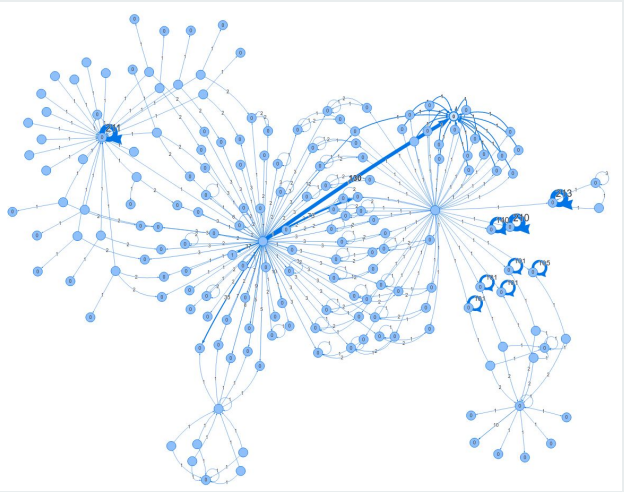
Scala(ble) High-Performance Computing

http://tiny.cc/scalable

Jakub Bujas & Dawid Dworak • 22.02.2018

lambda DAλS



Goals

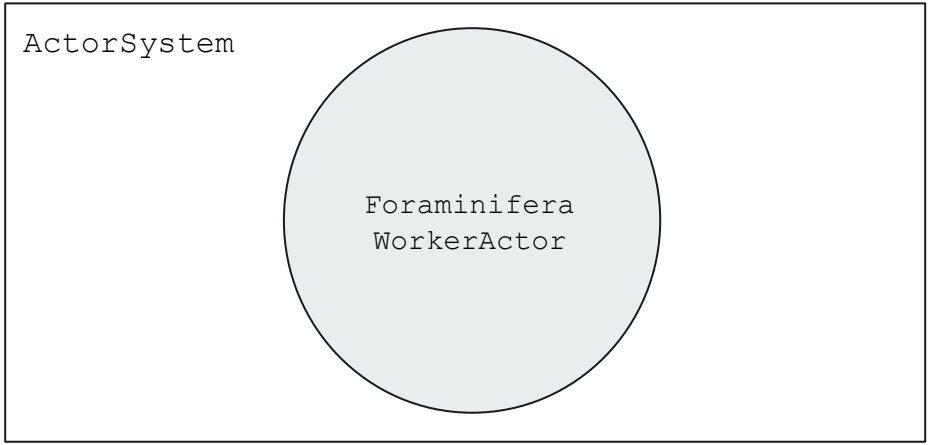
Simulate the life of a foraminiferal habitat:

- platform: HPC
- grid size > 10^9
- concurrency - actors responsible for computations related to a part of the grid
- distribution
- desynchronization mechanisms
- Hello ~~Mike~~ Martin?

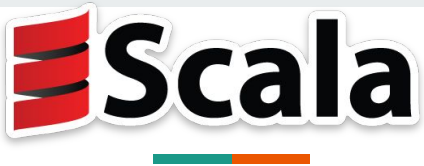
From your regular college dropouts to research contributors

First things first

"A single actor is no actor at all"



- single node
- sequential
- single biological model
- smell propagation mechanism
- 10^5 cells

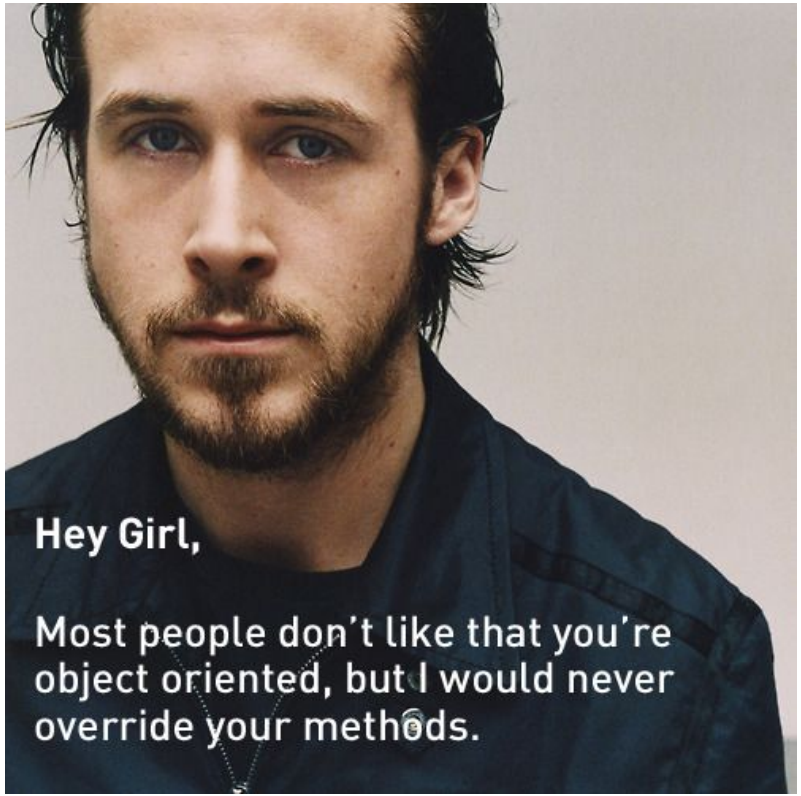


The good, old JVM:

- reliable performance
- enterprise-grade monitoring, profiling and deployment

The better, old (already) Scala:

- the name says it all
- expressivity
- flexibility
- poetic concurrency
- multi-platform - shameless plug: https://udash.io/

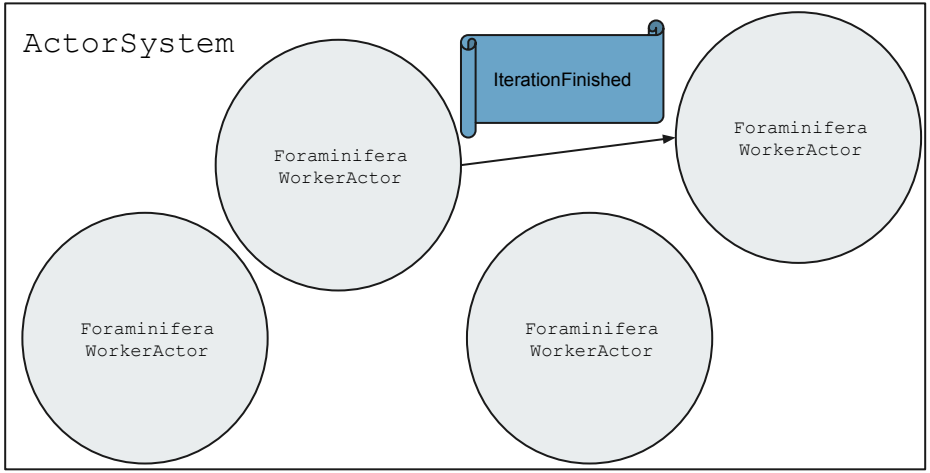


http://programmerryangosling.tumblr.com/

I can do things

"The possibility of incorrect results in the presence of unlucky timing is so important in concurrent programming that it has a name."

~Brian Goetz

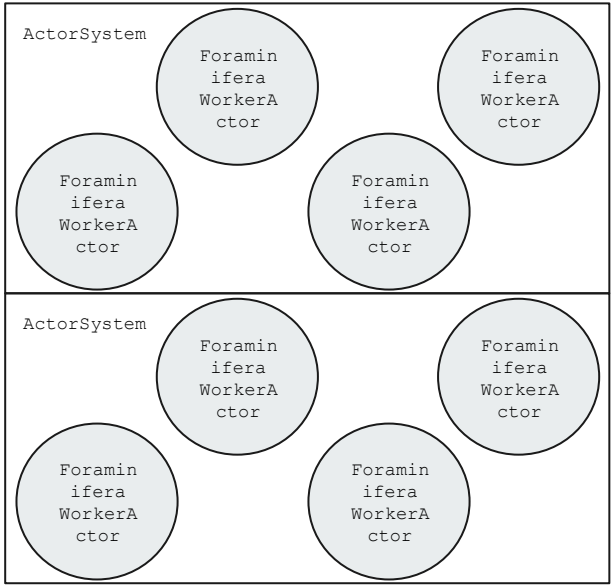


- concurrent
- desynchronised
- conflict resolution
- 10^6 cells

We can do things

"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable."

~Leslie Lamport



...

- distributed
- custom serialization
- 10^9 cells

Cluster sharding

```
val idExtractor: ShardRegion.IdExtractor = {
   case cmd: Command => (cmd.concertId, cmd)
}
```

IdExtractor allows ShardRegion to route commands to actors

```
val shardResolver: ShardRegion.ShardResolver =
   msg => msg match {
      case cmd: Command => hash(cmd.concert)
   }
```

ShardResolver assigns new actors to shards

```
ClusterSharding(system).start(
 typeName = "Concert",
 entryProps = Some(ConcertActor.props()),
 idExtractor = ConcertActor.idExtractor,
 shardResolver = ConcertActor.shardResolver)
```

Initialize ClusterSharding extension

```
val concertRegion: ActorRef = ClusterSharding(system).shardRegion("Concert")

concertRegion ! BuyTickets(concertId = 123, user = "Sander", quantity = 1)
```

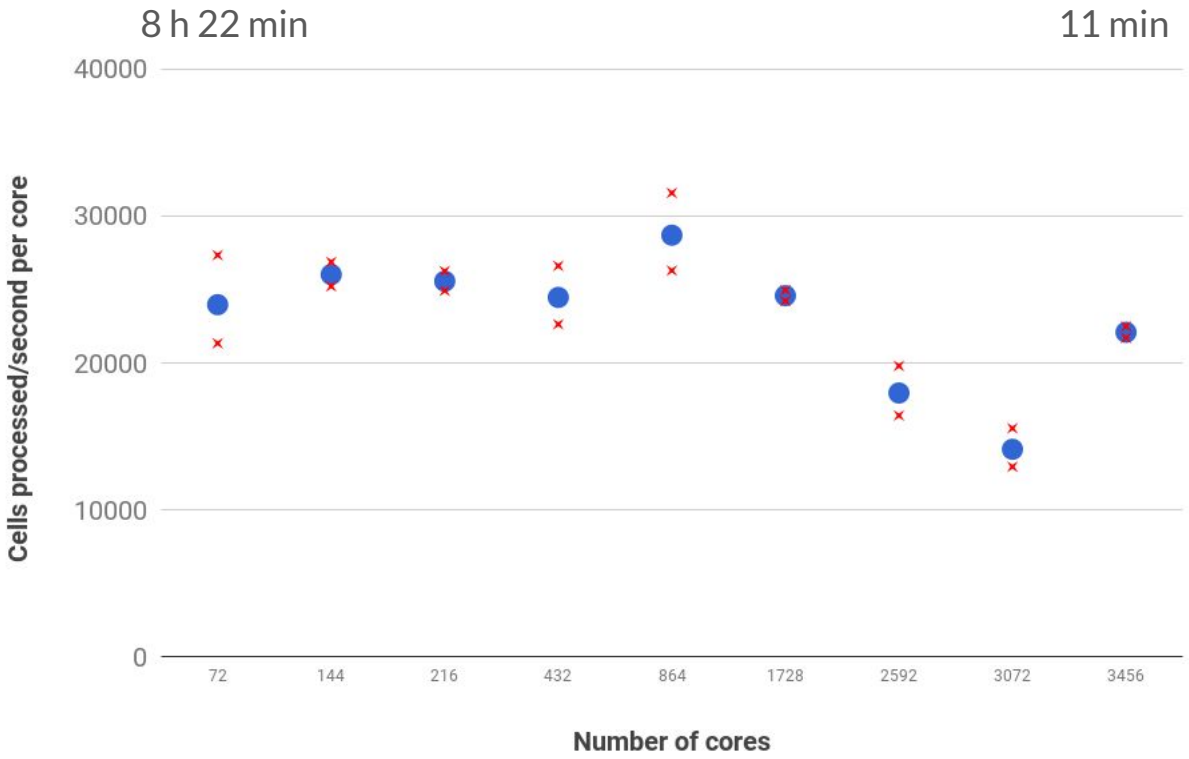
Xinuk Framework

- implement a model (cell types, behaviour) and run distributed
- user defined simulation metrics per iteration
- grid-level aggregation tools
- performance metrics at different levels of the stack (hardware, OS, JVM, concurrency, actors)
- interchangeable conflict resolution and signal emission
- soon (~2 months) open-source - follow @ddworak on GitHub

Performance - strong scaling

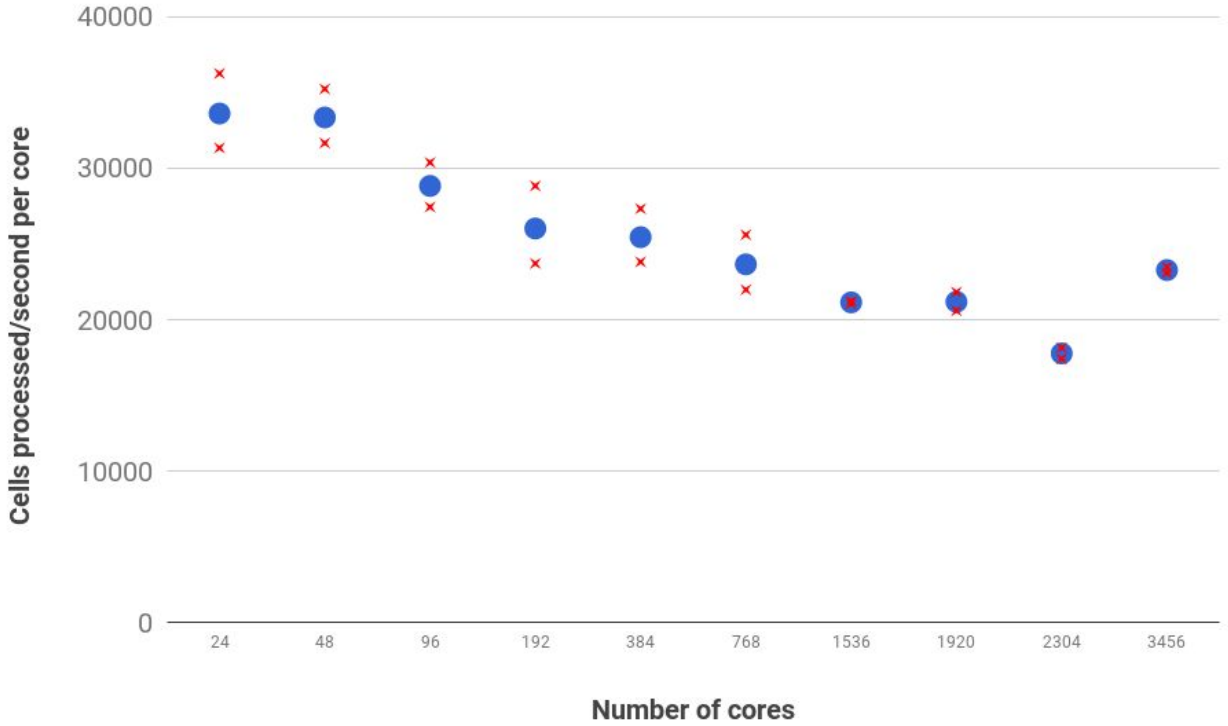
grid size 10^7

300GB



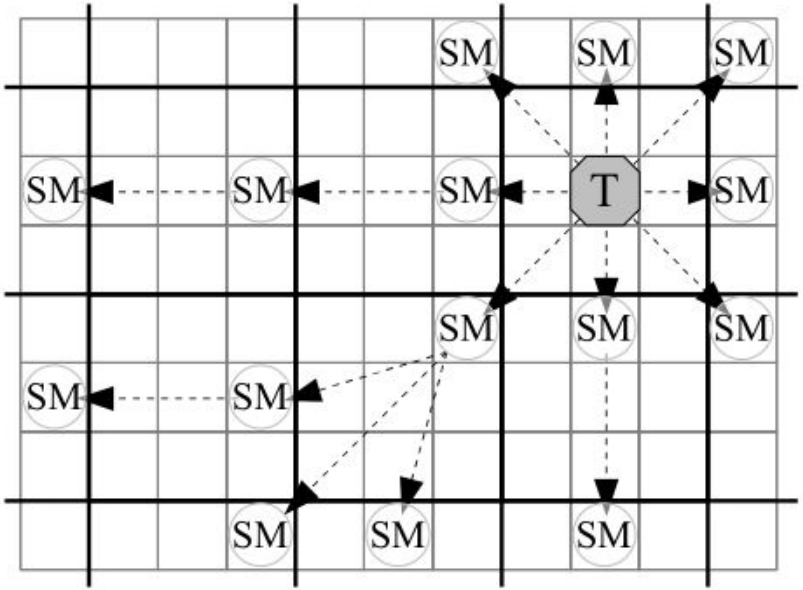
Performance - weak scaling

solution time: 45 minutes -> 64 minutes **+44%**
grid size: 2.16E+06 -> 3.11E+08 **+14398%**



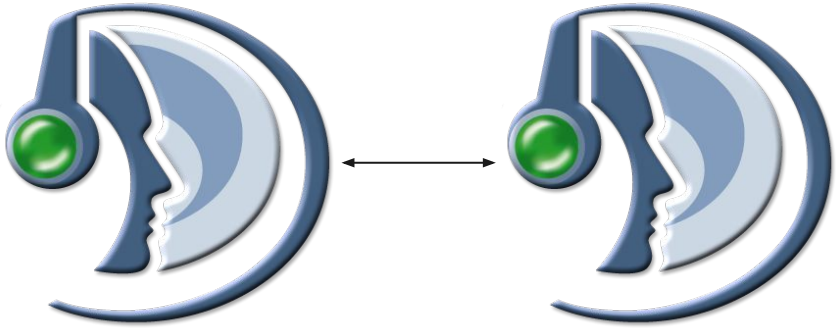
Implementation

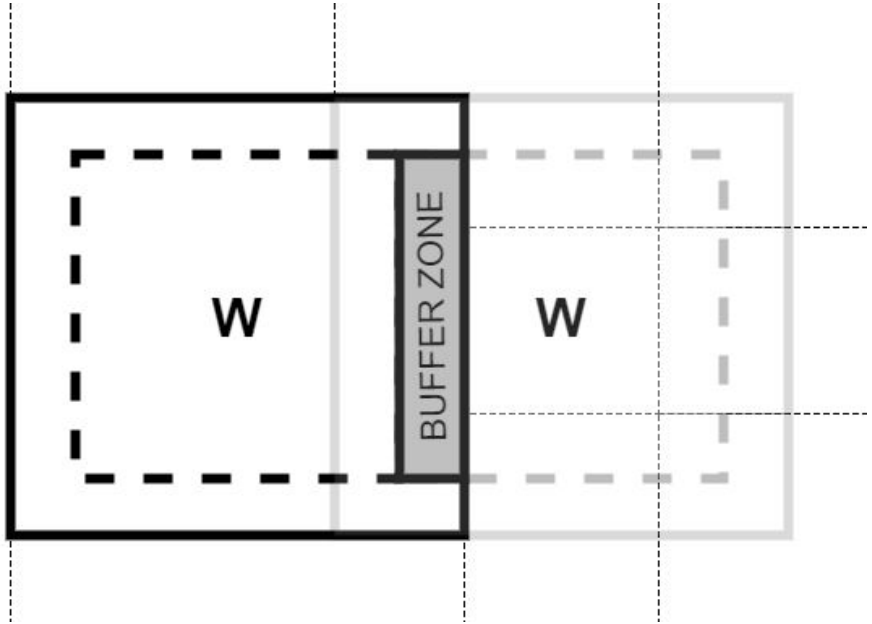
Smell propagation



T. Sosnicki and W. Turek and K. Cetnarowicz and M. Zabinska, 2013, Dynamic assignment of tasks to mobile robots in presence of obstacles, 18th Int. Conf. on Methods Models in Automation Robotics (MMAR), 538-543

Buffer Zones





The pseudo-code of an example conflict resolution implementation for foraminiferal habitat simulation

```
Foreach incoming bufferCell(x,y):
  case:
      Foraminifera(incomingEnergy,
      incomingSignal) and workerCell(x,y) is
      Foraminifera(energy, signal) then
      workerCell(x,y) = Foraminifera(incomingEnergy
      + energy, incomingSignal + signal)
  case:
      Foraminifera(incomingEnergy,
      incomingSignal) and workerCell(x,y) is
      EmptyCell(signal) then
      workerCell(x,y) =
Foraminifera(incomingEnergy,
      incomingSignal + signal)
  case:
      Foraminifera(incomingEnergy,
      incomingSignal) and workerCell(x,y) is
      Algae(signal) then
      workerCell(x,y) = Foraminifera(incomingEnergy
      + algaeEnergeticCapacity, incomingSignal
      + signal)
```

Conflict Resolution

- Scala’s strong type system
- Fully customizable and configurable on runtime
- Pattern matching mechanism

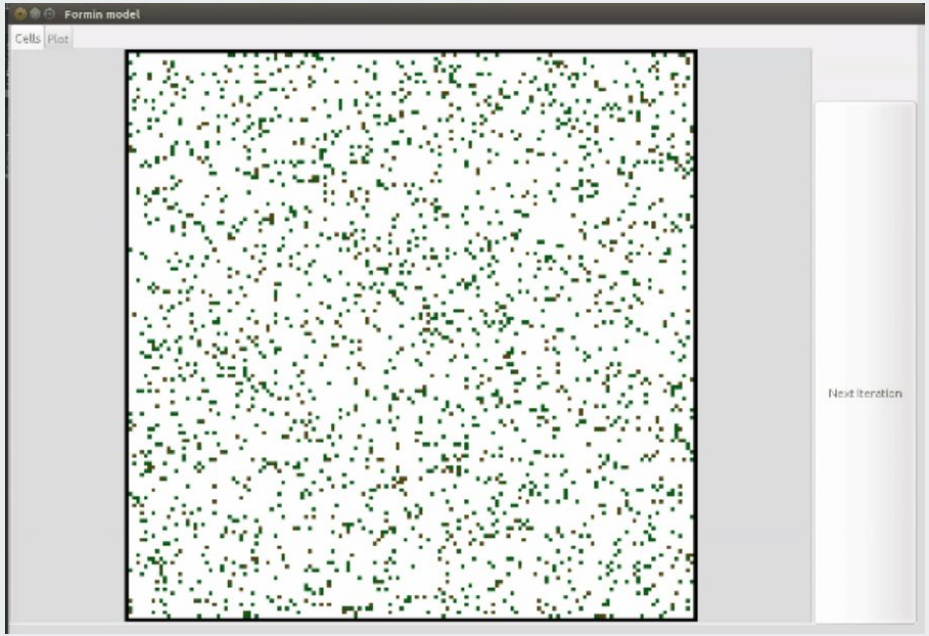
Experiments

Foraminifera

Small Creatures, Complicated Life

- First specimen-> foraminifera habitat simulation
- Multiples papers and previous research
- We needed something similar but simpler...

First 150 iterations of the simulation

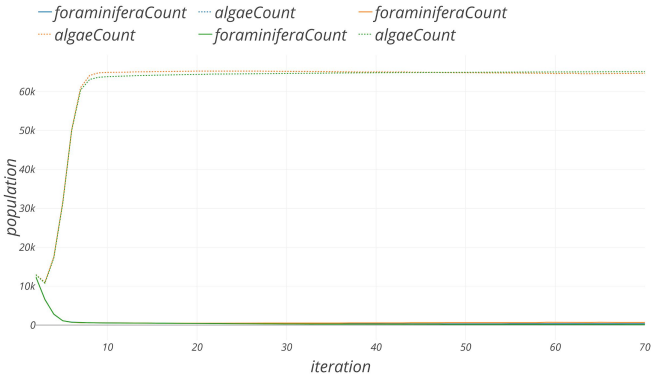


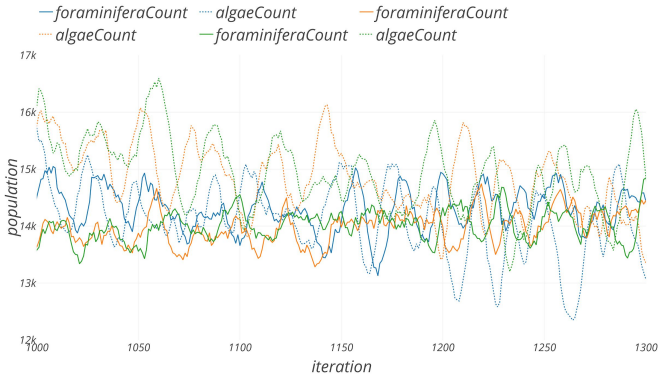
‘Foraminifera’ configuration

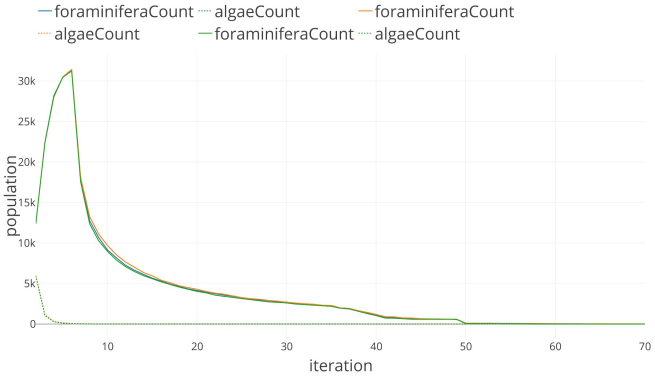
- FSE - Foraminifera Start Energy
- FRC - Foraminifera Reproduction Cost
- FRT - Foraminifera Reproduction Threshold
- FLAC - Foraminifera Life Activity Cost
- ARF - Algae Reproduction Frequency
- ARC - Algae Reproduction Cost
- SSR - Signal Speed Ratio

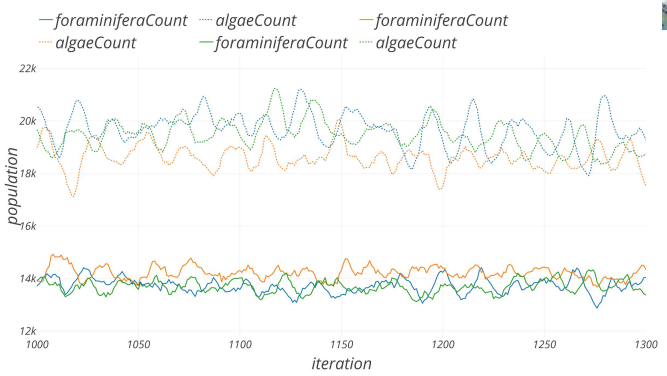
	FSE	FRC	FRT	FLAC	ARF	ARC	SSR
1	0.5	0.8	1.0	0.4	1.0	0.4	2.0
2	0.7	0.2	0.9	0.02	5.0	1.0	2.0
3	0.5	0.3	1.0	0.1	1.0	0.6	2.0
4	0.5	0.3	1.0	0.15	1.0	0.55	2.0

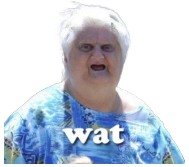
Experiments



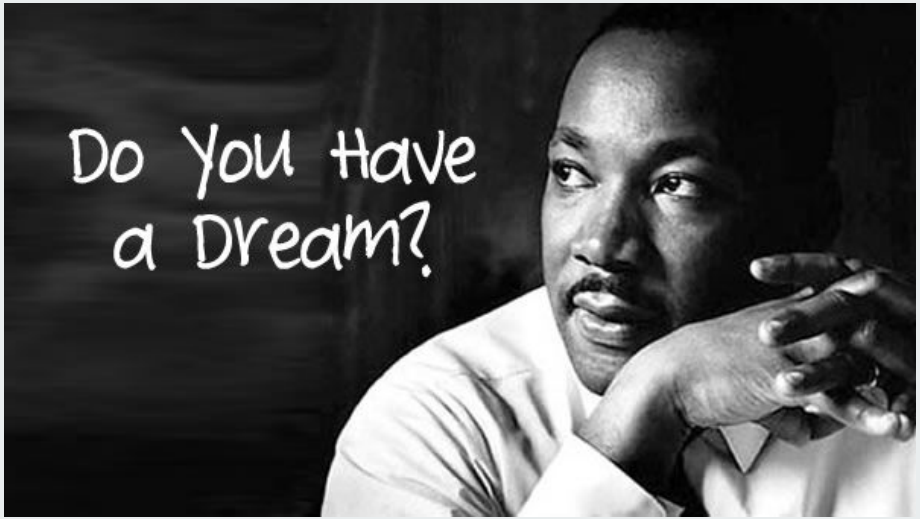








What's next



Yeah, we've got some!

Future work

- extend buffer zones
- configurable desynchronization
- smart hashing function for cluster sharding mechanism

Thanks!