Doing Some Interesting Experiments with ECJ and MASON Toolkit

Presenter: Khaled

Computational Optimization and Innovation (COIN) Laboratory
Michigan State University

talukde1@msu.edu

April 28, 2017

Disclaimer

- Introduction to some new toolkits for EC and other scientific experiments.
- Not about any of my current work.
- Does not give any specific idea, but might give you some interesting insights into different orthogonal topics.

Agenda: organized as follows –

- A brief introduction to ECJ toolkit.
- An example research with ECJ toolkit: Meta-EA
- A brief introduction to MASON framework.
- An example research with MASON framework: HiTAB + Robotics
- Gluing ECJ with MASON.
- A simple experiment to demonstrate ECJ + MASON.
- A more sophisticated scientific experiments with ECJ + MASON.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

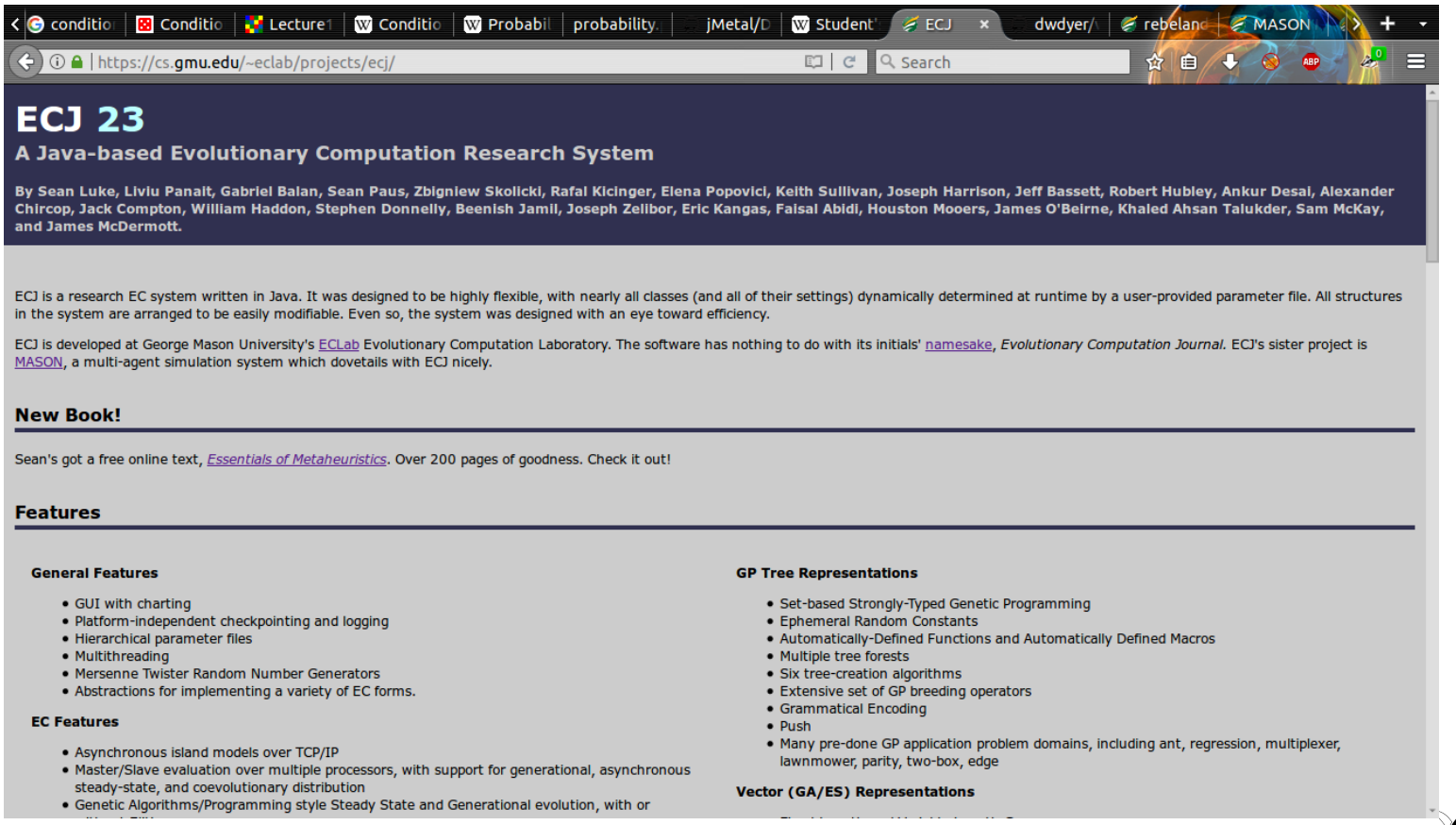
ECJ: **E**volutionary **C**omputation using **J**ava

- URL: `https://cs.gmu.edu/~eclab/projects/ecj/`
- Developed at the Evolutionary Computation Lab (ECLab) at George Mason University.
- First started as a Genetic Programming (GP) framework, and then extended.
- One of the most popular libraries cited/used by the EC community.
- Written in 100% Java.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

ECJ: Evolutionary Computation using Java



MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Features

- A fairly complete GP research system.
- Set-based GP, Strongly Typed GP, ADF, ADM, Multi-tree (forests), Tree generation algorithms, all sorts of GP operators, GE, Push, lots of examples and benchmark problem sets.
- Other EC Algorithms: GA, ES, DE, PSO, EMO and CMAES etc.
- EC models: Steady State, Island Models, Co-evolution, Massively parallel evaluation etc.
- All kinds of EC related operators.
- Fairly good collection of benchmark problems.
- EMO: only NSGA-II and SPEA2, ZDT and DTLZ problems, no performance measure facility.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Why/Why not ECJ?

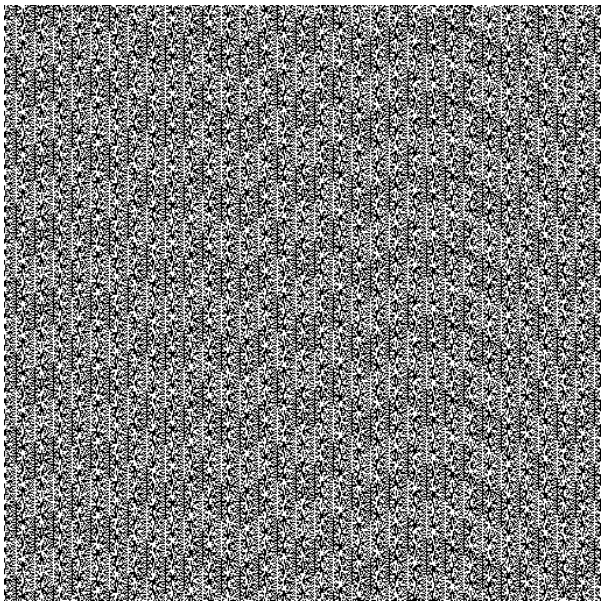


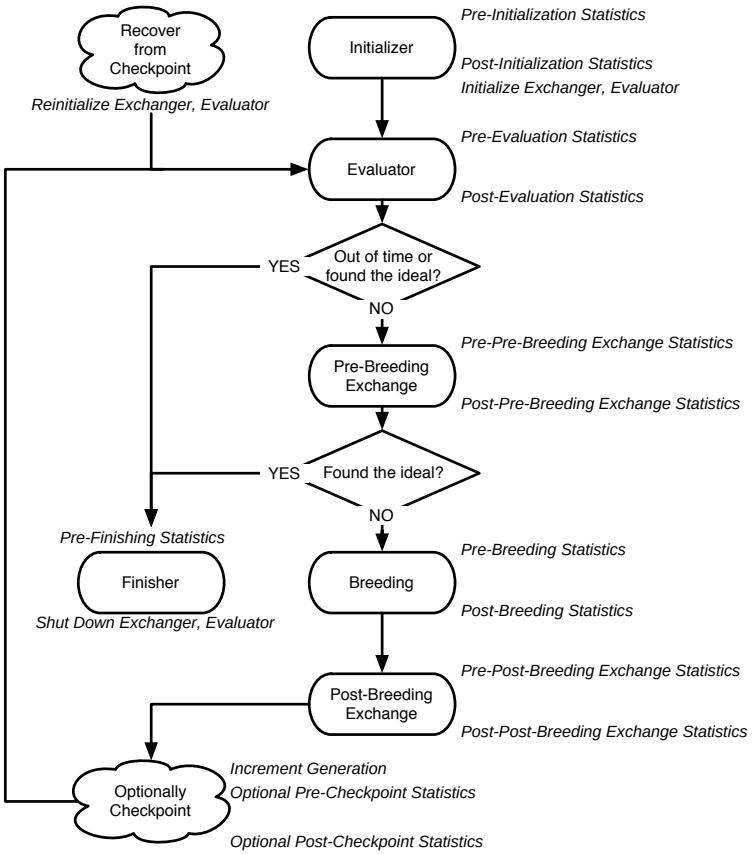
Figure: Pattern in `java.util.Random`

- No particular reason.
- Java random number generator (`java.util.Random`) used to have a bug (fixed in 2009). ECJ has it's own correctly implemented RNG (Mersenne Twister), very efficient and fast.
- Covers almost all aspects in EC in general. There are other frameworks (i.e. *watchmaker*), but not as vast as ECJ.
- Strong emphasis on efficient code. ($O(N^2)$ vs. $O(\log N)$) – *therefore, sometimes it's quite hard to understand the code, steep learning curve.*
- Time consuming to implement a simple concept, but very easy to do complicated experiments.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

ECJ Architecture



- Everything starts from a parameter file.
- Example run: `java ec.Evolve -file file.params`
- Objects are instantiated from parameter files.
- Parameters can be inherited.
- Parameters can form hierarchies.
- 300 page manual: `https://cs.gmu.edu/~eclab/projects/ecj/docs/manual/manual.pdf`
- **Show an example.**

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

An Example Experiment: Meta-EA

- Meta-EAs: finding the best EA parameters for a particular algorithm to solve a specific problem.
- Digression: Given a specific FE budget, and a parallel computing resource, when it is worthwhile to run a Meta-EA? and how should we run it?
- Is it possible to smooth out the EA parameter space? Testing one meta-individual (EA parameter settings) multiple times or just once?
- Evolve the meta-individuals according to the mean fitness from multiple test.
- Look for the strategy that will most likely to give the best solution to a particular problem.
- Massively parallel evaluation implemented using ECJ to run over a 128 node cluster, **14.2 million evolutionary runs** for different EAs and benchmark problems.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Meta-EA: Sample Result

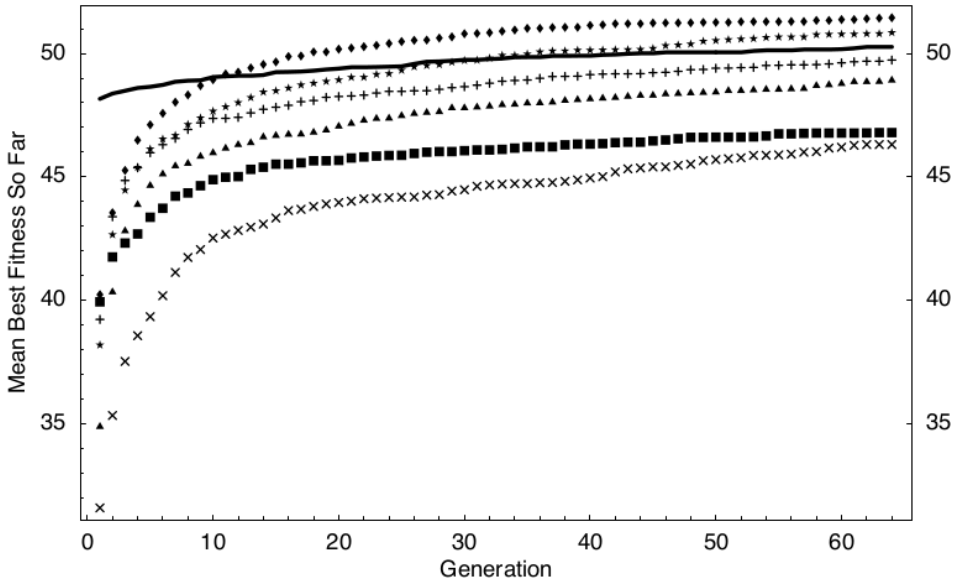


Figure 3: Meta-level mean best-fitness-so-far for the Lennard-Jones problem (DE, maximization). ♦ 1 Test ★ 2 Tests + 4 Tests ▲ 8 Tests × 16 Tests ■ Random Search — Best Discovered Setting

- Turns out that smoothing of the parameter space is not very useful if the budget is fixed.
- Specific example: 1, 2, 4, 8, 16 tests vs. 128, 64, 32, 16, 8 meta level pop. size.
- It's better to try 128 different parameter settings once than 8 parameter settings 8 times to get the "best solution discovering" parameter.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

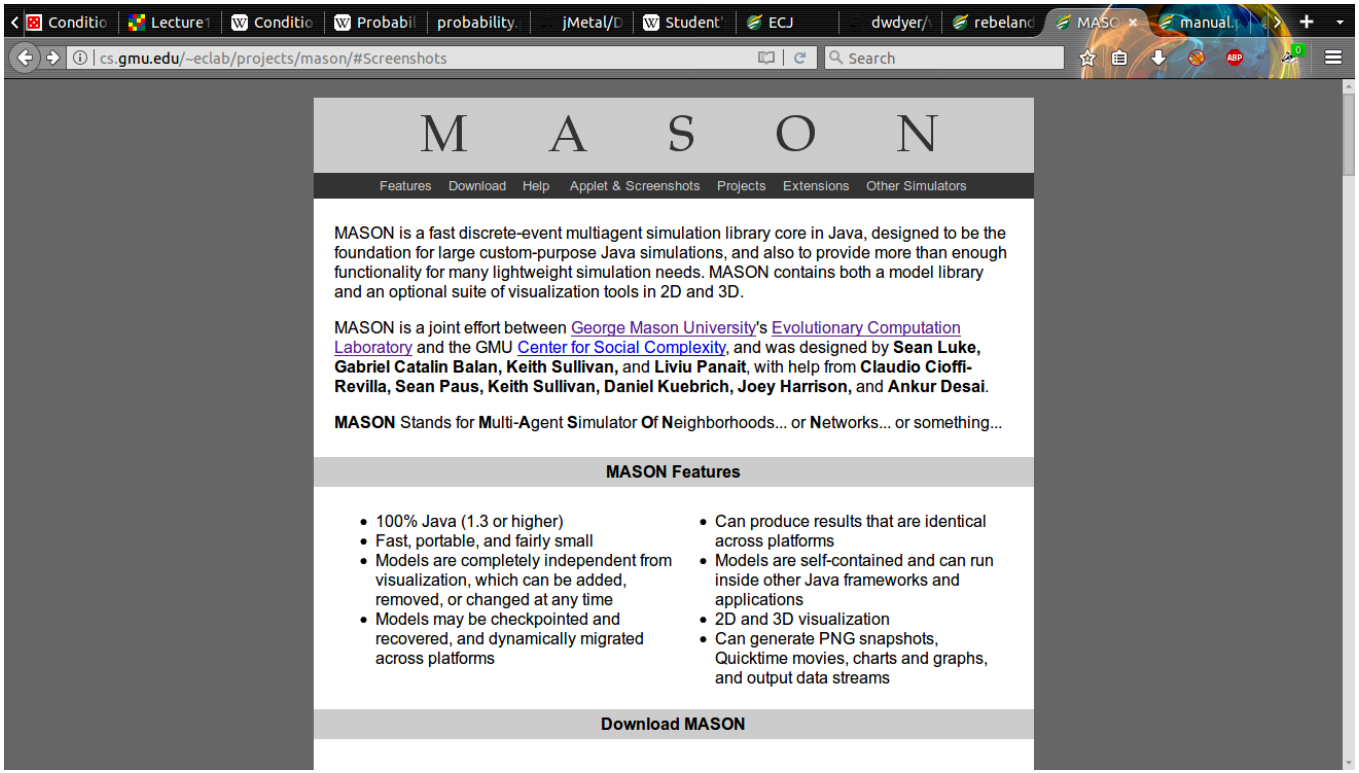
MASON: **M**ulti-**A**gent **S**imulator of **N**eighborhoods... or **N**etworks... or something...

- Fast discrete event simulator developed at GMU Autonomous Robotics Lab and Center for Social Complexity, Krasnow Institute of Advanced Studies.
- Initial objective was to do black-box simulation optimization with ECJ, but later came out as an independent project.
- More popular than ECJ, especially in the simulation science community.
- Also written in Java.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

MASON: **M**ulti-**A**gent **S**imulator of **N**eighborhoods... or **N**etworks... or something...



MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Features

- Can represent continuous, discrete, or hexagonal 2D, 3D, or Network data, and any combination of it.
- Full-fledge GUI for visualization, probing and monitoring simulation activities.
- The Console gives access to model data, plays, stops, pauses, and steps the simulation, checkpoints and recovers models, and performs other tasks.
- The user can also view and change per-object model data by selecting objects in the visualization displays and bringing up their inspectors.
- Very fast and efficiently written (like ECJ)
- It also comes with a 300 page manual (`http://cs.gmu.edu/~eclab/projects/mason/manual.pdf`)

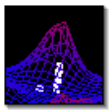
MICHIGAN STATE UNIVERSITY

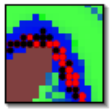
Computational Optimization and Innovation Laboratory
Michigan State University

Examples

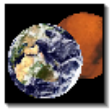
Click Thumbnail for Image


HeatBugs is a classic multiagent example popularized by the **Swarm** multiagent simulation toolkit.

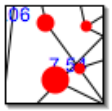

HeatBugs shown in wireframe 3D. Quicktime Movie.


Ants is an ant colony foraging simulation using two pheromones.


Flockers is an implementation of Craig Reynolds' **Boids** algorithm. 2000 flockers shown.

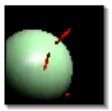

Solar System (Tutorial 6) is a simple (indeed simplistic) demo of planets orbiting the Sun.


HexaBugs is HeatBugs on a hex grid, inspired by the **RePast** simulation toolkit. (Shown zoomed in).


Balls and Bands (Tutorial 5) simulates Hooke's Law with balls connected by rubber bands of different strengths.


Mouse Traps and Ping Pong Balls When a ball hits a trap, it pops the ball back up, plus another ball. Quicktime Movie.


Woims is another **Boids** example. Quicktime Movie.


Woims in 3D shows the Woims in 3D space. Quicktime Movie.


Keepaway simulates a Keep-Away Soccer game. At present the robots have stupid "go to the ball and kick it" behaviors.


L-system is a deterministic, bracketed context-free L-system generator with basic turtle graphics.


Conway's Game of Life, a classic cellular automata, here as a simple demo seeded with the b-heptomino.


Cooperative Observation implements a k-means clustering method for cooperative observation of randomly-moving targets.


The LightCycles Game is a discretized version of the old game popularized by the movie Tron.


MAV is a basic platform for simulating simple Micro-air Vehicle behaviors.

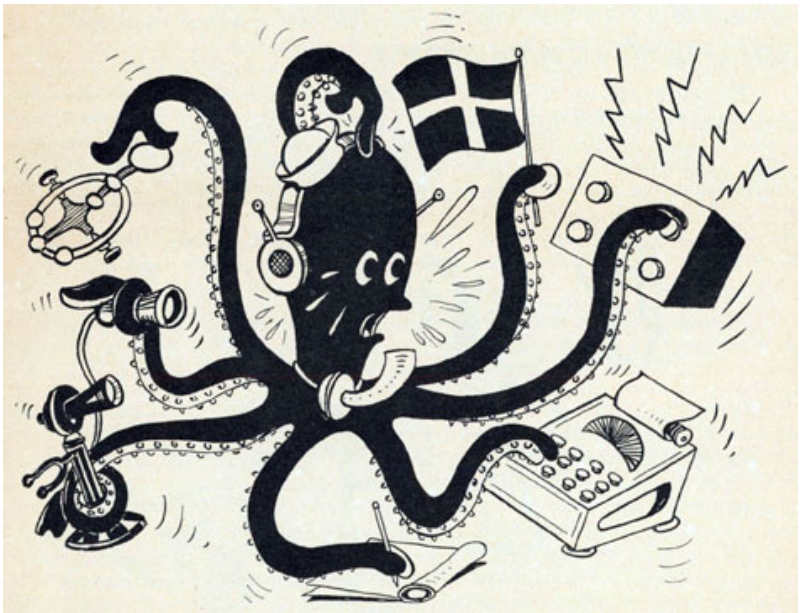

Bouncing Particles is a tutorial


Virus is a simple platform for

GAN STATE UNIVERSITY

COIN Computational Optimization and Innovation Laboratory Michigan State University

Gluing ECJ with MASON



- Scientists make **models**.
- Complicated models (i.e. ABM) requires a large number of **multiple inter-dependent parameters** to be tuned manually – prohibitive.
- **Can we automate this ?**

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Preliminaries

Model Parameter Types

- Known Parameters – whose settings already known for a fact
- Inferable Parameters – whose settings should be set in a certain way (*according to a particular theory*)
- Insensitive Parameters – over whose settings the model is expected to be insensitive
- **Unknown Parameters – on which an experimenter has no idea/control over.** (*most interesting ?*)

Goals

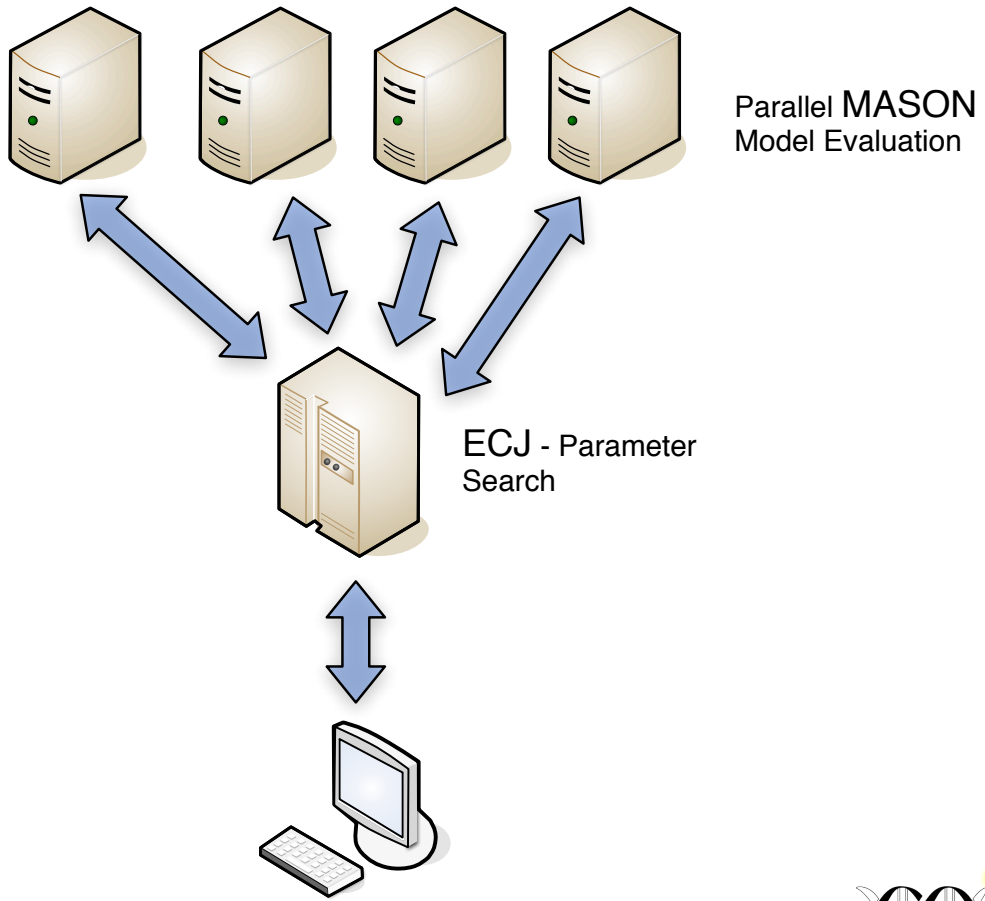
- To produce a certain kind of output which is predicted by a theory, or
- to match and validate against known real-world results.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Parallel Evolutionary Optimization – A Bird's-Eye View

- Extremely expensive – need to optimize the model by running it many times.
- Ought to be done in parallel.



MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

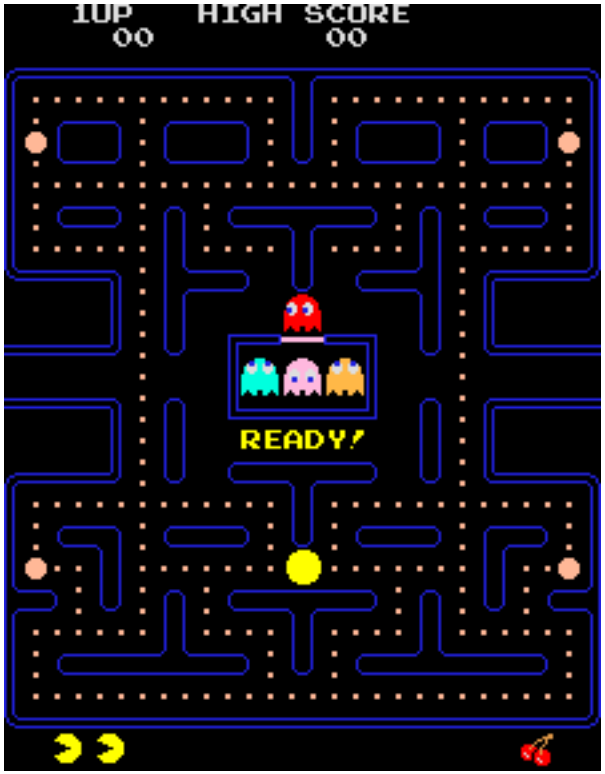
A Very Simple GP Experiment: Evolving PacMan Behavior

- MASON → **Behaviour** specifications
- ECJ → **Optimization** by **Evolution**
- MASON + ECJ → Evolving Optimized Behaviour
- **Target**: The game of PacMan

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

An Evolved Pac Behaviour



Show Demo

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

A More Sophisticated Example: RebeLand

- Built on top of MASON to simulate socio-economic instability.
- Developed by the Center for Social Complexity, Krasnow Institute of Advanced Study, GMU and Smithsonian Institution.
- Simulate a government policies according to canonical political science and theory from economics
- Experimenter can start the simulation with a set of initial parameters and check the validity of a theory by gathering the statistical information from the simulation agents.
- Includes historical and geoglogical data to model resources for the population.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

RebeLand Model

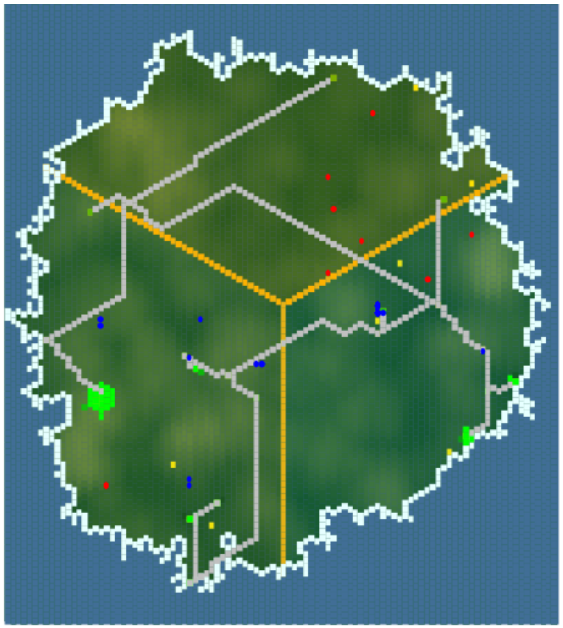


Fig. 1. Map of RebeLand Island showing its main natural and social features. *Legend:* Cities are shown in green, natural resources in yellow, and rebel and government forces in red and blue, respectively. Roads and provincial boundaries are in gray and yellow, respectively. Physical topography is shown on a green-tone scale and the island is surrounded by ocean. *Source:* Prepared by the authors.

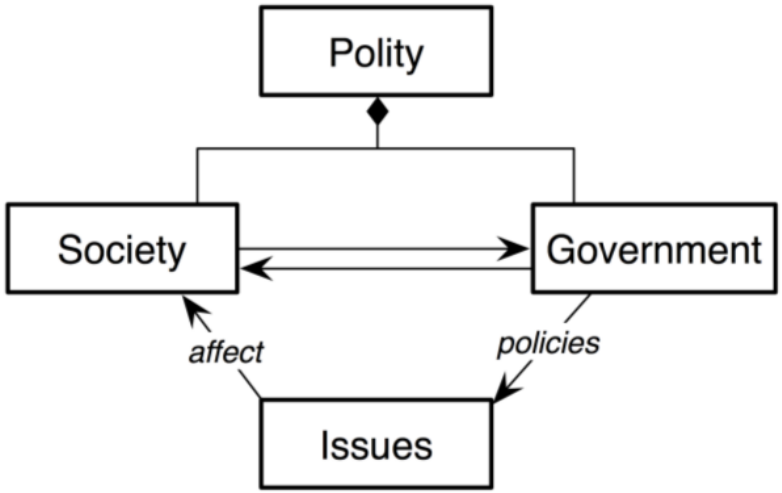
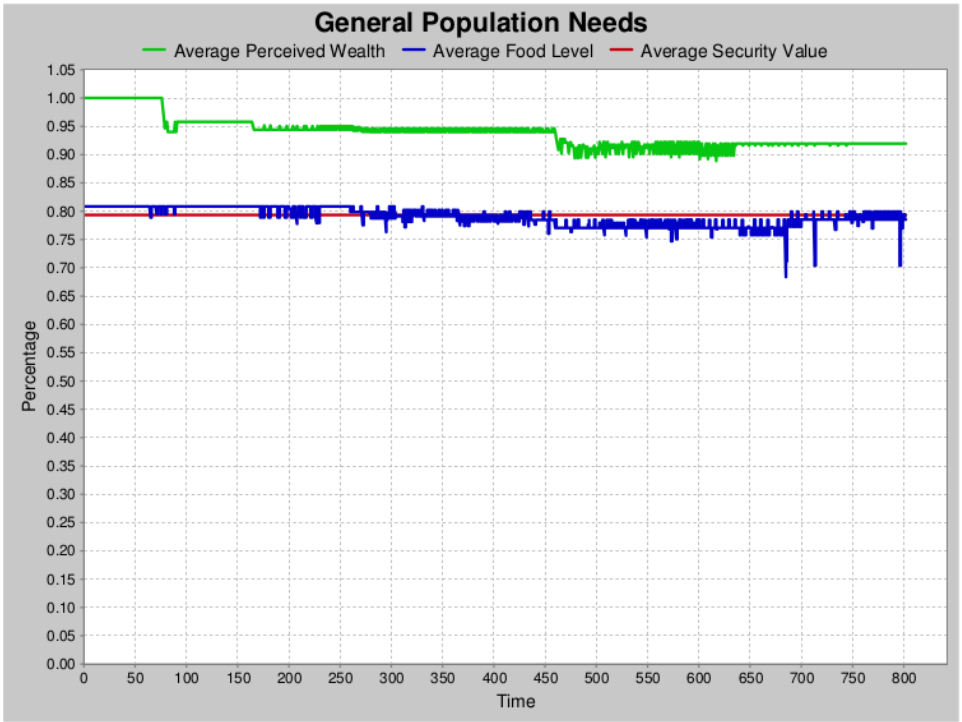


Fig. 2. High-level UML class diagram of a simple polity such as that represented in RebeLand. Government manages public issues through policies, as detailed in Figure 2. Source: Cioffi-Revilla 2009.

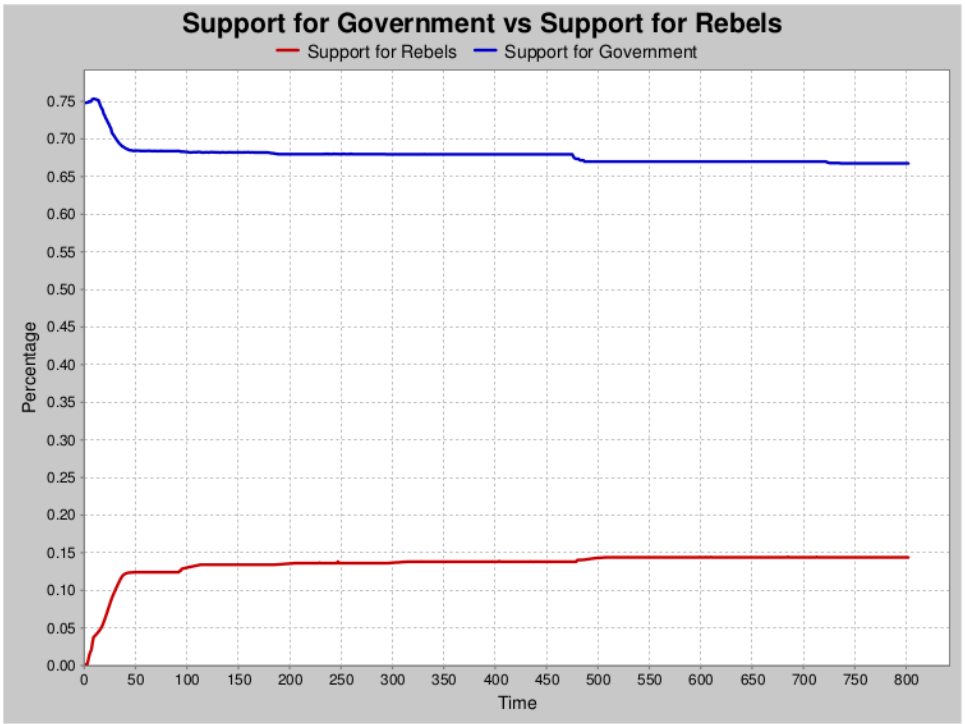
MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

RebeLand Simulation Snapshot 1



(a) General population needs under a stable scenario.

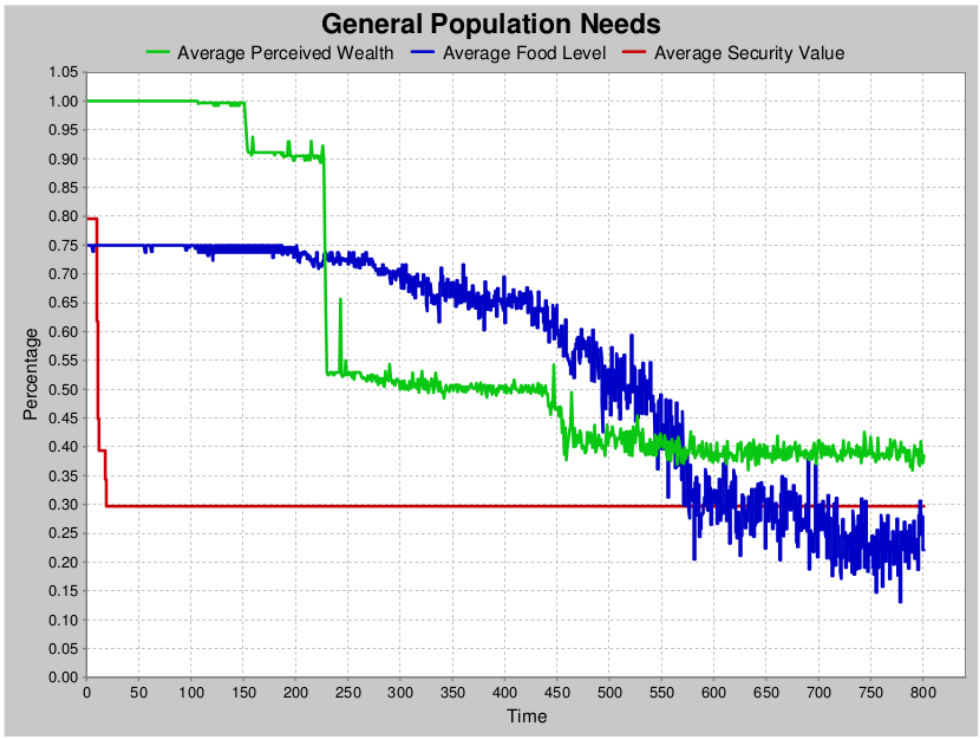


(b) Support for government and support for rebels under a stable scenario.

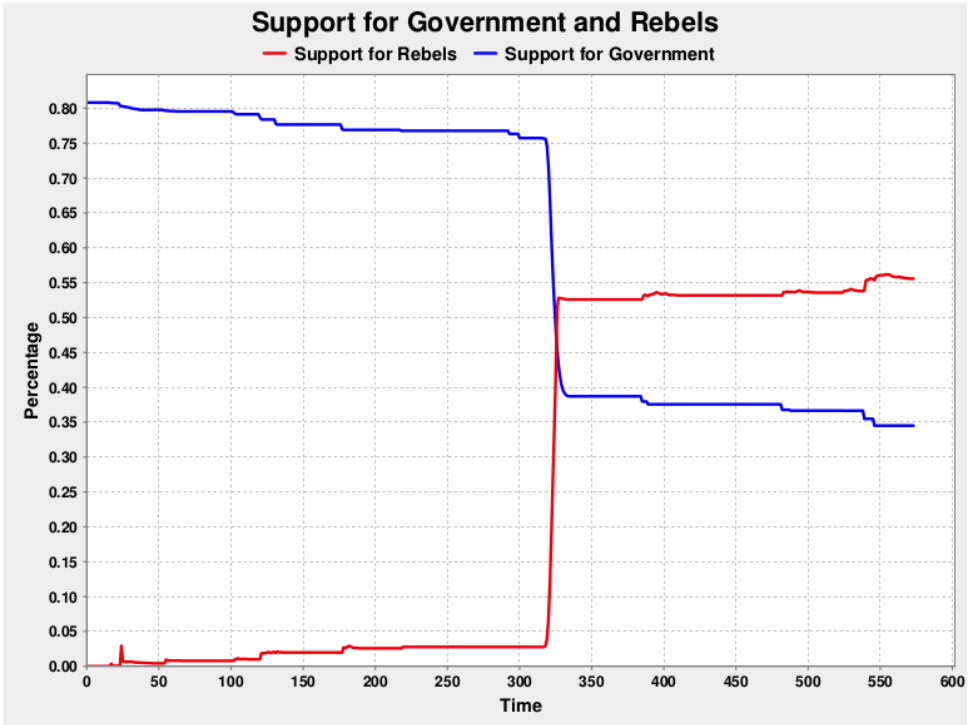
MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

RebeLand Simulation Snapshot 2



(a) General population needs under an unstable scenario.



(b) Support for government and support for rebels under an unstable scenario.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Parameters/Knobs

Unknown Parameters to Set (*all scaled to 0...1*)

- State Corruption Rate
- State Tax Rate
- Maximum State Reserve
- Minimum State Reserve
- Minimum Spending on Populace
- Maximum Police Per Capita
- Initial Reserve Army Ratio
- Standing Army Size
- State Attack Interval

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory Michigan State University

Details You Don't Care About

Optimization Algorithm

NSGA-II, 6000 evaluations.

Testing (i.e. *Generalization*)

Individual *Parameter Settings* are tested by running RebeLand on MASON 8 times – mean results were considered.

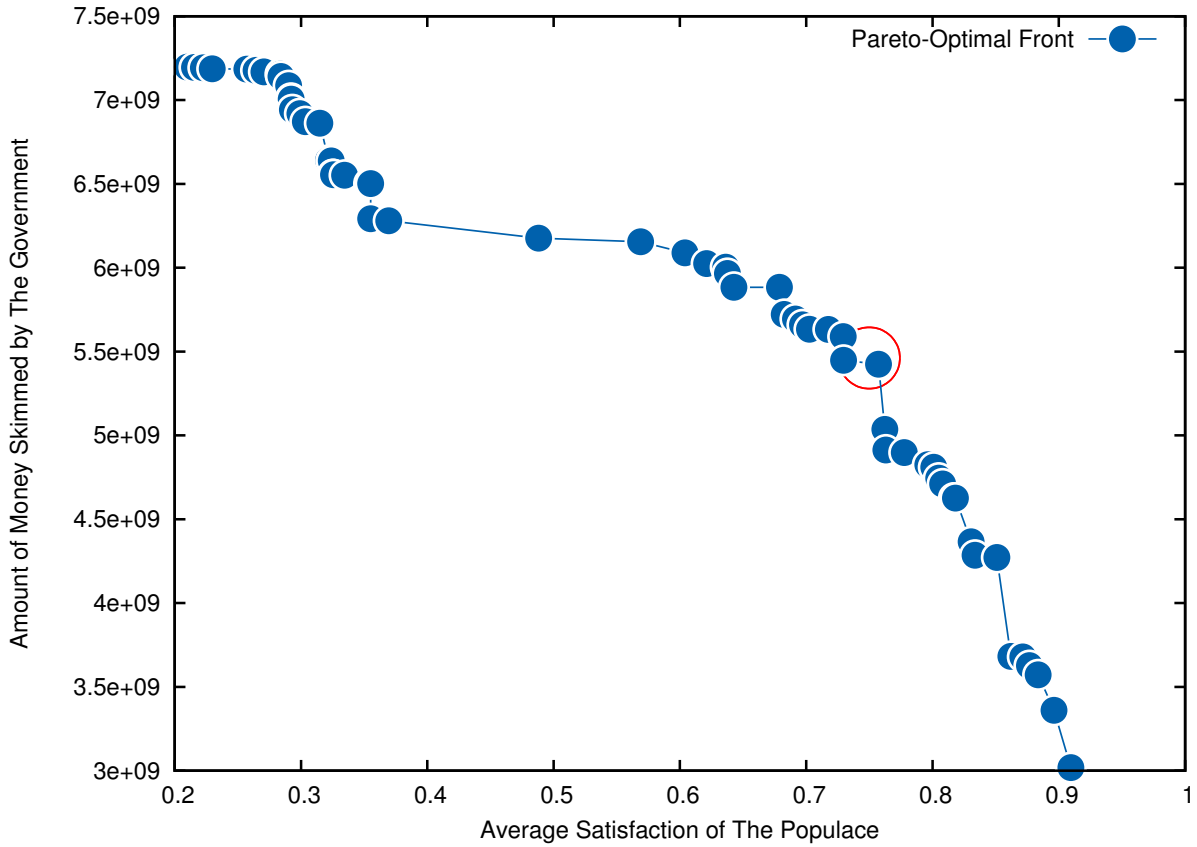
Parallelization

Master-Slave Evaluation, 30 Slave Units on Hydra
Total evaluation time: about 1 hour

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

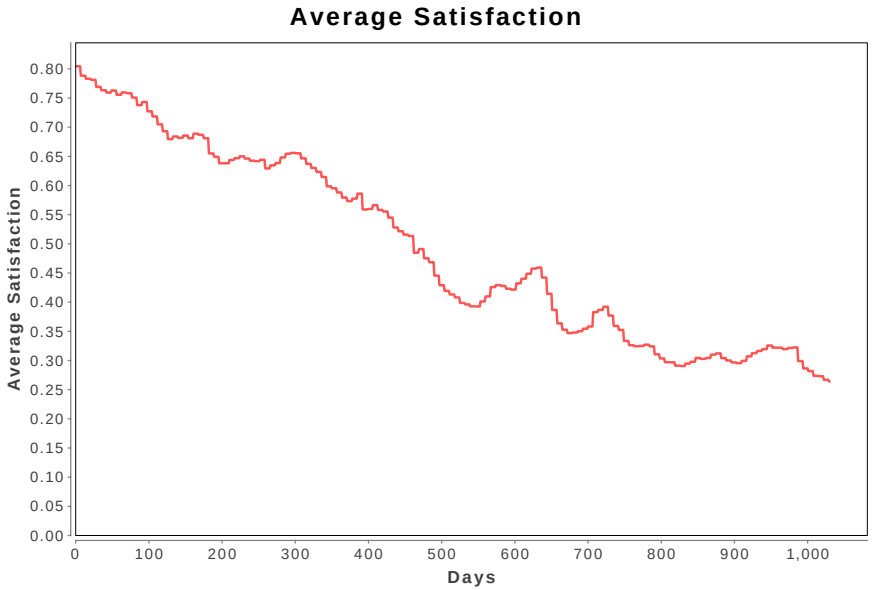
Results: The *Pareto Front*



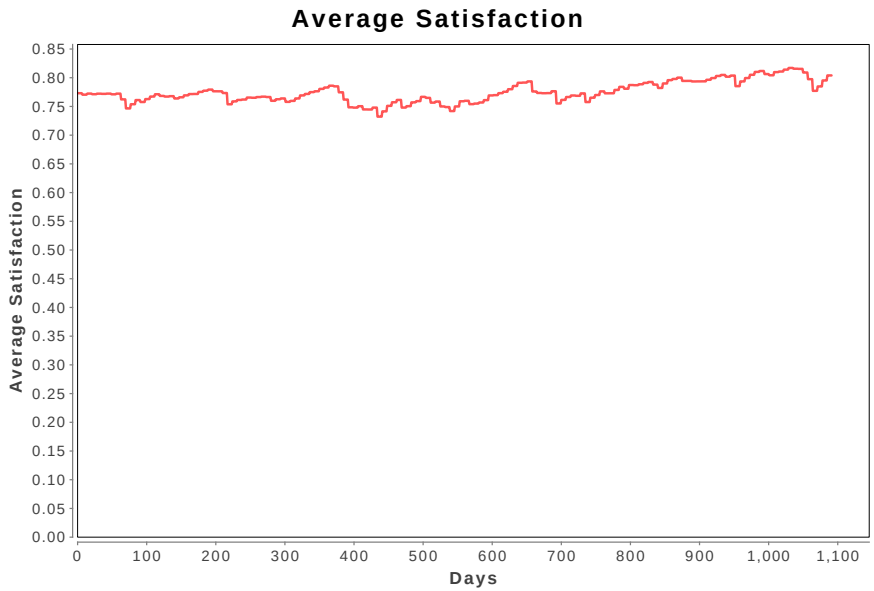
MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Results: Population Satisfaction



Before Optimzation
(Original Parameters)

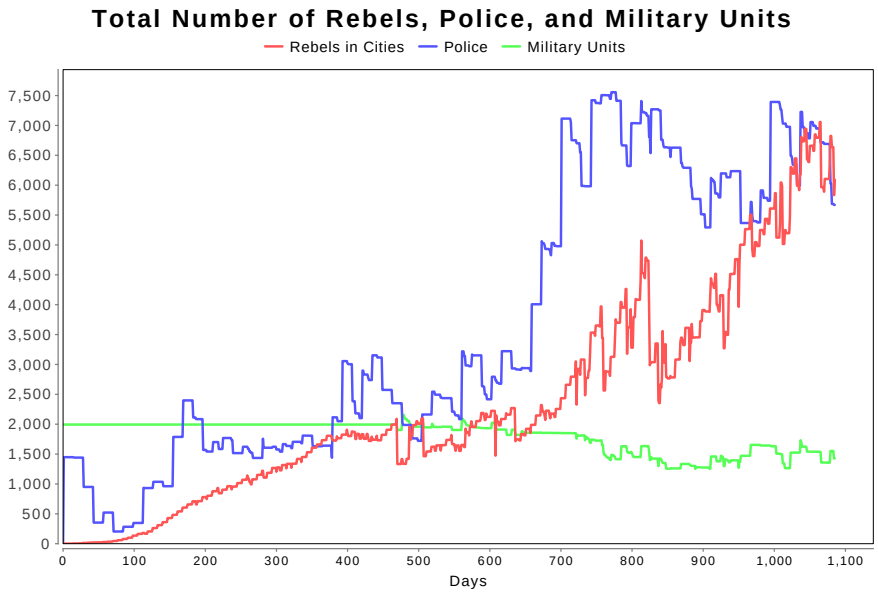


After
Optimzation

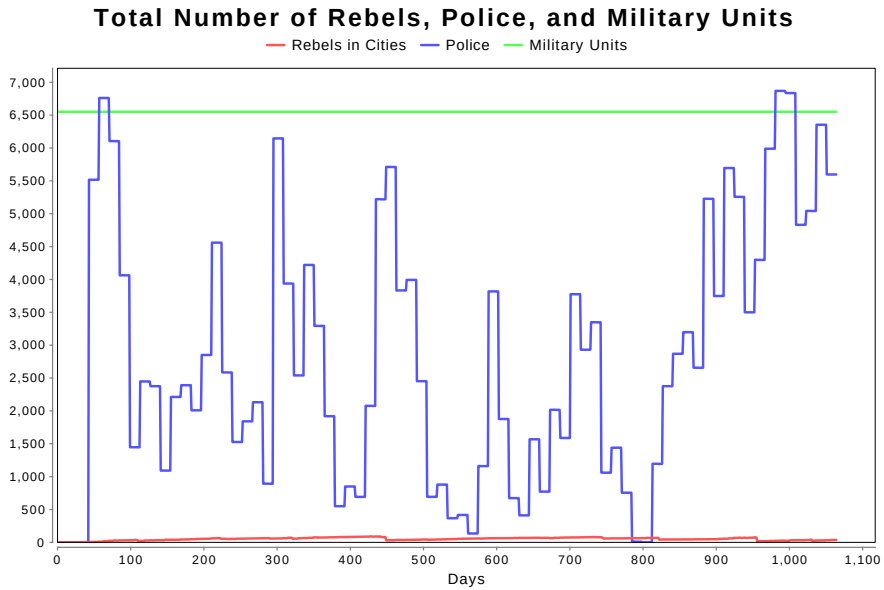
MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Results: The Overall Situation



Before Optimzation
(Original Parameters)



After
Optimzation

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Results: Some (interesting) numbers

- State's corruption rate – **86%**
- State's tax rate – **77%**
- State's maximum reserve rate – 84%
- State's minimum reserve rate – 77%
- Minimum benefit share to the populace (i.e. public spending) – **98%**
- Maximum number of police force per capita – **31%**
- Initial reserve army ratio – 0.01%
- Standing army size – 0.06%
- State's attack frequency (to suppress rebels) – **78%**

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Conclusions

Lesson ?

It's **fine** to run a **corrupt** government and keep its people **happy** – *only* if you have a **high** public spending, **large police force** and a very **frequent attack** on rebels.

Utopia ?

These results are "interesting" – i.e. *communist dictatorship* ?

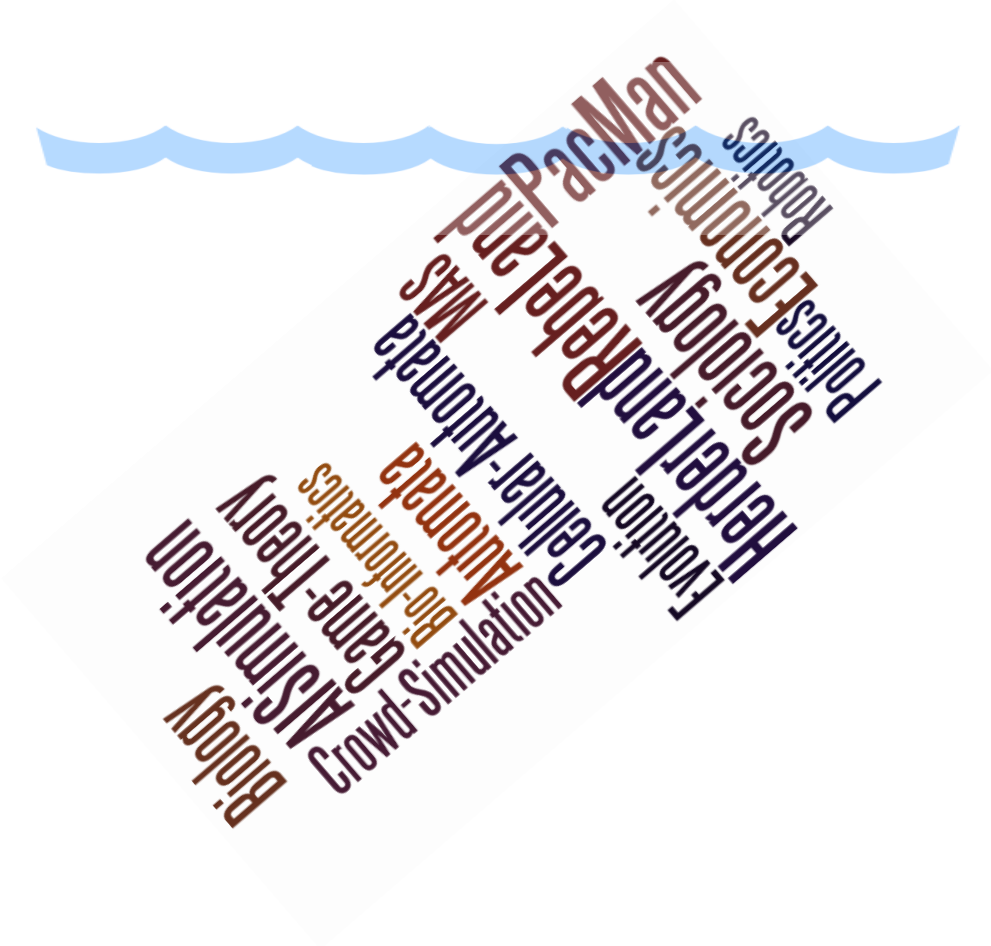
Revelations ?

Does this reveal cracks in the *RebeLand* model semantics? Or a bug in the code?

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Summary



- ECJ and MASON can be used to conduct many interesting experiments.
- Surely, symbiosis between MASON and ECJ – has a lot more to offer.
- Building a unified/standardized APIs to help plugging ECJ capabilities into MASON models – could just be a tip of the iceberg.

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University

Questions?

Discussions?

MICHIGAN STATE UNIVERSITY

Computational Optimization and Innovation Laboratory
Michigan State University