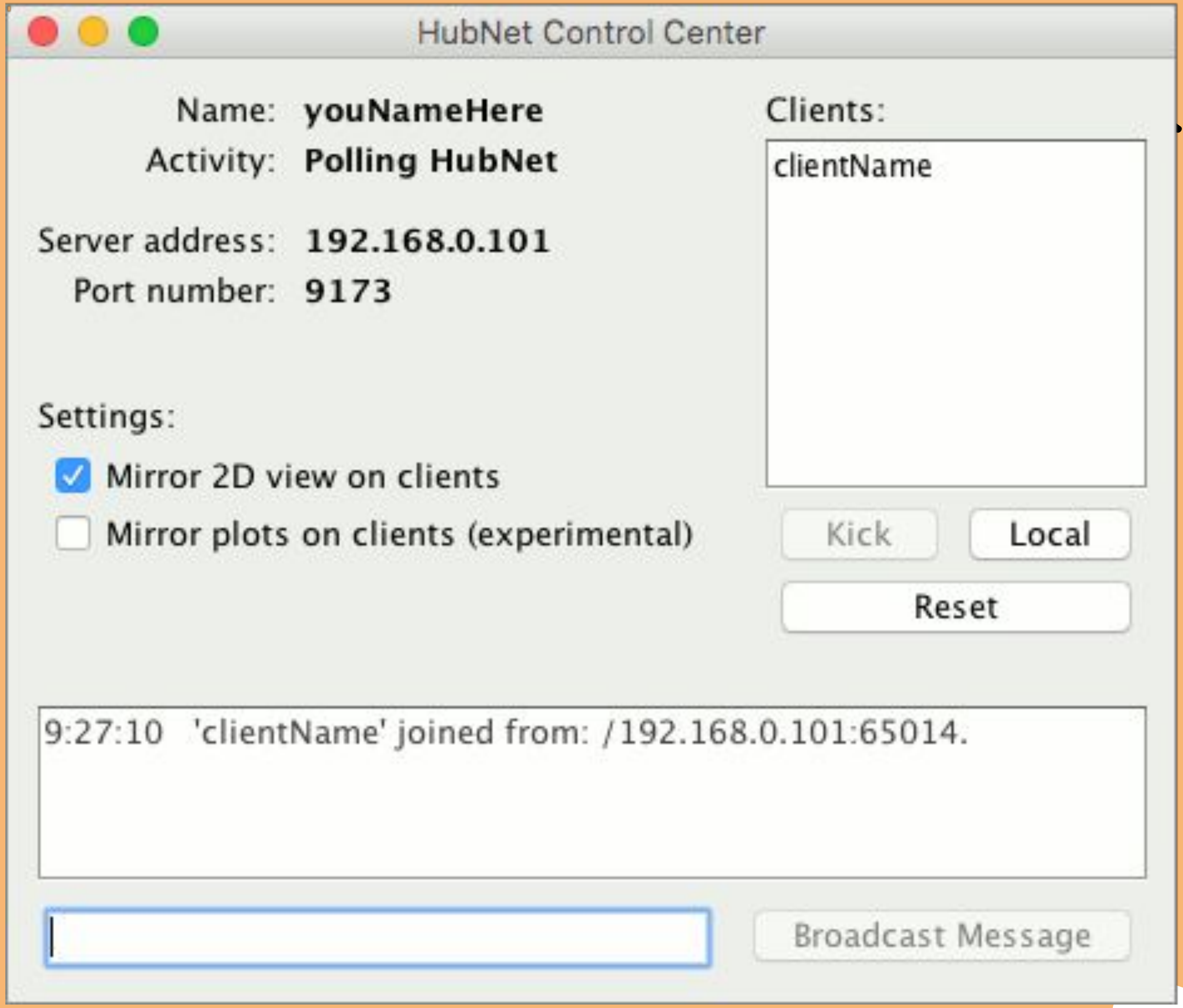
Start HubNet Activity

Session name: <your name here>

☑ Broadcast server location

Broadcast network connection on:

en0: /192.168.0.101



HubNet Control Center

Name: youNameHere
Activity: Polling HubNet

Server address: 192.168.0.101
Port number: 9173

Settings:

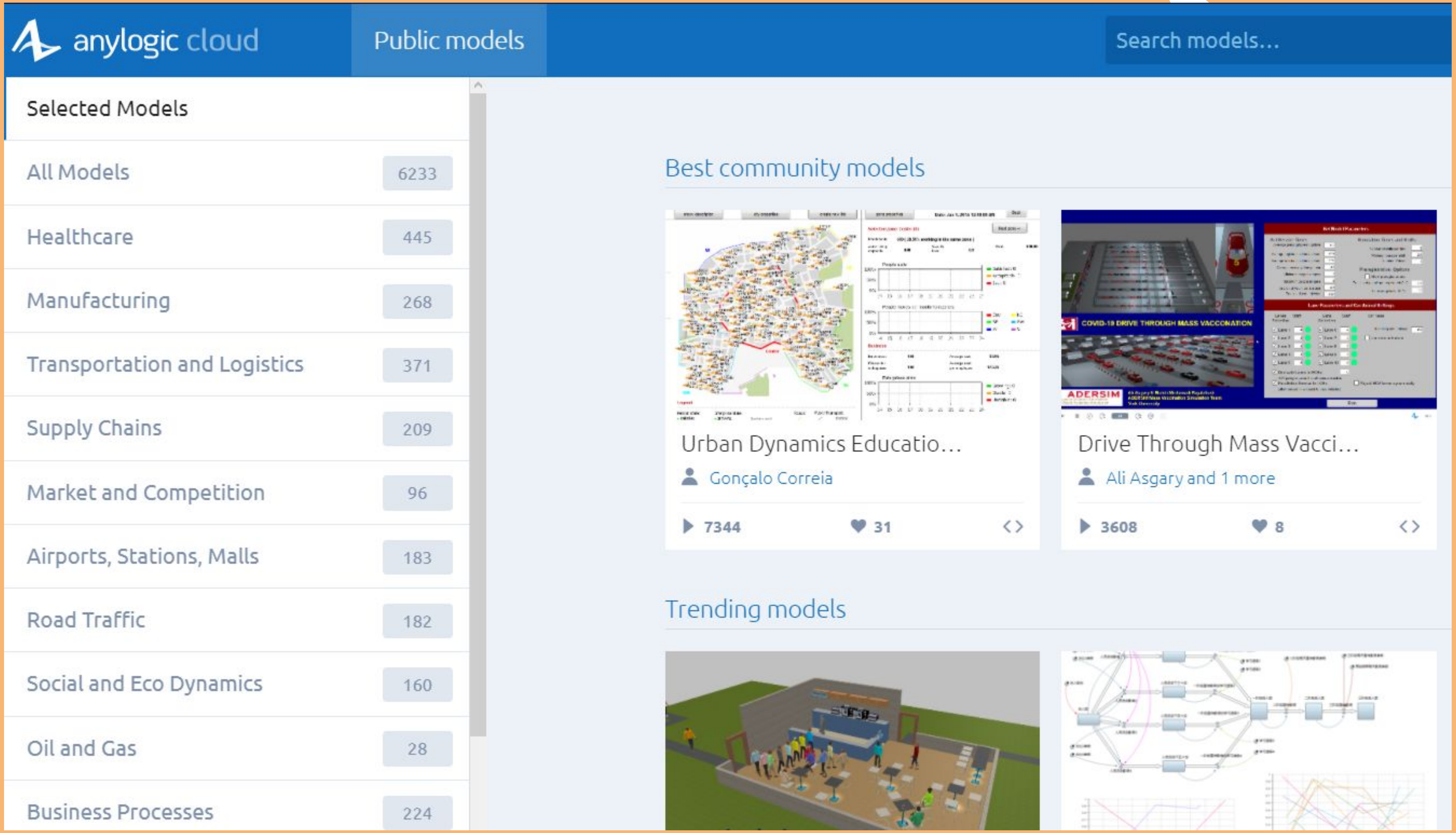
☑ Mirror 2D view on clients
□ Mirror plots on clients (experimental)

Clients:

clientName

```
9:27:10   'clientName' joined from: /192.168.0.101:65014.
```

Anylogic WBS:
Community Repo



anylogic cloud | Public models | Search models...

Selected Models	
All Models	6233
Healthcare	445
Manufacturing	268
Transportation and Logistics	371
Supply Chains	209
Market and Competition	96
Airports, Stations, Malls	183
Road Traffic	182
Social and Eco Dynamics	160
Oil and Gas	28
Business Processes	224

Best community models

Urban Dynamics Educatio...
Gonçalo Correia
▶ 7344 ♥ 31 < >

Drive Through Mass Vacci...
Ali Asgary and 1 more
▶ 3608 ♥ 8 < >

Trending models

Rust-AB

ISISLab
Framework

Current DevTeam

Team Coordinator: Spagnuolo Carmine

Caramante Pasquale

Parallelization, Benchmarks

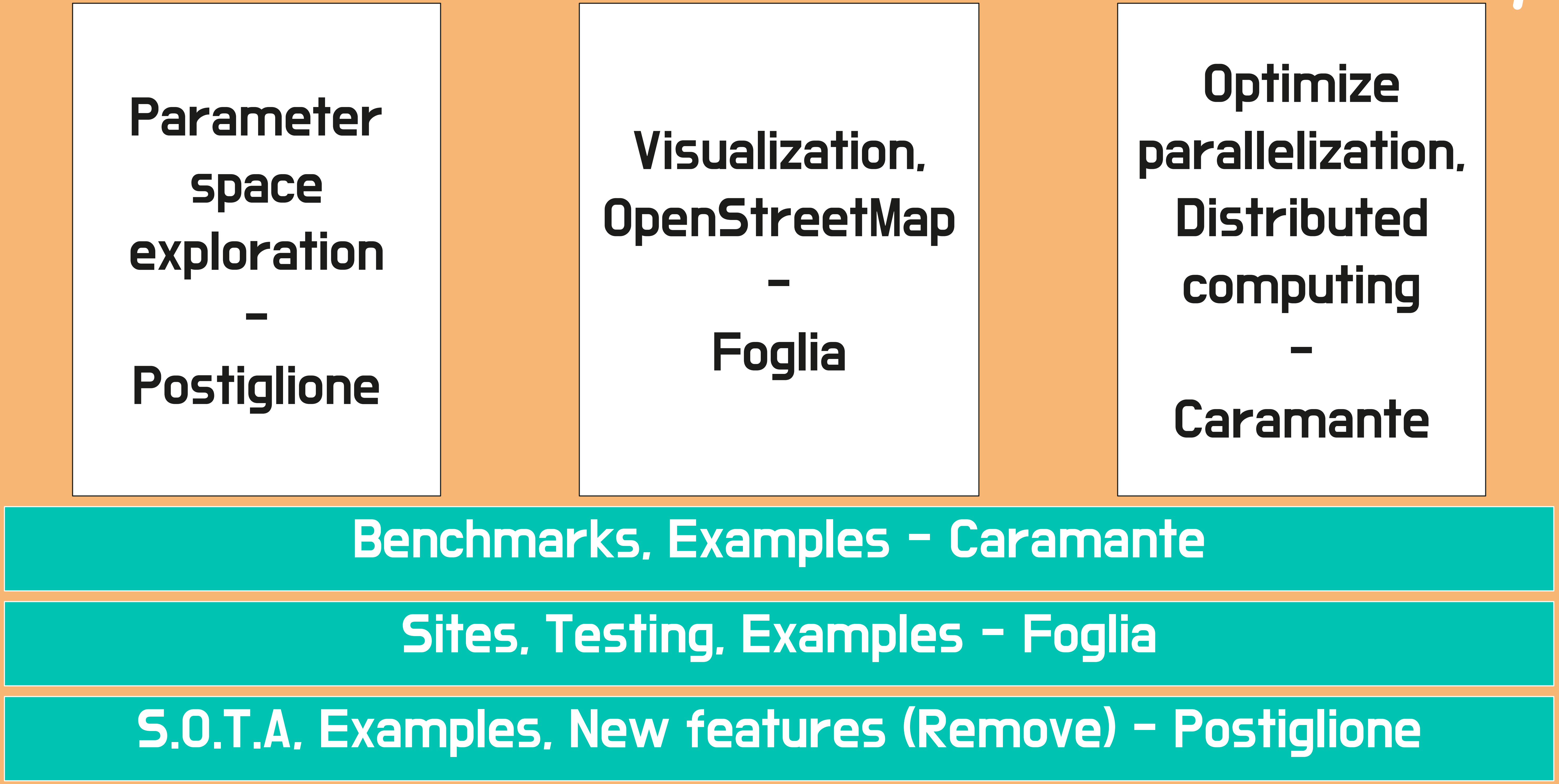
Foglia Francesco

Visualization, WebSite

Postiglione Luca

Parameter space exploration and optimization

Rust-AB: RoadMap



Parameter space exploration

-

Postiglione

Visualization, OpenStreetMap

-

Foglia

Optimize parallelization, Distributed computing

-

Caramante

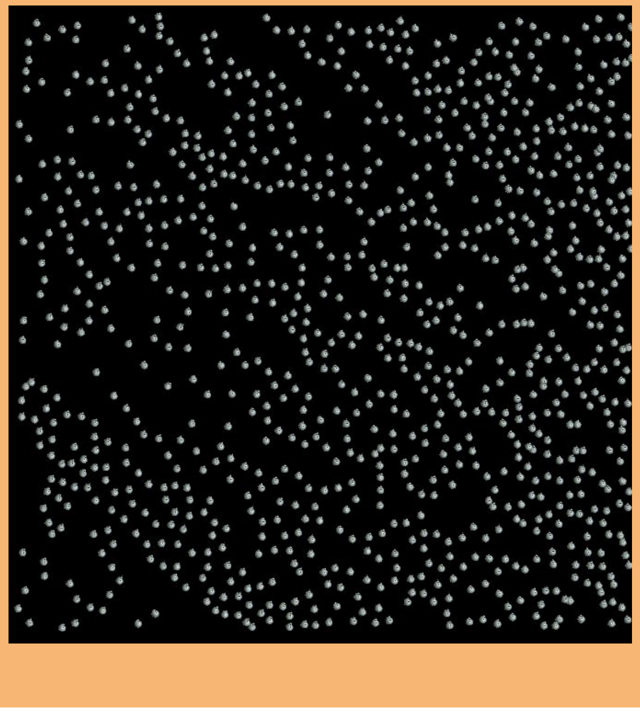
Benchmarks, Examples - Caramante

Sites, Testing, Examples - Foglia

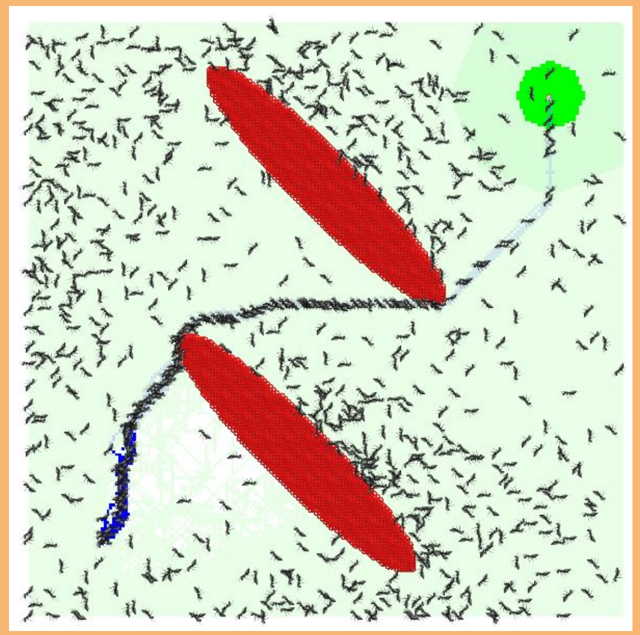
S.O.T.A, Examples, New features (Remove) - Postiglione

Models: Available & in progress

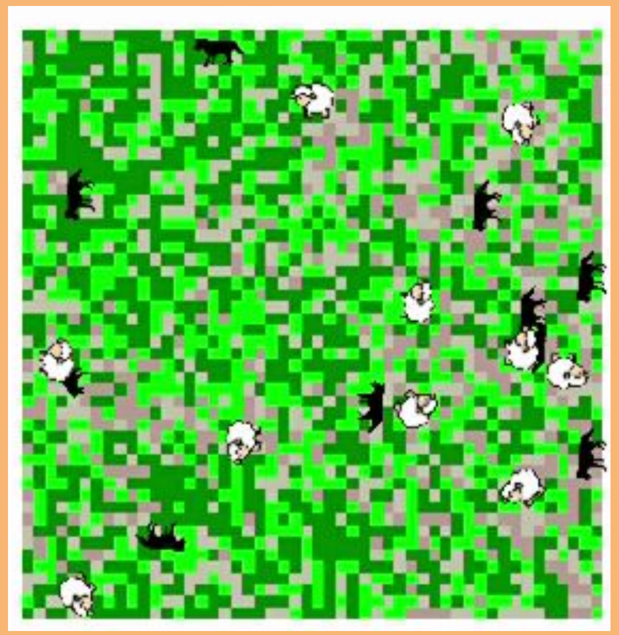
Boids



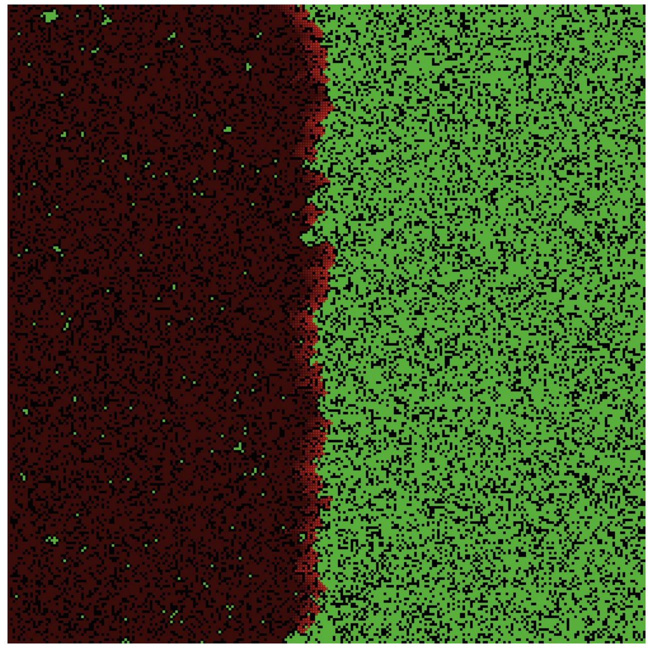
Ants Foraging



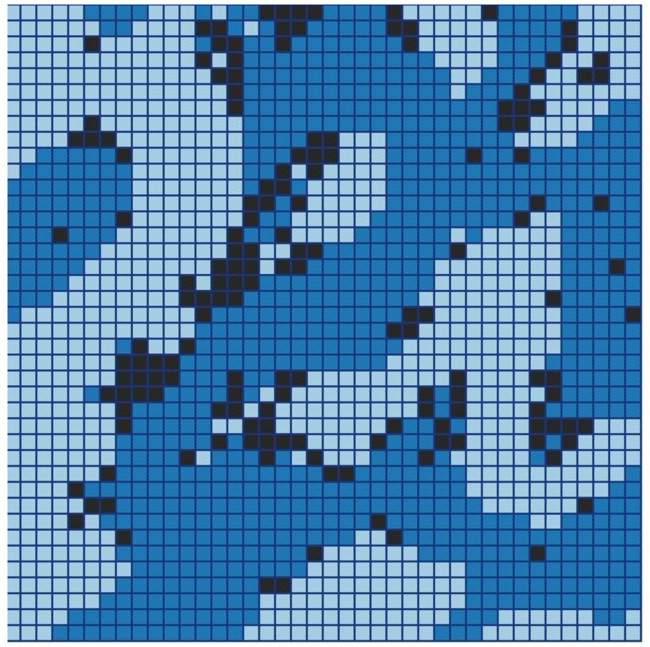
WolfSheepGrass



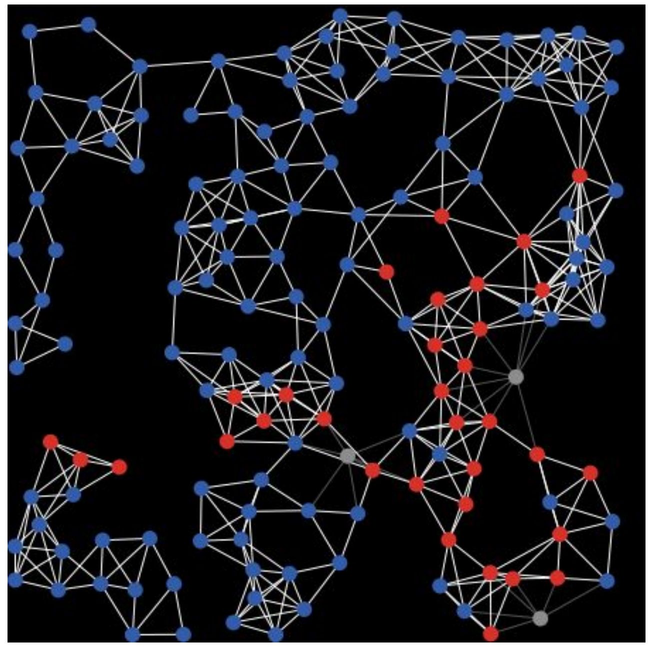
Forest Fire



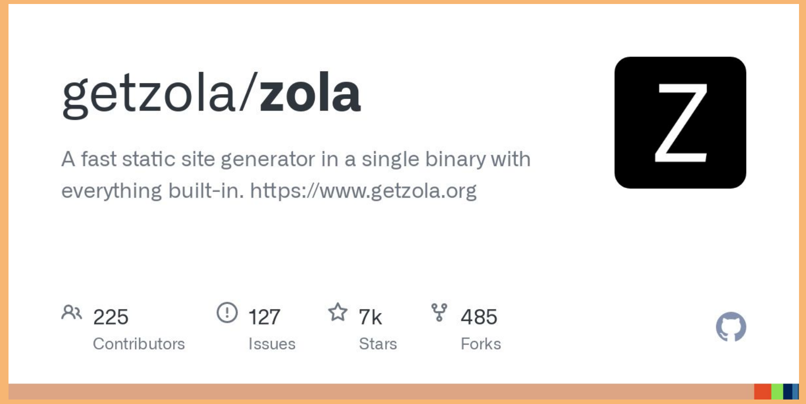
Schelling



Virus on a Network



Teammates are working on



getzola/**zola**

A fast static site generator in a single binary with everything built-in. https://www.getzola.org

225 Contributors · 127 Issues · 7k Stars · 485 Forks

01 WebSite

With documentation, examples and benchmarks

02 Dynamic Scheduling

Add/Remove agents from scheduler

03 New Graphic engine

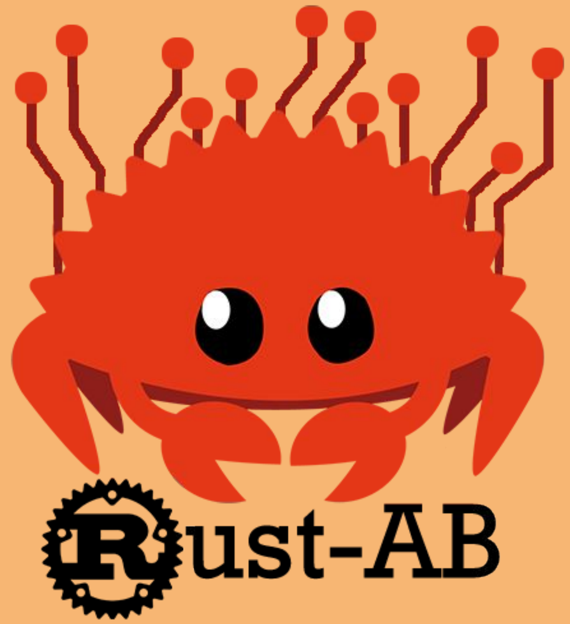
Previously based on **Amethyst**, now moved to **Bevy**

04 Benchmarks

Auto-update using github actions and publish result on the site

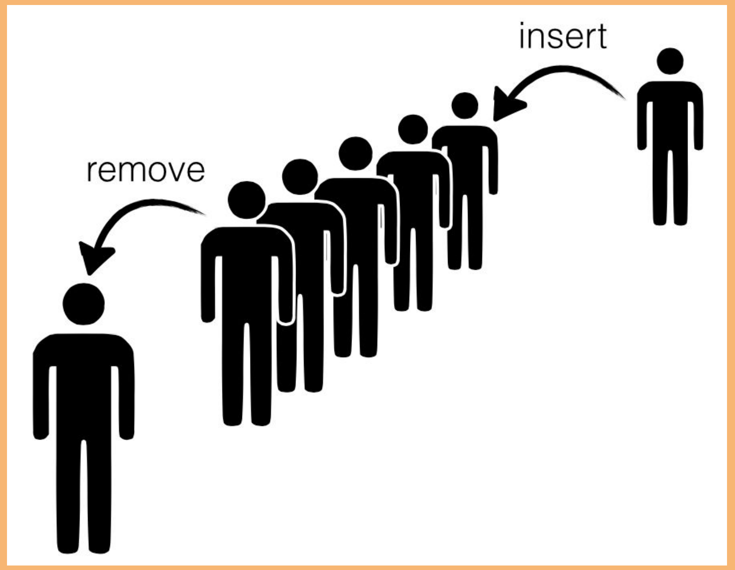
What I've Done

Kick-off



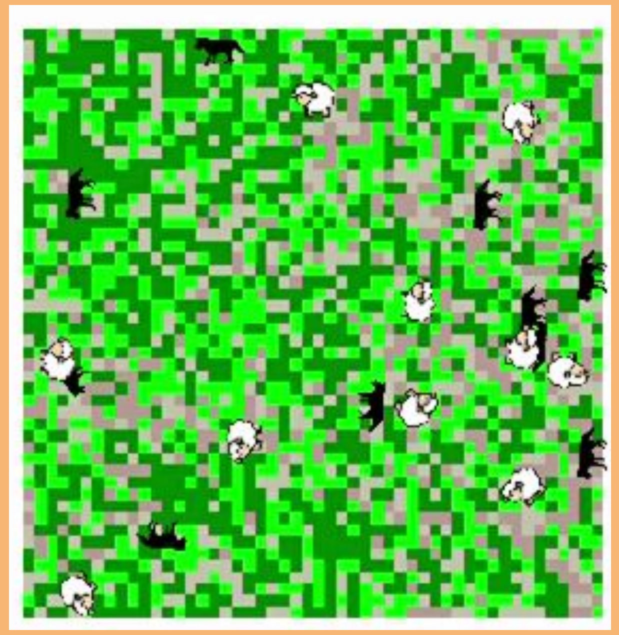
1. Learning **Rust**
2. Rust-AB analysis

Model Analysis

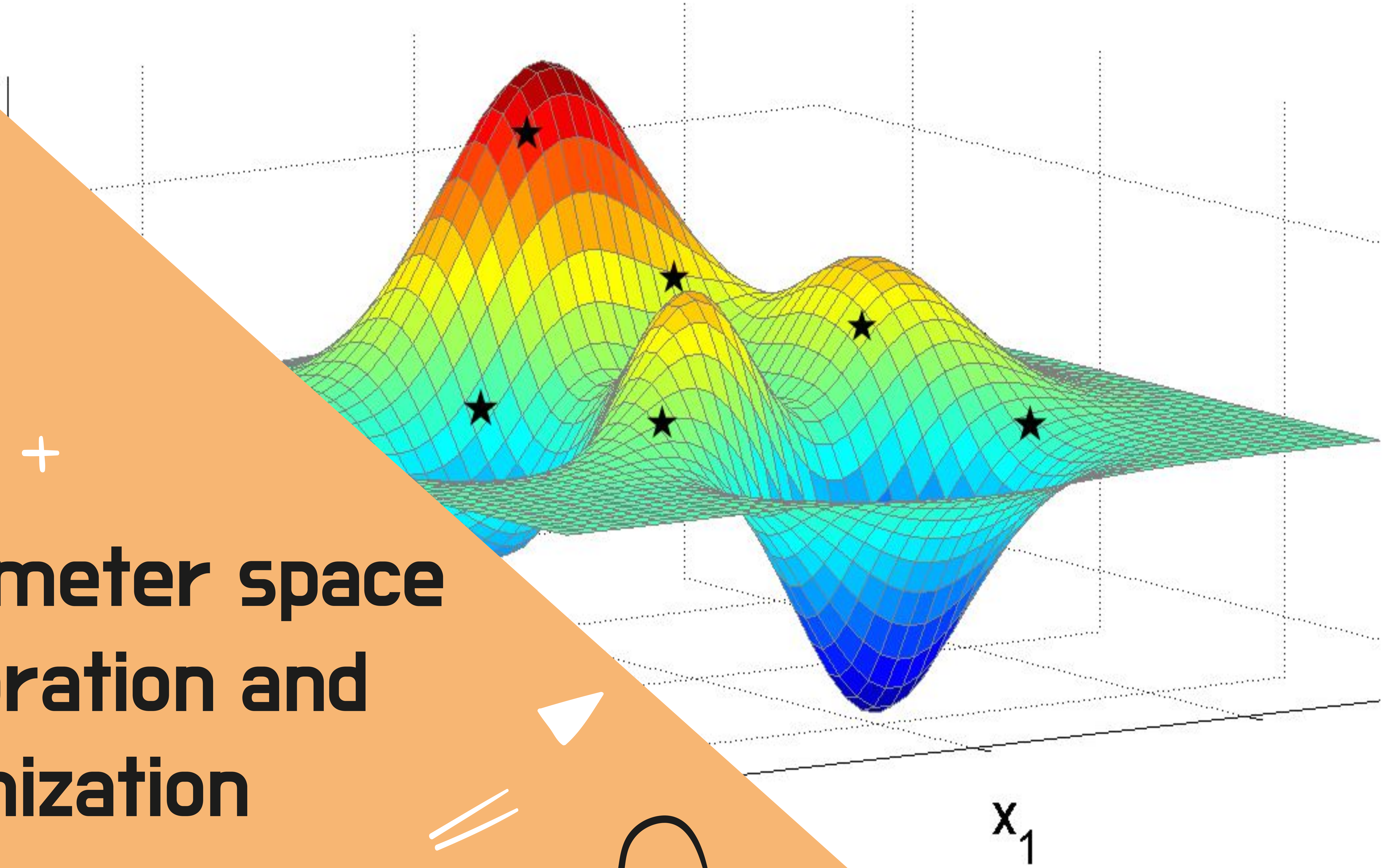


1. Defining features for WSG
2. Future problems of the model

WolfSheepGrass



1. Model Definition
2. "Remove from grid"
3. Bug Detection



Parameter space exploration and optimization

Tuning parameters

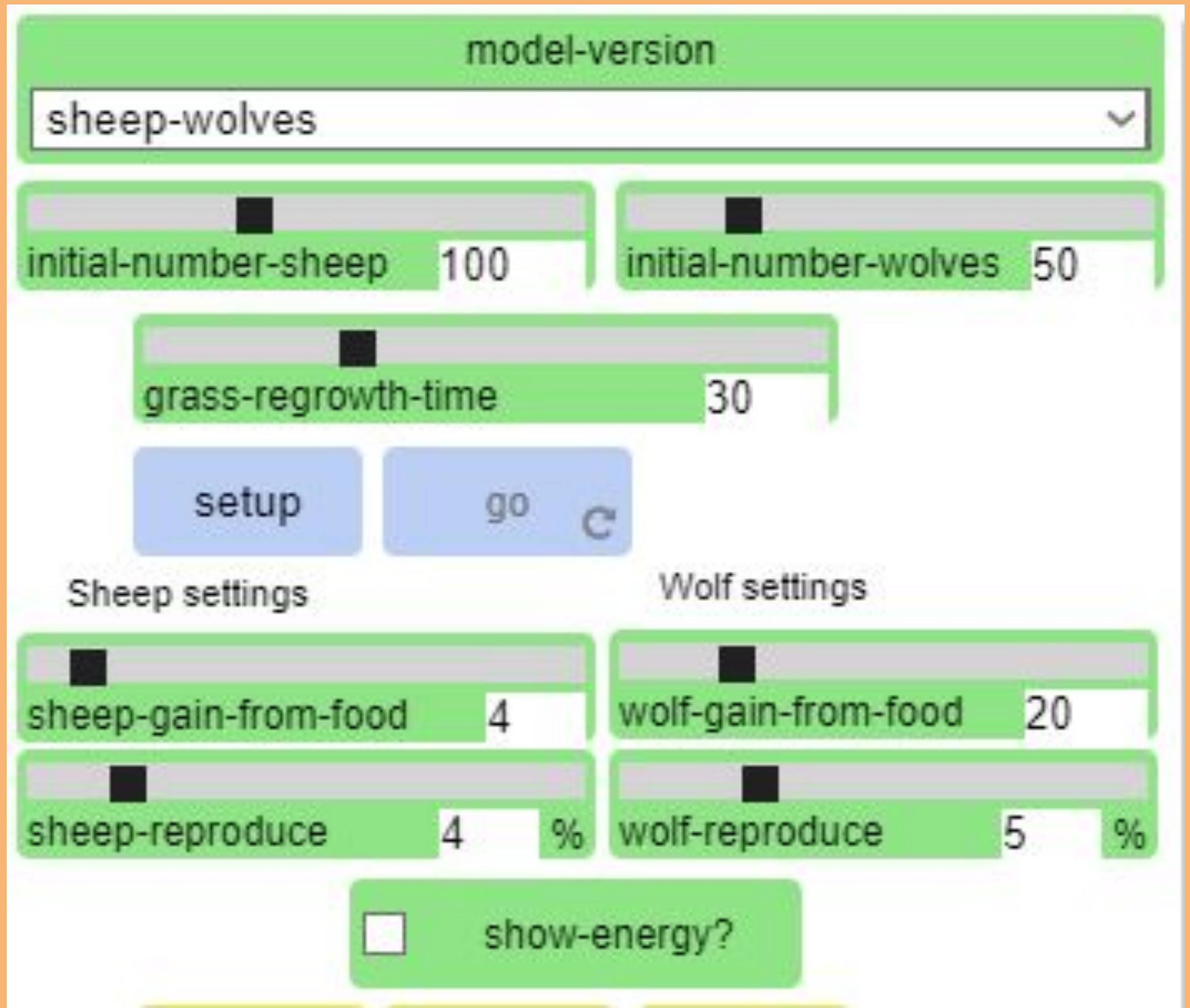
Manual testing to choose parameters is possible with the simplest models, with discrete domain.

But with real numbers?

jMetaL

Some tool is required to automatize this process

We want "real" simulation



model-version

sheep-wolves

initial-number-sheep 100

initial-number-wolves 50

grass-regrowth-time 30

Sheep settings

sheep-gain-from-food 4

sheep-reproduce 4 %

Wolf settings

wolf-gain-from-food 20

wolf-reproduce 5 %

□ show-energy?

The objective of AB simulation is reproduce better as possible what happened in real system.

Change values means change simulation results.

Integrated in a framework

MASON

Completely based on **Java**

NetLogo

It's a **programming language** written in Java/Scala

Repast

Provides also an **HPC** version

Agents.jl

Based on Julia. It's part of **JuliaDynamics**

FLAME GPU

Extension of another framework.

Mesa

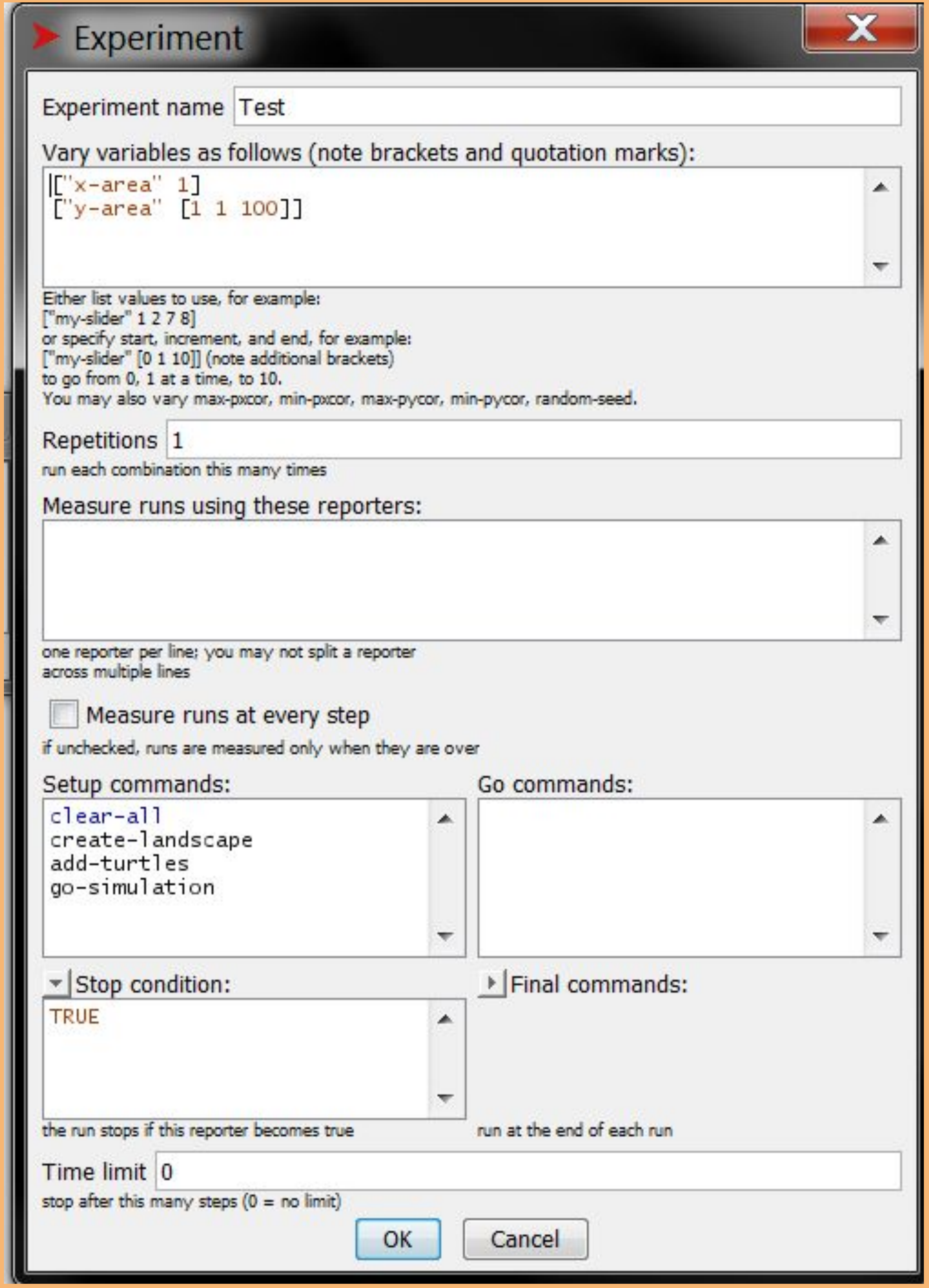
A **Python** framework

AnyLogic

Simulation software for **business**

BehaviorSpace

- Multiple Runs:
  - Multiple settings;
  - Repeat multiple time;
- Recording results;
- Generates dataset of results;
- “Explore” datasets:



Experiment

Experiment name Test

Vary variables as follows (note brackets and quotation marks):

```
["x-area" 1]
["y-area" [1 1 100]]
```

Either list values to use, for example:
["my-slider" 1 2 7 8]
or specify start, increment, and end, for example:
["my-slider" [0 1 10]] (note additional brackets)
to go from 0, 1 at a time, to 10.
You may also vary max-pxcor, min-pxcor, max-pycor, min-pycor, random-seed.

Repetitions 1

run each combination this many times

Measure runs using these reporters:

one reporter per line; you may not split a reporter across multiple lines

□ Measure runs at every step

if unchecked, runs are measured only when they are over

Setup commands:

```
clear-all
create-landscape
add-turtles
go-simulation
```

Go commands:

Stop condition:

```
TRUE
```

the run stops if this reporter becomes true

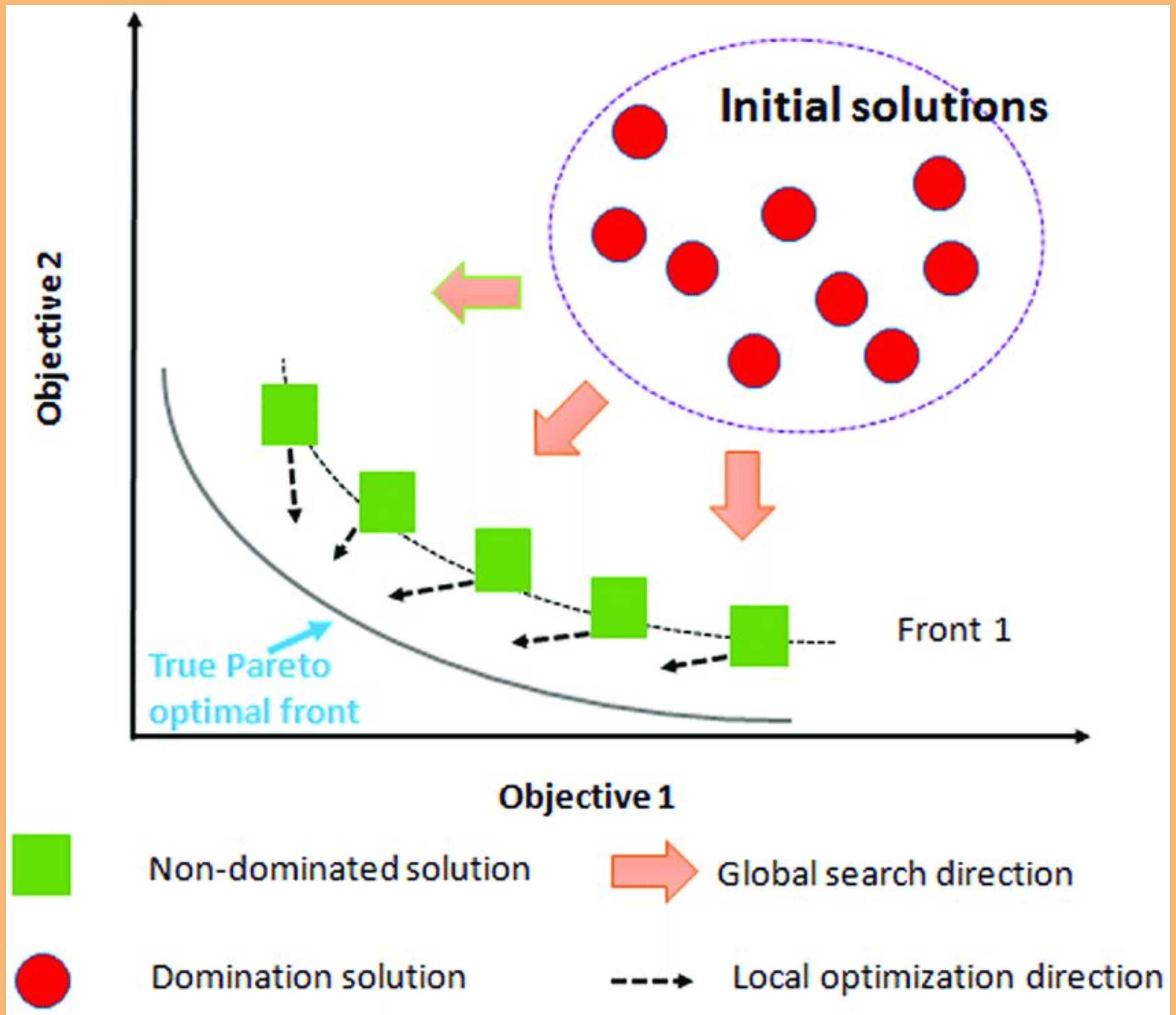
Final commands:

run at the end of each run

Time limit 0

stop after this many steps (0 = no limit)

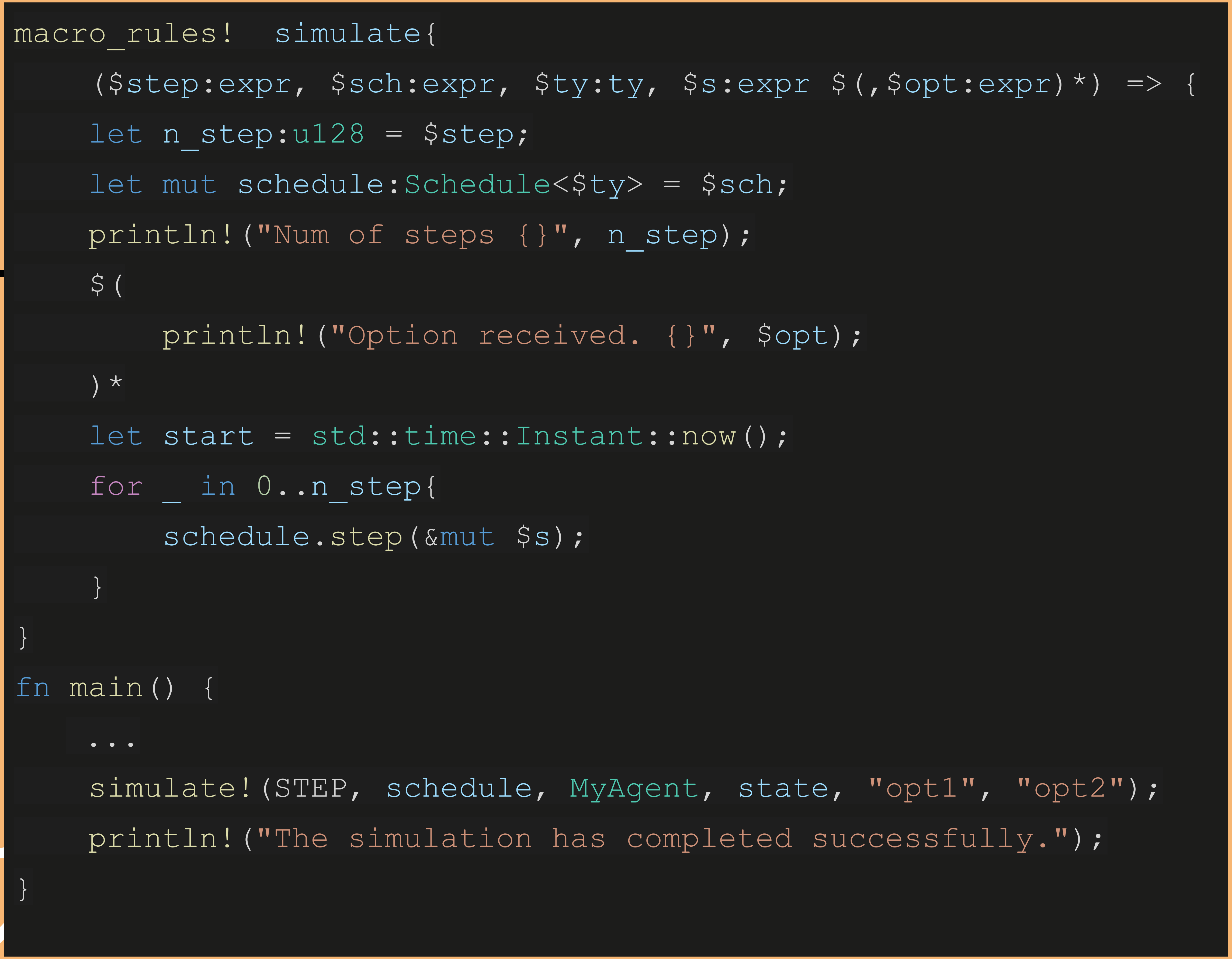
An optimization problem



Multi-objective optimization, also know as **Pareto optimization.**

I'll focus on Evolutionary algorithms, especially **NSGA-II** (**N**on-dominated **S**orting **G**enetic **A**lgorithm-**II**).

Metaprogramming in Rust



```rust
macro_rules!  simulate{
    ($step:expr, $sch:expr, $ty:ty, $s:expr $(,$opt:expr)*) => {
    let n_step:u128 = $step;
    let mut schedule:Schedule<$ty> = $sch;
    println!("Num of steps {}", n_step);
    $(
        println!("Option received. {}", $opt);
    )*
    let start = std::time::Instant::now();
    for _ in 0..n_step{
        schedule.step(&mut $s);
    }
}
fn main() {
    ...
    simulate!(STEP, schedule, MyAgent, state, "opt1", "opt2");
    println!("The simulation has completed successfully.");
}
```

"Code that generates code"

More than a function:

- Don't repeat yourself;
- Domain-specific languages;
- Variadic interfaces;

That's all falks!

(for now)

Do you are interested, curious or you have some question?
Join our meetings!

Rust-AB Team meeting:
Every Tuesday, 11.15 a.m.,
ISISLab Discord,
voice chat RoomTwo

Git Organization: Rust-AB
Email:
l.postiglione4@studenti.unisa.it