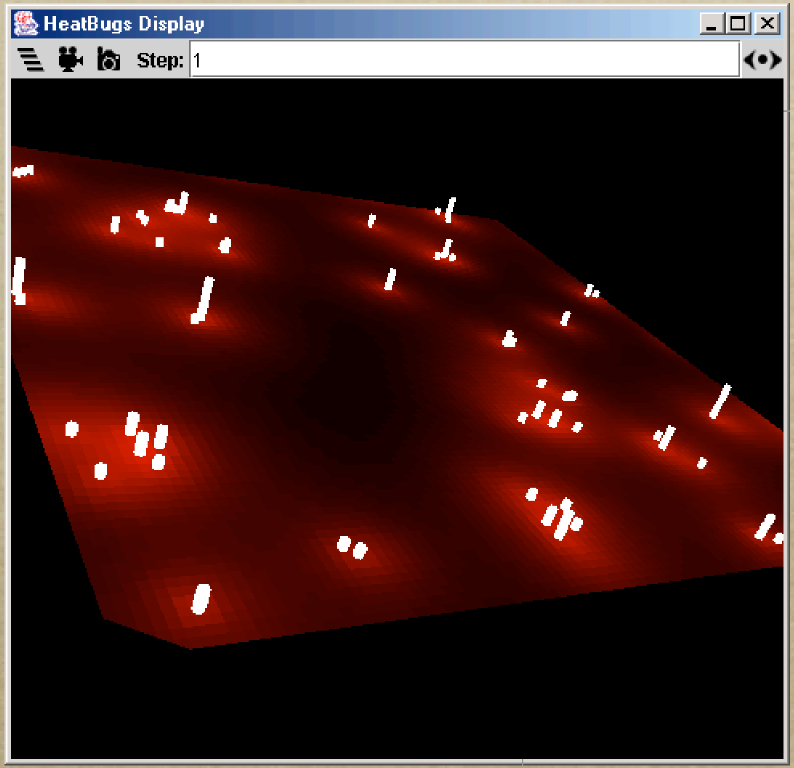
MASON *A Java Multi-agent Simulation Library*

Sean Luke
Gabriel Catalin Balan
Liviu Panait
Claudio Cioffi-Revilla
Sean Paus

George Mason University's Center for Social Complexity and Department of Computer Science

MASON

- ***M***ulti ***A***gent ***S***imulation ***O***f ***N***eighborhoods... or ***N***etworks... or something...
- Fast, portable, multi-agent core in Java, plus visualization tools and media tools
- Designed for both artificial intelligence and computational social science agent-based modeling. Dual-purpose is intentional for cross-fertilization.



The Big Picture

- Why MASON exists
  1. Produce new discoveries (Galileo and Smarr)
  2. Replicate prior results
  3. Provide new computational facilities (von Neumann)
  4. Model new agent architectures
  5. Inspire & implement new formalisms
  6. Open new research frontiers (Bronowski)
  7. Inspire future improvements
- Positive evaluation of MASON's predecessors by these standards.

---

*BTW: How does/should CSS formally evaluate a simulation environment? We know how to evaluate concepts, hypotheses, models, theories; but simulators?

The Big Picture

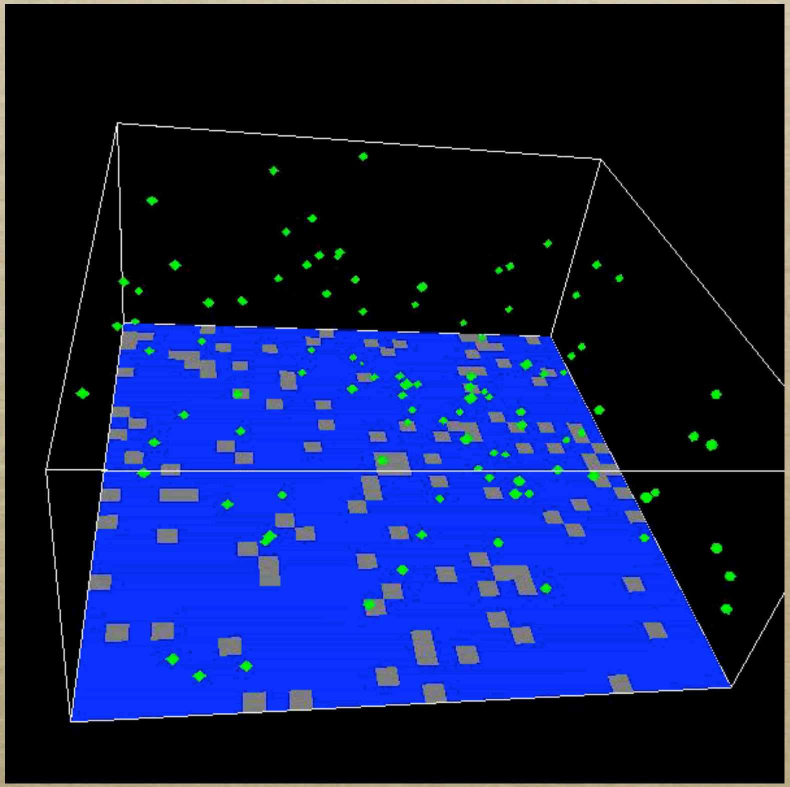
- MASON design goals
  - Large numbers of simulations
  - Guaranteed duplicatable scientific results
  - High degree of modularity and flexibility
  - Small, easy to understand core model
  - *Separate* visualization tools

The Big Picture

- We present MASON as an evolution stemming from a tradition of inspiring precursors: Swarm, Ascape, Repast
- MASON is a joint project by George Mason University's Center for Social Complexity (C. Cioffi) and the Evolutionary Computation Lab (S. Luke).
- "Vertical team" approach: Faculty & student involvement from GMU (& TJ)

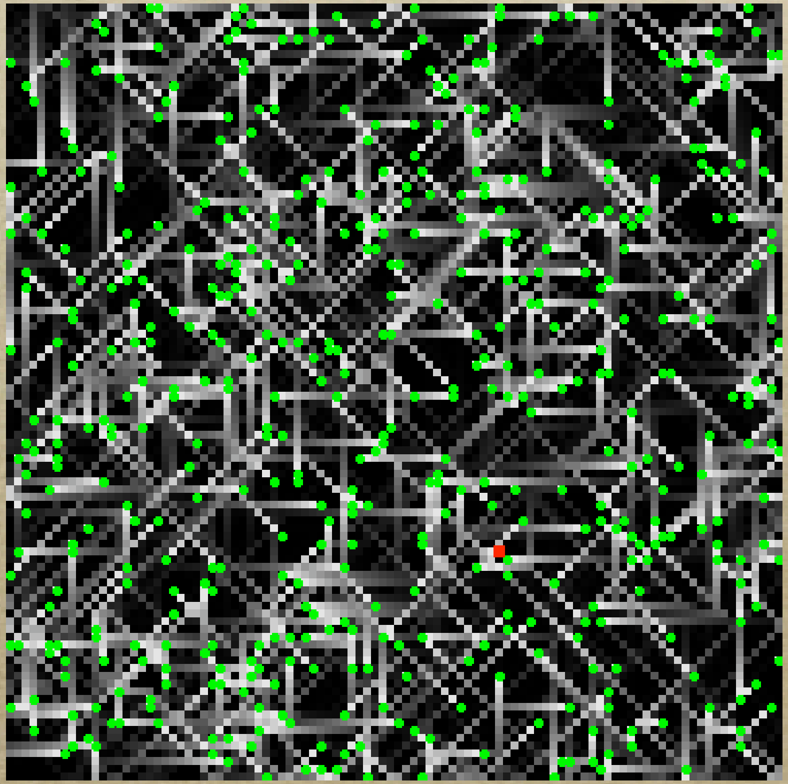
MASON

- General-purpose, single-process, discrete-event simulator
- Efficiently supports large numbers of agents
- Applications as diverse as
  - Social complexity
  - Physical Modeling
  - Abstract Agents
  - AI, Machine Learning



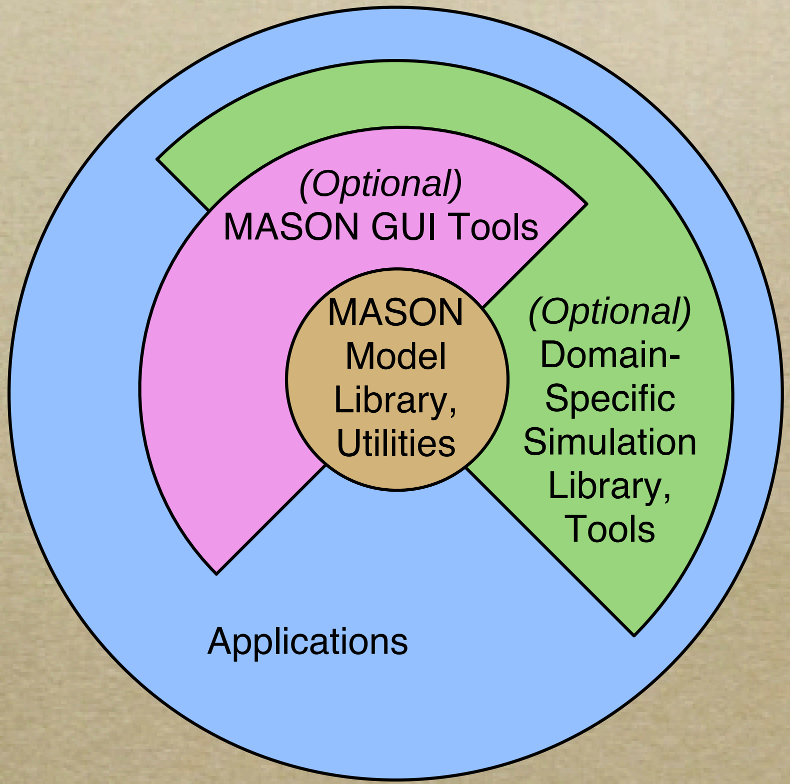
MASON Features

- Highly modular, layered architecture
- Portable, guaranteed duplicatable results across different platforms
- Total separation of model from visualization
  - Dynamically add, change, remove visualization
  - Cross-platform checkpointing, recovery

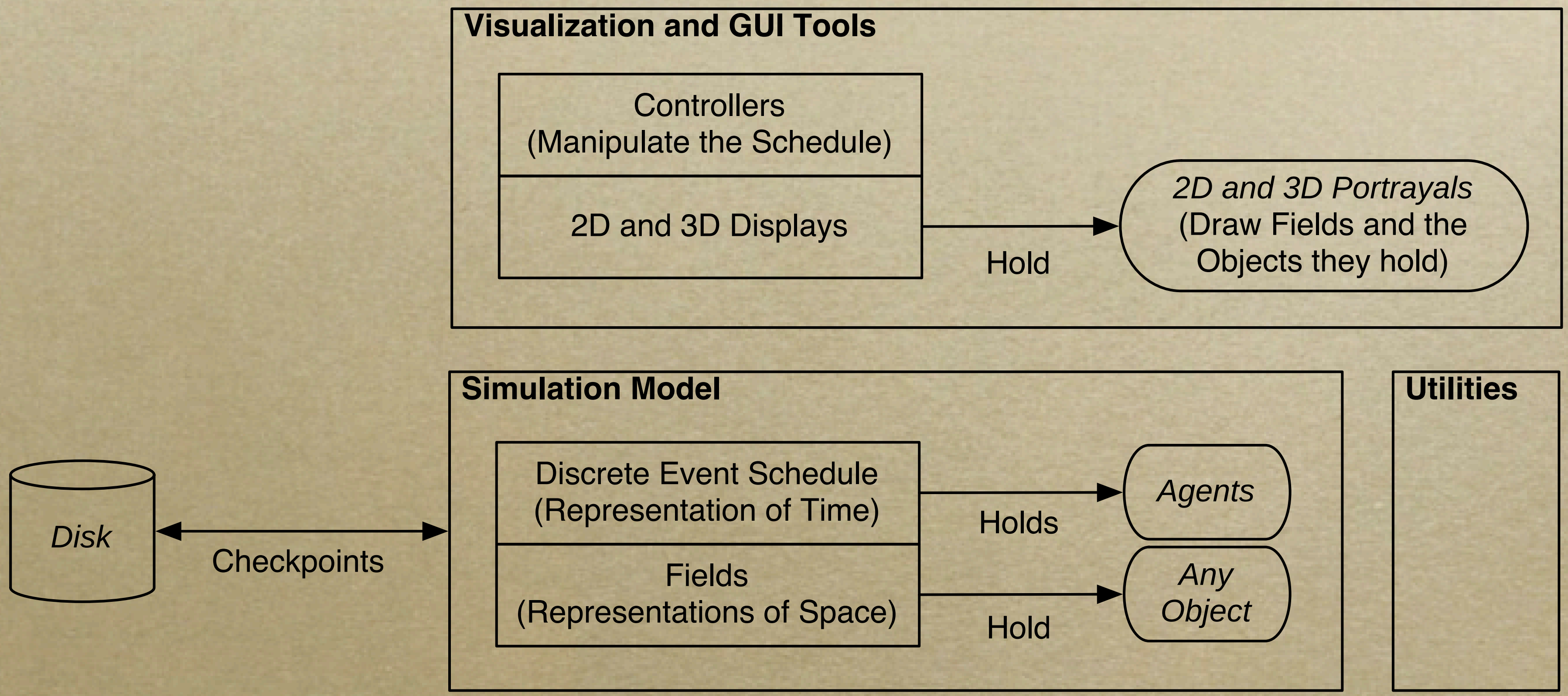


MASON Layered Architecture

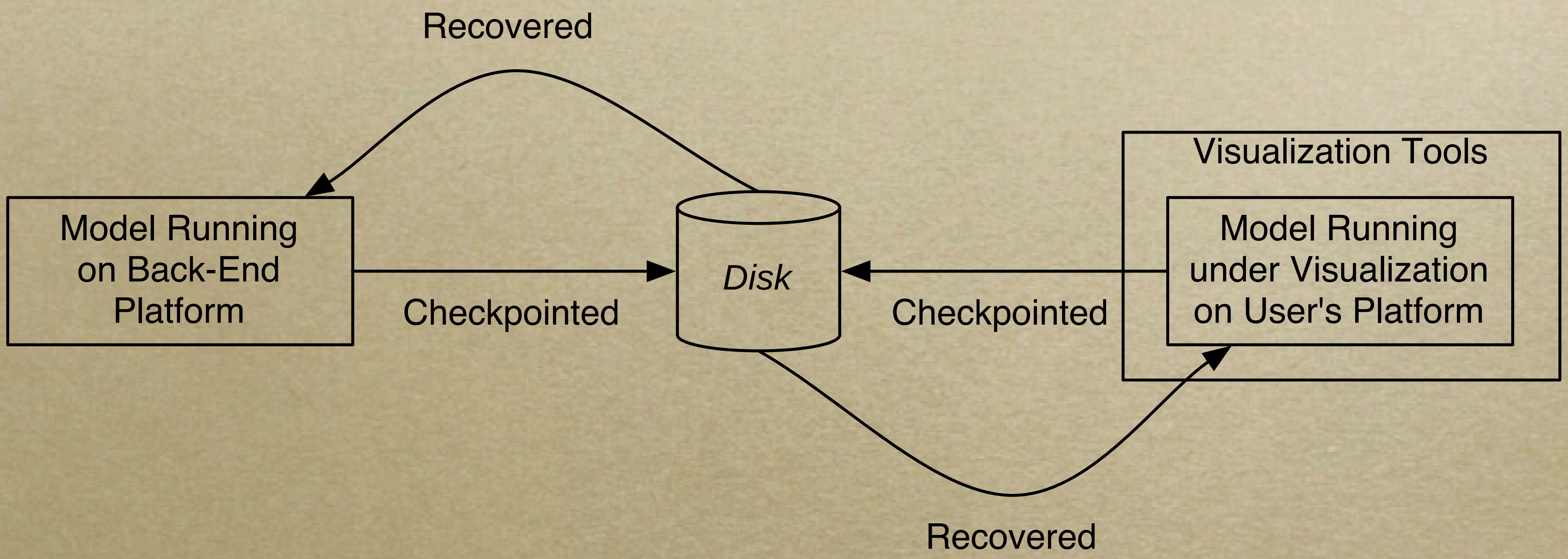
- Utilities
- Core model library
- Visualization tools
- Custom simulation layers
- Simulation applications



Layer Interactions

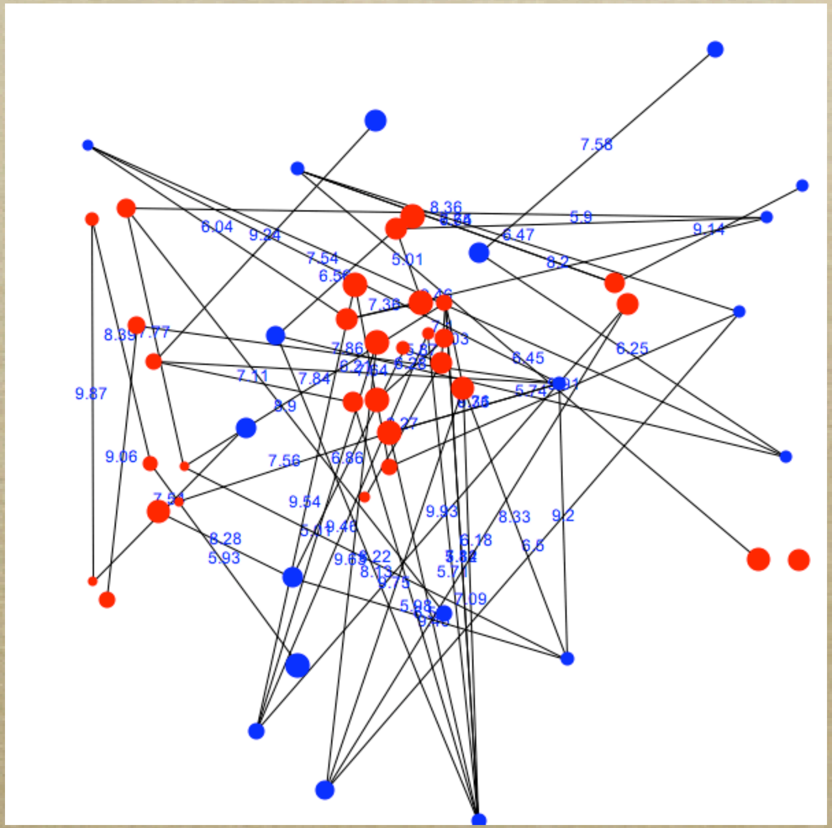


Checkpointing and Recovery



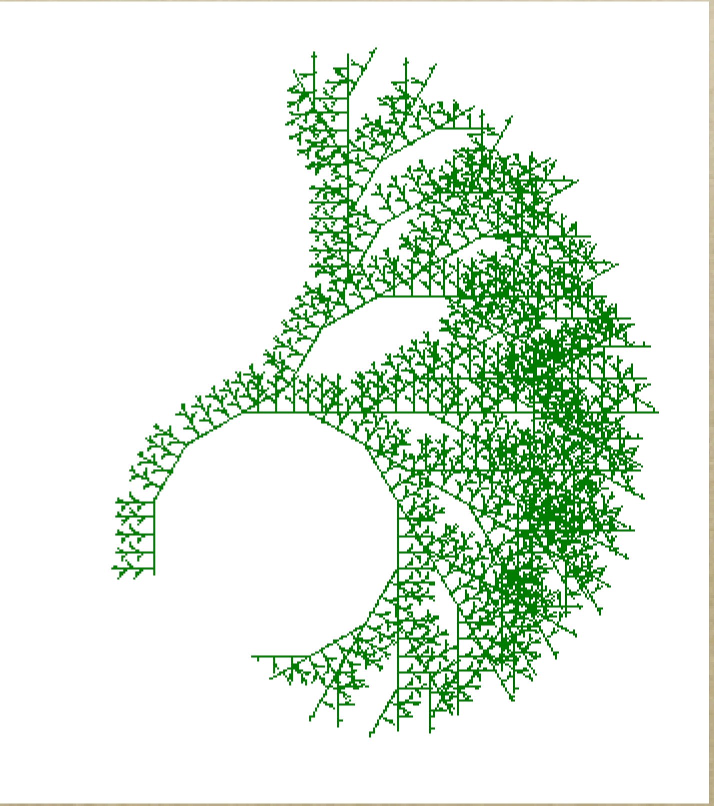
MASON Neighborhoods

- 2D, 3D Fields
- Hexagonal, Toroidal
- Discrete, Continuous
- Network Fields
  - (Directed Graphs)
- 2D and 3D Visualization



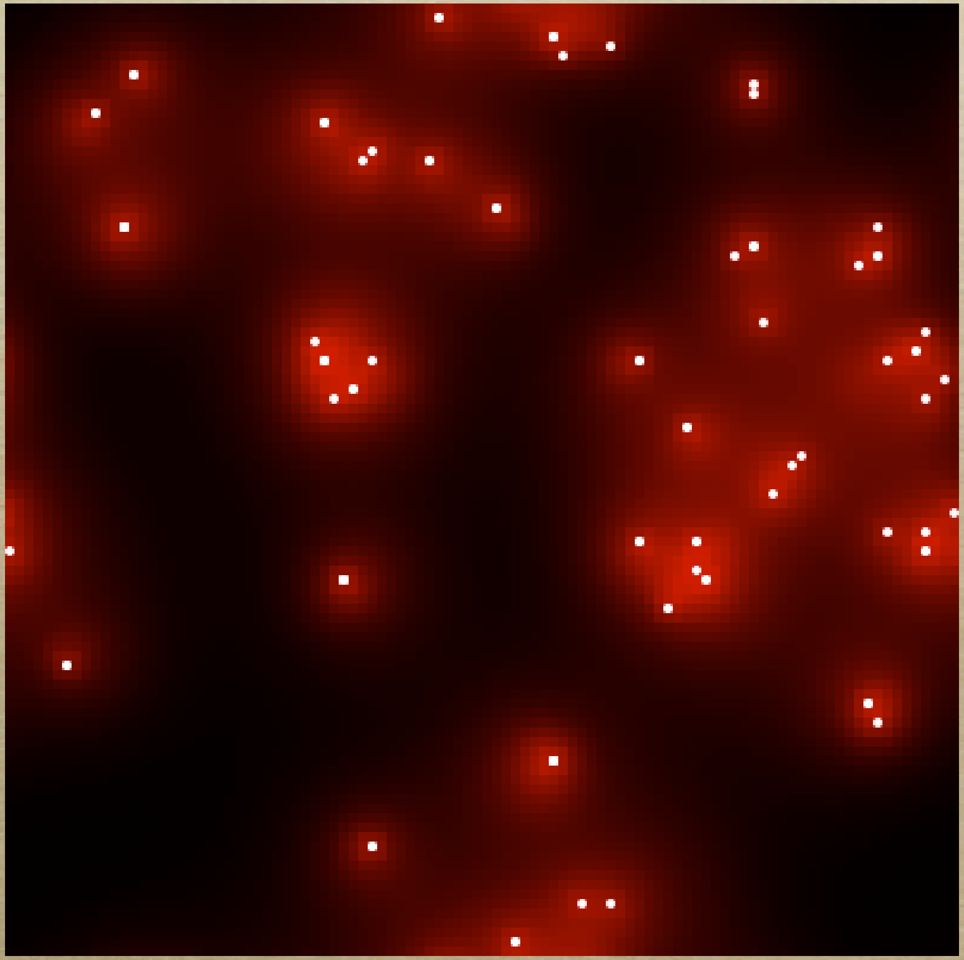
Differences with RePast

- MASON...
  - Model - visualization separated
  - 3D models and displays
  - Faster, especially on MacOS X
  - Cleaner, smaller
- RePast has built-in...
  - GIS, Excel import/export, charts and graphs, SimBuilder
    - In MASON these would be in the "custom simulation library" layer

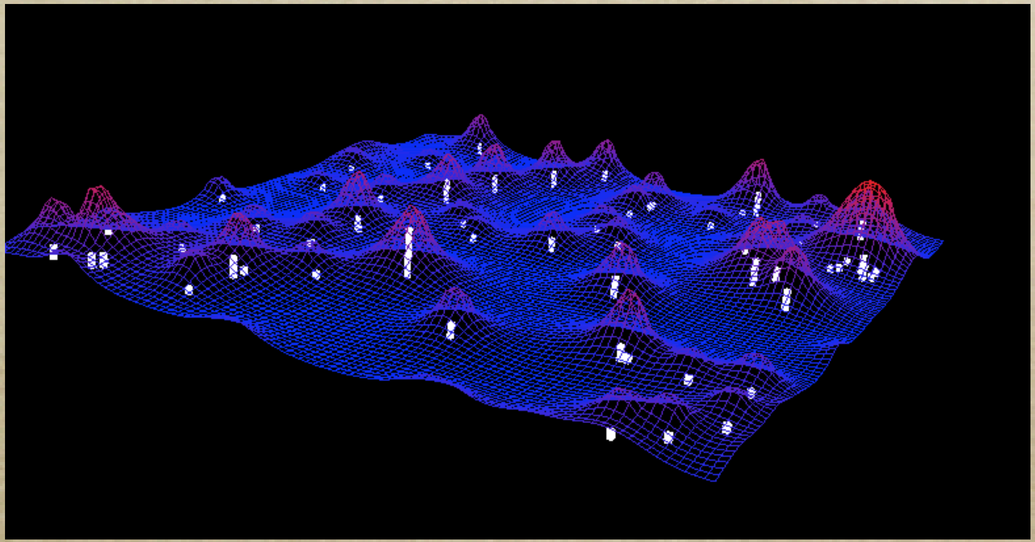


MASON doesn't have... (*yet!*)

- RePast uses linearized array classes; MASON uses Java arrays
- RePast's schedule uses doubles, MASON's uses longs with double extensions
- RePast allows objects to be moved by the mouse



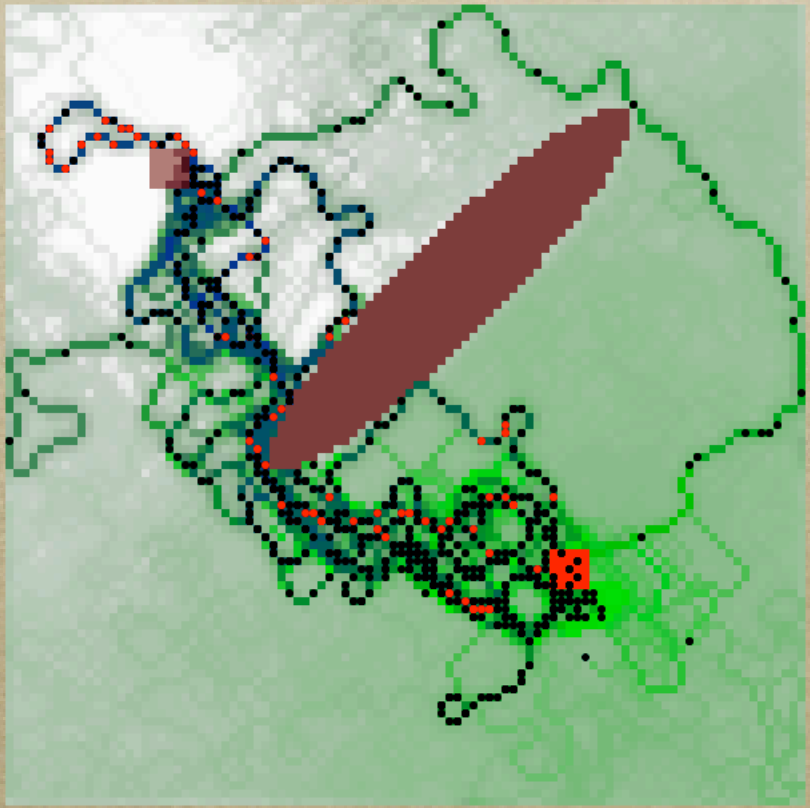
Test Cases



- Ant Foraging
- Micro Air Vehicles
- HeatBugs
  - *to compare with RePast, Swarm*
- Anthrax Dispersion in Human Body
  - *port of existing Swarm simulation*

Ant-Inspired Foraging

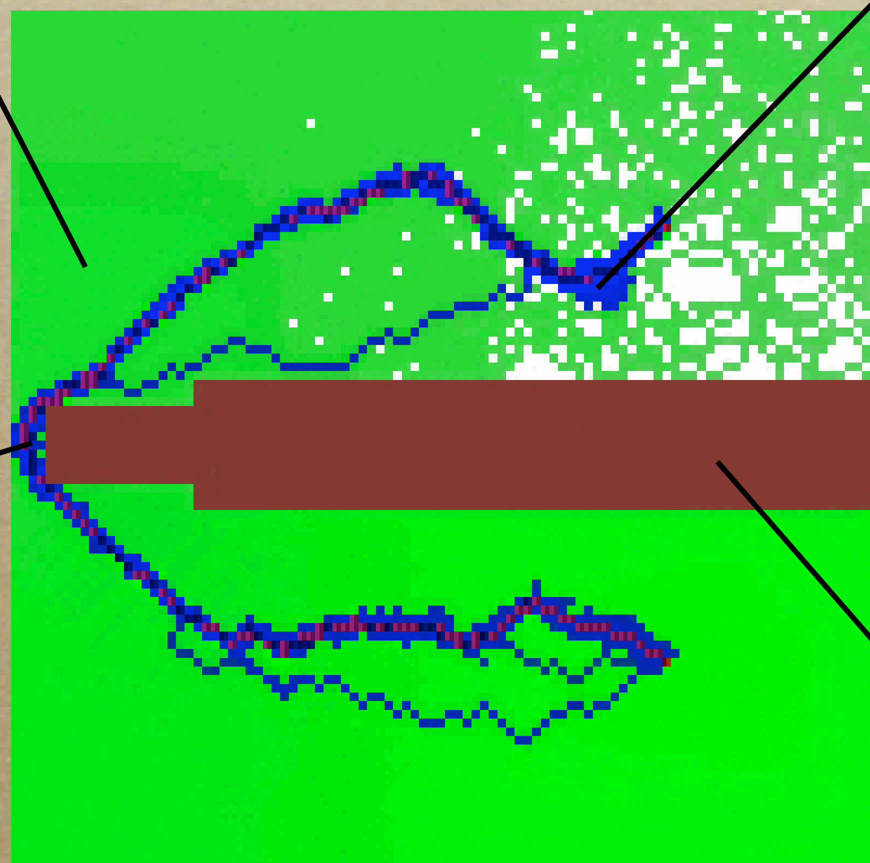
- Second International Workshop on the Mathematics and Algorithms of Social Insects
- Problem domain involving a large number of agents
- Task: locate the food source and repeatedly carry food items back to the nest
- Agents use pheromones to mark trails connecting sites



Ants: MASON Setup

Pheromones for direction to nest (DoubleGrid2D)

Pheromones for direction to food (DoubleGrid2D)



Agents (with or without food) (SparseGrid2D)

Obstacles (DoubleGrid2D)

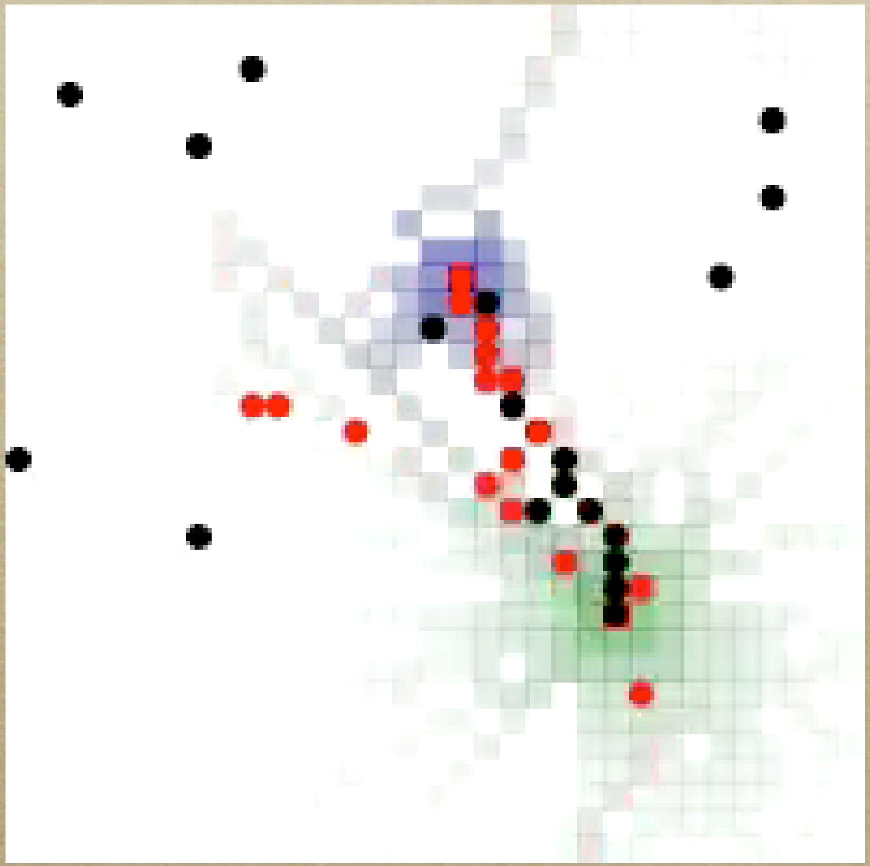
Evaporation & Diffusion Agent

Birth-Control Agent

- Ant agents are created in the nest
- Ant agents die after a number of time steps
- An additional simulation agent manages the creation of new foraging agents when needed

Learning Foraging Behaviors

- Hooked up MASON with ECJ evolutionary computation library
- ECJ spawns large numbers of MASON simulations to evaluate performance of candidate ant behaviors



Micro-Air Vehicles

- Small (under 1 meter) unmanned aerial vehicles
- Inexpensive
- Large “swarms” of vehicles for cooperative surveillance

MAV Challenges

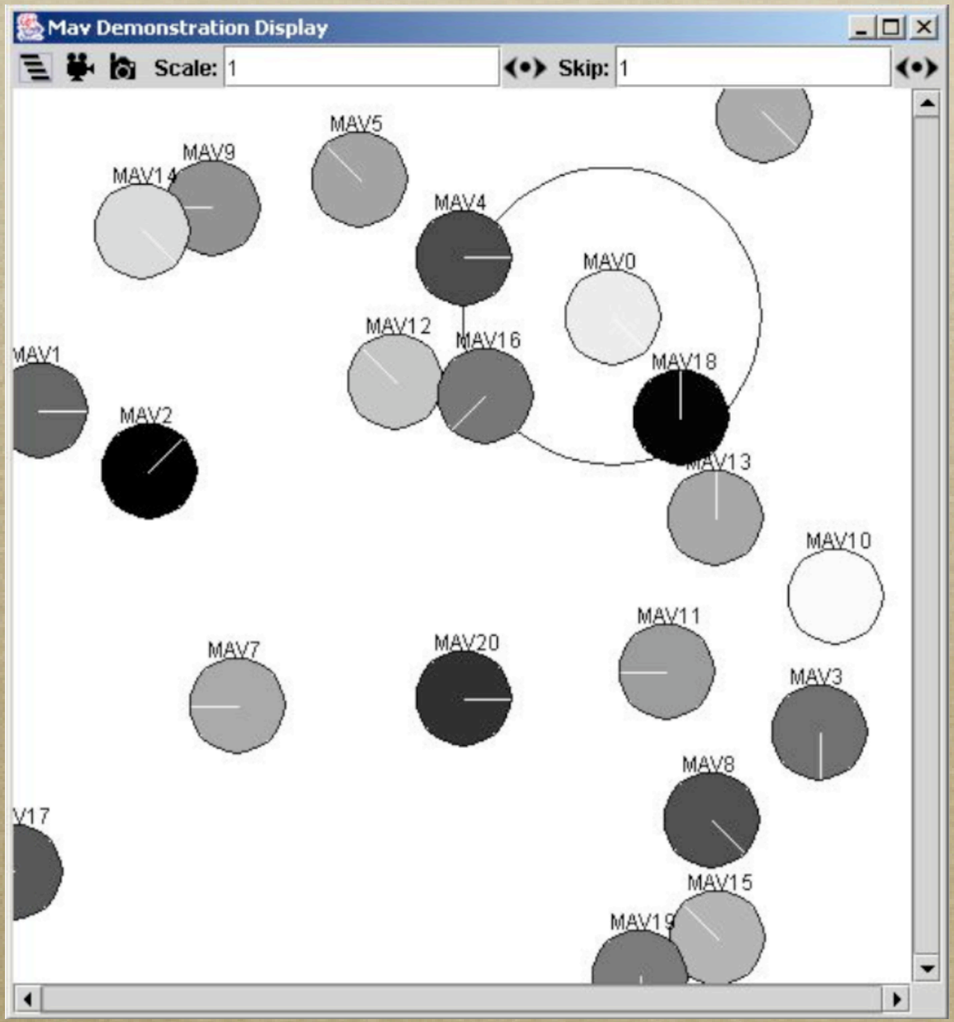
- Unmanned Aerial Vehicles (UAVs) are ordinarily operated by remote control: team of 6 people per UAV
- But a swarm of 1,000 MAVs = 6,000 people, plus coordination between them!
  - MAV swarms ***must*** be autonomous
- Programming autonomous behaviors by hand is ***hard***

Learn the MAV Behaviors

- Use machine learning to develop autonomous MAV swarm behaviors
- Evolutionary computation, reinforcement learning
- *Requires:*
  - EC system to invent behaviors
  - Fast simulator run on many machines in parallel to test behaviors

MAV Swarm Simulation

- 10 – 10,000 MAVs
- Continuous 2D Field in MASON
- Connected to EC system
  - Evolved behaviors to perform maximum coverage of desired areas without crashing into one another



Where to find MASON

- Evolutionary Computation Laboratory
  Department of Computer Science
  http://cs.gmu.edu/~eclab/
- Center for Social Complexity
  http://socialcomplexity.gmu.edu
- (Will be up immediately after Agent2003)
- *...or ask us during conference to burn a CD*

MASON *A Java Multi-agent Simulation Library*

Sean Luke
Gabriel Catalin Balan
Liviu Panait
Claudio Cioffi-Revilla
Sean Paus

George Mason University's Center for Social Complexity and Department of Computer Science