A Survey, Model Design, and Benchmark Test Implementation of Agent-Based Applications

By Caleb Yang
With Professor Munehiro Fukuda

Contents

Introduction

The process begins with the surveying of simulation applications in the fields of biology, business/industry, and economics/social sciences that rely on agent-based modeling. The key information to extrapolate from the research papers is how can that application be re-implemented into C++. This will consist of a few factors like the various agents in the system, an agent's micro-behaviors, how will it communicate with other agents, even how the topology is constructed. Possible topologies could be like a 2-D plane divided into nodes, a network of connected nodes, or most importantly topologies that are not easily represented in MASS C++.

Application-to-Model Mapping

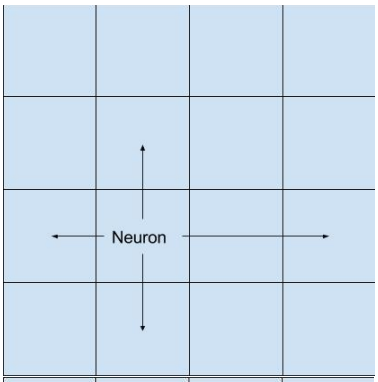
Biology	
Brain Grid	[1]
Sugar Scape	[2]
Dengue Fever <Swarm>	[3]
Microvascular	[4]
Tuberculosis (TB)	[*]
Business/Industry	
Virtual Design Team (VDT)	[*]
Social Network Modeling	[*]
Economics/Social Sciences	
Bank Bail-in/out	[*]

Preexisting MASS C++ Models
[1] Brain Grid [Neural Network]
[2] Sugar Scape
[3] Random Walk
[4] Game of Life [Cellular Automata]

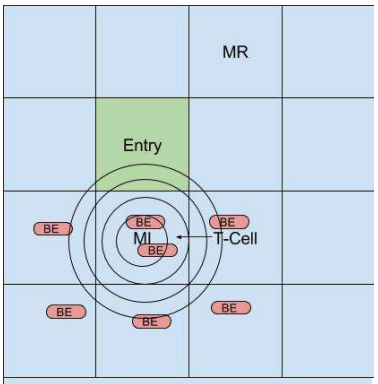
[*] Unique Models
- Tuberculosis (TB)
- Virtual Design Team (VDT)
- Bank Bail-in/out [Clustered Agents]
- Social Network Modeling
  [Network of Static Agents]

Models

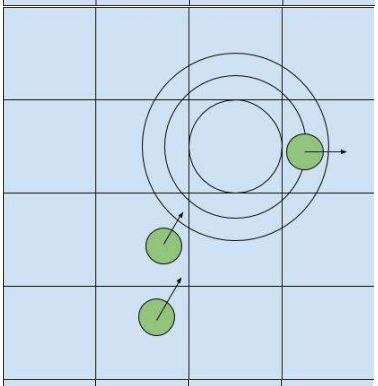
Neural Network



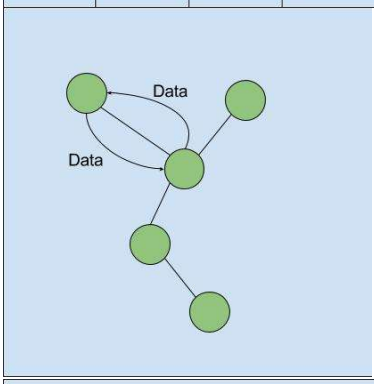
Tuberculosis



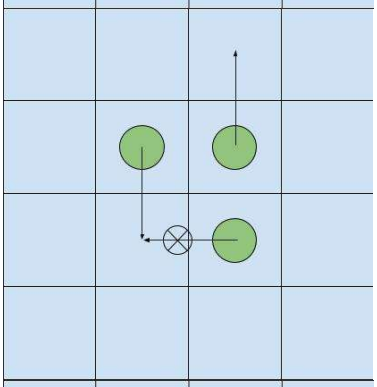
Sugar Scape



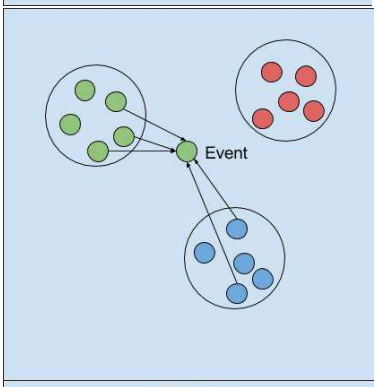
Network of Static Agents



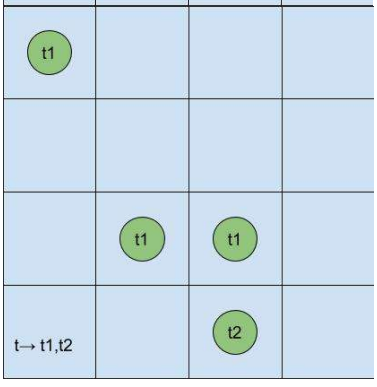
Random Walk



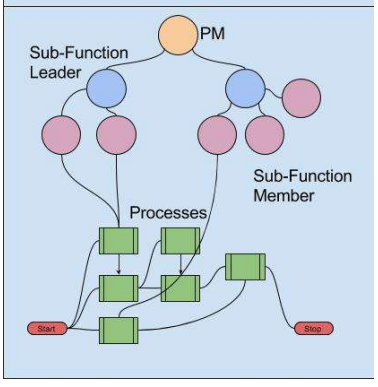
Clustered Agents



Cellular Automata



Virtual Design Team



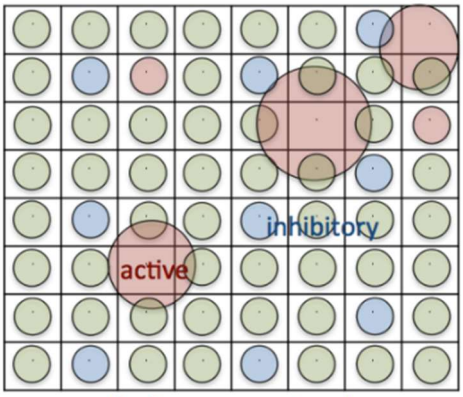
Source: Fukuda, M., Stiber, M., Salathe, E., & Kim, W., "CDS&E: Small: Multi-Agent-Based Parallelization of Scientific Data Analysis and Simulation.", 2013

Application Synopsis

Brain Grid

Brain Grid is a neural network simulator that imitates the growth of electrical activation and synapses among neurons. Neighbors are determined while the active neuron grows connections with other neurons. While creating larger networks of neighbors a calculation of synapses is taken which represents brain activity. It pairs off neurons determining the outcome based off the given type of neuron.

Figure: Sample Neural Network Simulation



8 x 8 neuron network

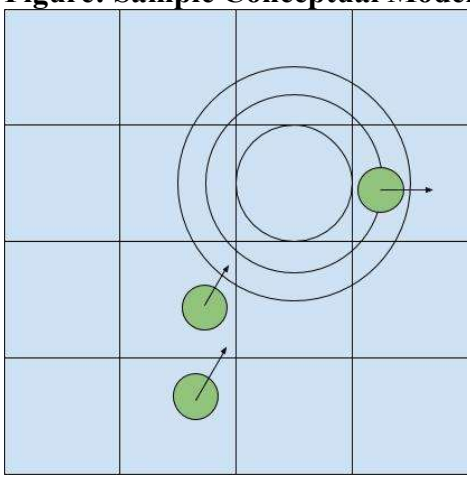
Represented Simulation "Entities":
Red – Active Neuron
Blue – Inhibitory Neuron
Green – Neutral Neuron

Why evaluate all neurons for activity?

The purpose of the interacting Neuron nodes is to simulate brain activity. The ratio of Active-to-Inhibitory Neurons determine the probability for brain activity in the given region of the simulation environment.

Sugar Scape

Figure: Sample Conceptual Model



Sugar Scape is a well-known model type where there exist the agents/inhabitants and the environment, where the agents interact based off a set of rules in a 2-D array. In regards to MASS C++ development, Sugar Scape has been defined as a simulation for ants reacting to the environment's pile(s) of sugar. A pile of sugar is defined as a region where the resource extends out in a circle, decreasing in concentration the farther the cell is from the center of the pile. This version is a simplified version where sugar is not consumed, or traded.

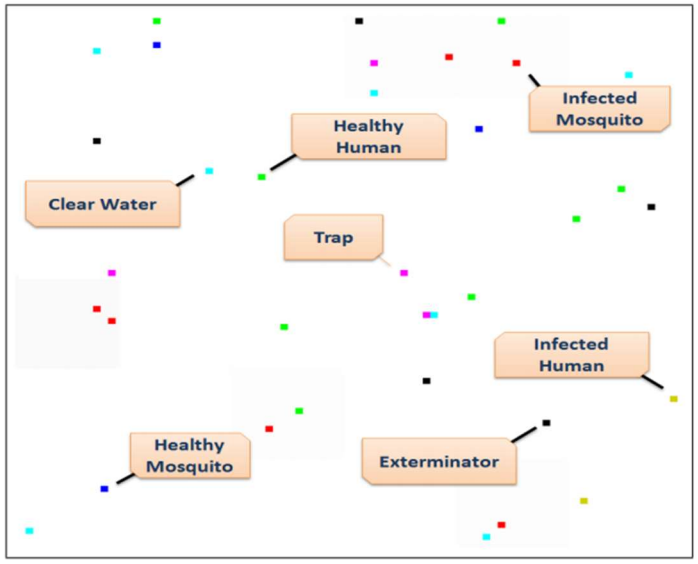
Source: Luis F.O. Jacintho, Andre F.M. Batista, Terry L. Ruas, Maria G.B. Marietto, Fabio A. Silva, "An Agent-Based Model for the Spread of the Dengue Fever: A Swarm Platform Simulation Approach", Spring Simulation Multiconference, 2010

Dengue Fever

In response to the spread of Dengue Fever in Brazil, a team at the Federal University of ABC Centre of Mathematics decided to approach the problem through agent-based modeling. The immediate goal of this research was to develop a simulation that produced similar results to real-world Dengue Fever studies. Ultimately, the researchers want to get to the point where they can simulate the outbreak of Dengue Fever, to predict mosquito behavior and inform future plans.

For this simulation Swarm was chosen to be the platform of the simulation. Swarm's hierarchical modeling for agent-based systems made transitioning a conceptual model into a computational model a lot easier. Additionally, Swarm's agents can represent a group of agents and just an individual, which assist in representing the swarms of mosquito populations.

Figure: Sample Dengue Fever Simulation



Represented Simulation "Entities":
Mosquito – Infection vector
Human – Multiple infections kills
Exterminator – Kills mosquitos
Food – i.e. based off the stage in life
Clear Water – Place to lay eggs
Trap – Prevent mosquito proliferation

Mosquito Lifecycle:
Egg – Most resilient stage in life
Larva – Primarily feeds
Pupa – Least affected by larvicide
Adult Mosquito – Female mosquitos require blood for the maturation of

Note: Mosquito lifecycle is sensitive to humidity and temperature

The environment includes a food rate, humidity, temperature as well as clear water and traps. Studies show that humidity and temperature not only affect the lifecycle of mosquitos but determines a lower likelihood of infection during a given season.

Human Agent: Human agents do not grow in population or die of old age. In this simulation human agents randomly walk around. If infected they could infect healthy mosquitos with the Dengue Fever, or be infected again. A multiple infection rate is significant because the higher the amount of times a human agent is infected the greater the fatality rate.

Mosquito Agent: This simulation includes the entire lifecycle of mosquitos in order to provide opportunities for exterminator agents to reduce the spread of Dengue Fever as well as encapsulate the effects climate has on the spread of Dengue Fever.

Exterminator Agent: This agent is responsible for laying down trap environmental entities. The activity of exterminator agents represents an entity that can see where large populations of mosquitos because of "sensors." In actuality they represent the organizational intervention that the researchers plan to develop further.

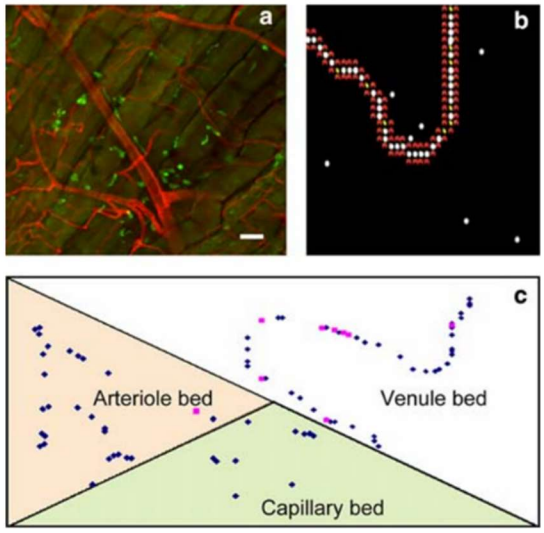
Source: Bryan C. Thorne, Alexander M. Bailey, "Multi-cell Agent-based Simulation of the Microvasculature to Study the Dynamics of Circulating Inflammatory Cell Trafficking, Annals of Biomedical Engineering, 2007

Microvascular

A study was conducted at the Department of Biomedical Engineering, University of Virginia that utilized an agent-based computational model and included network blood flow simulation in order to study the negative effects associated with inflammation on the body. It includes network blood flow simulation in order to account for monocyte (large white blood cell) traffic throughout the microvascular network (i.e. arteries, veins, capillaries). This study focuses on monocyte since it has a major contribution to the inflammation activity, such as cytokine or chemokine secretions.

The simulation is constructed like cellular automata but starts with a 1-1 relationship with the space element. Additionally, the monocyte agents follow the designated vascular network in the system. An agent in this context represents a bodily cell (i.e. 7 major agents) that are listed below.

Figure: Sample Modeling Process



[A] Dyed Real-World Sample
[B] Manual Translated Virtual Model
[C] Conceptual Representation of the 3 Vascular Cell/Agent Construction

Represented Simulation "Entities":

Monocyte – A large white blood cell that rolls and sticks to the vascular network in order to survey the body.
Arteries – Vascular network portion with diameter greater than 10nm.
Veins – Vascular network portion with diameter
Capillaries – Vascular network portion with diameter equal to or less than 10nm.
Arteriole Bed – Determined by proximity to the arteries.
Venule Bed – Determined by proximity to veins
Capillary Bed – Determined by proximity to capillaries.

This simulation depended on two platforms. NetLogo for the agent-based modeling, and MATLab for the network blood flow analysis. When taking into account for the network blood flow analysis, several related parameters are needed to properly model the simulation as well as the effects of monocyte on the system. These network blood flow analysis parameters include rolling and adhesion factors, red blood cell viscosity, as well as blood pressures and flows at inlet/outlet points or split vascular sections.

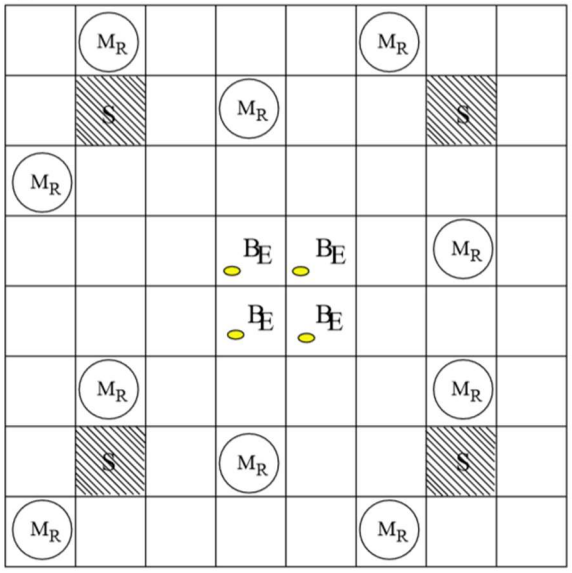
Source: Jose L. Segovia-Juarez, Suman Ganguli, “Identifying control mechanisms of granuloma formation during M. tuberculosis infection using an agent-based model”, Journal of Theoretical Biology, 2004

Tuberculosis

At the Department of Microbiology and Immunology, University of Michigan, there was a study conducted with an agent-based model to identify control mechanism of granuloma formation during an infection of M. tuberculosis. Granuloma is a mass of granulation tissue, typically produced in response to infection, inflammation, or the presence of a foreign substance. To do this they defined key parameters to their system to model agent and environmental behavior for 1000 scenarios. 1000 scenarios were generated within the acceptable ranges defined by the field of Microbiology and Immunology, since an actual biological system is not static but dynamic. The study produced three classes of scenarios; clearance (eliminated infection), containment (controlled infection), dissemination (uncontrolled infection).

Environmental parameters included limitations such as diffusion and decay rates of chemokine/cytokine, necrosis (bodily cell death) qualifying metrics, and intra/extra-cellular bacteria growth rates. Agent-Based parameters included limitations such as speeds of agents, recruitment probability (i.e. introduction of new agents to the system), and agent lifespan.

Figure Initial Simulation State



Represented Simulation “Entities”:
S – An entry point for newly recruited Agents
BE – Extracellular Bacteria
MR – Macrophage <Resting>
MI – Macrophage <Infected>
MA – Macrophage <Active>
MC – Macrophage <Chronically Infected>
T – T-Cell

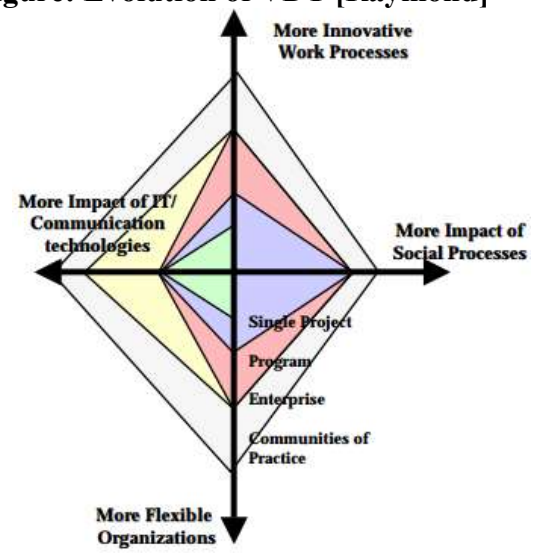
Unrepresented Simulation “Entities”:
Intracellular Bacteria – Inside Macrophage agents, their preferred environment
Chemokines – Produced by Macrophage agents, increases recruitment probability of Macrophage and T-Cell agents
Cytokines – Produced by T-Cells to activate a Macrophage <Infected>

The model is Cellular Automata-like, where the 2-D array is warped into a torus. There are two agents in the model, T-Cell and Macrophage agents. A T-Cell’s goal is to either activate an Infected Macrophage, or kill a Chronically Infected Macrophage. Where the main difference between Infected and Chronically Infected Macrophages is the threshold number of Intracellular Bacteria. Once activated the Macrophage will become hostile to both Intra/Extra-cellular Bacteria. Once a Chronically Infected Macrophage dies (via a T-Cell’s intervention, a Macrophage’s old age, or the Macrophage’s capacity being exceeded by bacteria) it will burst releasing Intracellular Bacteria into the Moore Neighborhoods around it. Agents detect chemicals within Moore neighborhoods, while diffusion of chemokines and cytokines move within von Neumann neighborhoods.

Virtual Design Team (VDT)

Virtual Design Team modeling arose as researchers in organizational theories wanted to have more dynamic testing capabilities as well as scientifically verifiable capabilities for micro-organizational and macro-organizational theories. VDT was also designed for real-world usage, where managers in fast-track projects can model the intended organizational structure, tasks, and tools to determine it efficiencies. Fast-track projects tend to shift responsibilities and set impossible sequences of task within an unreasonable timeframe. Informational bottlenecks such as one caused by a model that tends towards a worker throwing back an exception could be remedy by a preemptive modeling. An exception could be a lack in knowledge capabilities, resources, or a structural block [Olivier].

Figure: Evolution of VDT [Raymond]



Versions of VDT:

VDT-2 – Factory-Like / Strict Structure
VDT-3 – More flexible executions. Included goal incongruences between actors' goals.
VDT-4 – More complex exception handling
VDT-5 – More complex tools than in-trays
VDT-6 – N/A

Agent Types:

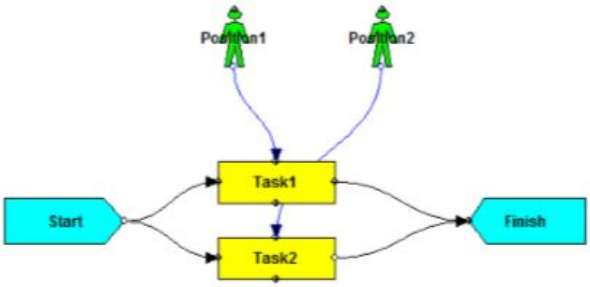
Project Manager
Sub-Team Leader
Sub-Team Member

Components:

Tasks
Actors/Agents
Organizational Structure
Communication/Additional Tools

The key to understanding the evolution of the VDT models is that it starts by laying a foundation of organizational theories (i.e. "organizational information flow physics"), while slowly adding additional complexities (i.e. "organizational chemistry") like goal incongruences. [Raymond]

Figure: Sample SimVision Simulation



SimVision UI Summary:

Agent relations can be dynamically constructed to represent any given organizational structure. Additionally, while the task flows like a "network planification" sub-teams can split to have higher granularity in execution. Tasks such as meetings may also be included in the "network planification." [Olivier]

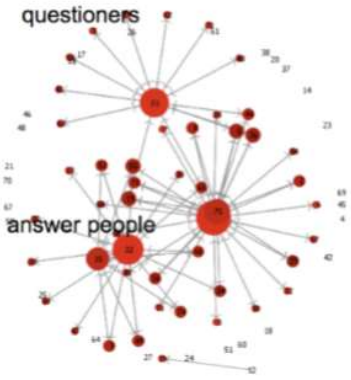
The corresponding output to these simulations can be derived from each agent's task completion rate, task duration, exceptions made, and amount of interactions made. [Olivier]

Source: Chee Siang Ang, Panayiotis Zaphiris, “Simulating Social Networks of Online Communities: Simulation as a Method for Social Design”, Human-Computer Interaction, 2009

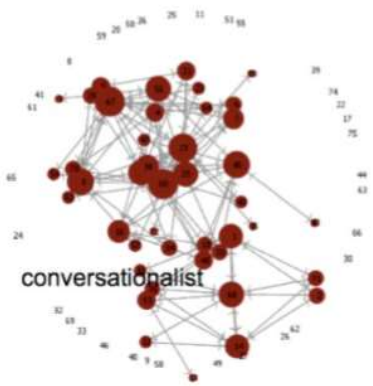
Social Network Modeling

In a joint effort between the University of Kent and the Cyprus University of Technology, a study was done on social network modeling. The goal was to follow the flow of current studies on social network modeling and use agent-based modeling, while engaging new emerging social network models such as the ones generated through MMORPG communities. In this study a relatively large guild in World of Warcraft (WoW), with 76 members at time of observation, was chosen to observe a more “developed” or “established” social network. Given that the way the platform was three interaction types emerged; help provision (fulfill request), help seeking (make request), friendly chat (idle socializing).

Through the in-game chat-log function, 1944 guild message within 30 hours were collected to generate four social matrices via thematic and content analysis. This was done by tabulating messages identifying the agent-to-agent direction, as well as the type of interaction. Three social matrices were generated, one for each interaction type and one representing the aggregate social interactions. These social matrices will be able to generate social network analysis data to compare with a simulated social network to determine statistically a social network equivalency.



(a)



(b)

[A] Social network focused on help
[B] Social nework focued on socialability

Simulation Parameters [0-10]:
Social Budget – Impact of max limit for friends
Activeness Factor – Impact of actively initiating
Cohesion Factor – Impact of local/external centric activity
Help Threshold – Ability to service help requests
Chat Threshold – Tendency to engage in chatting
Note: The above are considered the most prominent factors that affect online social networks predetermined by design of chat functionality or policy interventions (e.g. profanity)

Social Network Analysis (SNA):
Density – Size metric of local friend network
In-Degree Centrality – Metric for incoming interactions
Out-Degree Centrality – Metric of outgoing interactions
Reprocity – Metric for continued interactions
Note: The above are considered the aggregate factors (i.e. aggregate characteristics) of a social network.

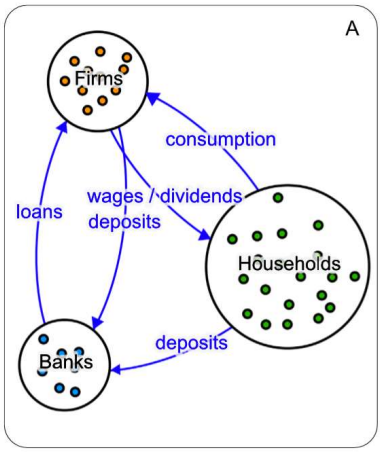
Rules:
Initiation Rules – Conditions to initiate at all
Interaction Type Decision Rules
– Conditions to initiate one the three interaction types
x3 Interaction Type Agent Behaviour Rules
– Defined behavior per interaction type

Source: Peter Klimek, Sebastian Poledna, J. Doyne Farmer, Stefan Thurner, "To bail-out or to bail-in? Answers from an agent-based model" Journal of Economics Dynamics & Control Vol 50, 2015

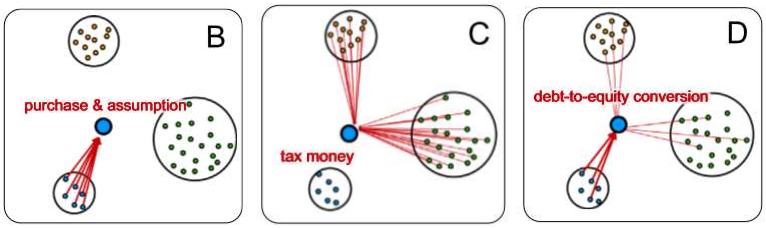
Bank Bail-in/out

After the success of the bail-in decision from the Cyprus financial crisis, economic researchers have begun to question the effectiveness of a bail-out is to a bail-in. Therefore, in response to the effectiveness of the bail-in in Cyprus, economic researchers decided to use agent-based modeling to determine the conditions where one crisis resolution is better and to also determine whether or not bail-ins should be used more. These conditions include cash flow, bank equity, labor productivity, and consumption that are determined or monitored during normal processes of between banks, firms, households (people) which is illustrated in image A.

[A] Sequence of Steps Considered per Time-Step:
1 – Firms define their labor demand and seek loans from banks
2 – Banks raise liquidity on the interbank market to service the loans to firms
3 – Firms hire or fire workers and produce goods
4 – Workers receive wages, spend their consumption budget on goods and save the remainder
5 – Firms pay out dividends, firms with negative cash go bankrupt
6 – Banks and firms repay loans
7 – Illiquid banks seek funds on the interbank market, insolvent banks are resolved.



Crisis Resolutions to an Insolvent Bank.
[B] Liquidated by a Purhase & Assumption
[C] Bail-Out
[D] Bail-In



Liquidated by a Purchase and Assumption: This solution divides the financial responsibilities as well as the negative equity among other banks that are healthy. In this way firm and household agents still retain their wealth in the failed bank, while not having to pay for failure of their chosen bank. This solution is hindered when other banks are not capable of compensating, this may have resulted from an overall instability.

Bail-Out: This solution is also an external intervention; this is typically paid out by tax payer money from the government. This is chosen by the government when the financial institution is "too big to fail," that is the failure of the given institution would have drastic effects on a regional or national level.

Bail-In: Unlike a bail-out, a bail-in is an internal intervention where creditors and depositors take a loss on their holdings (e.g. stock or bank account). This is considered more beneficial to creditors and depositors than losing their entire holdings at the failing institution.

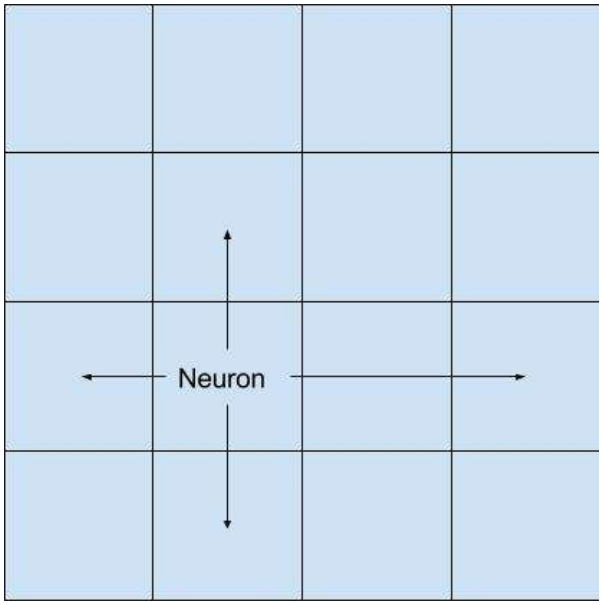
Application Computation Size Summary

Brain Grid	The simulation, based off the researcher's findings, is limited to a 100x100 2-D array on an NVIDIA Tesla C1060 GPA with CUDA which was an optimal choice for speed.
Sugar Scape	Unknown
Dengue Fever	According to the two simulation configurations (winter/summer season) 180 agents were used during the span of 60 days.
Microvascular	The simulation is set to run in 1500 time steps, with an assumed high complexity. It is an assumed high complexity because a full NxN space of agents required all of their states to be recorded, as well as those states impacting each neighboring agent along with the complex microbiological constraints.
Tuberculosis	The tuberculosis application explicitly explains that it runs 1000 scenarios (microbiological parameters configurations) upon a 2-D array of 100x100 where a cell is 2nm. These individual simulations run from 10-20min. When running all 1000 scenarios the total simulation time amounted to 200 days, where a single simulation ran for about 15 min on a Pentium Xeon 2.6 GHhz workstation.
Virtual Design Team	The source application document does not discuss computation size information. One can assume that the scale of the application does not require parallelization since SimVision, a VDT-2 implementation, is made available to managers who won't parallelize the execution. One can also assume that the maximum scale of the application ultimately depends on real-world fast-track projects at a large company. Possibly a project that last a few years, and utilizes the most dynamic version of VDT. Where the most dynamic version of VDT is one that accounts for additional virtual tools, goal incongruences, re-sequencing of tasks, and nebulas worker pool.
Social Network Modeling	Around 80 agents are used in the one simulation done, but in order to officially complete the simulation there is a need to have a statistically equivalent simulated social network. To do this the researchers generated simulation parameters until found, instead of utilizing a statistically backed approximation for iterations.
Bank Bail-in/out	The source application document does not discuss computation size information. One can guess that at the very least the simulation can run on 40 agents, since around 40 entities are present in the documents model image.

Classes of Applications

Class Neural Network

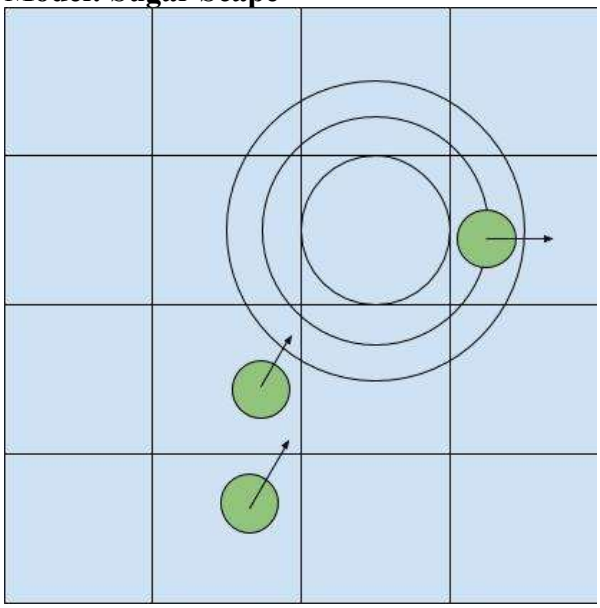
Model: Neural Network



In this simulation there are no agents only a 2-D array comprised of nodes. A node can be one of three kinds of neurons; active, inhibitory, neutral. If the node is active, then it will increase its known neighbors by adding the neighbors of neighbors. If the node is inhibitory, then it will not contribute to a synapse calculation when requested. If the node is neutral, then it will contribute to a synapse calculation.

Class Sugar Scape

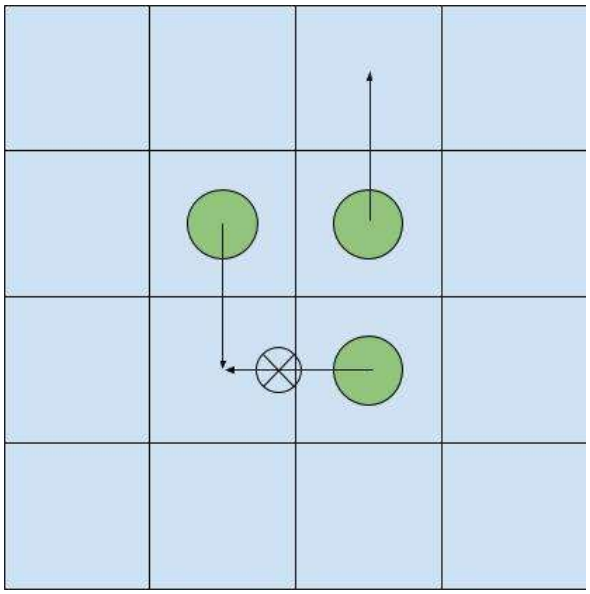
Model: Sugar Scape



In this simulation there are agents that operate on a 2-D array of nodes. Since this version of Sugar Scape is simplified, the ant agent's only function is to obtain sugar. Sugar is represented by the concentric circles illustrating a distributed pile of sugar (i.e. different concentrations of sugar). The movements of the ant agents are similar to random walk, but once they notice a concentration of sugar their migrations tend towards the source. The more sugar that is collected the more ants spawn focusing towards the sugar pile.

Class Random Walk

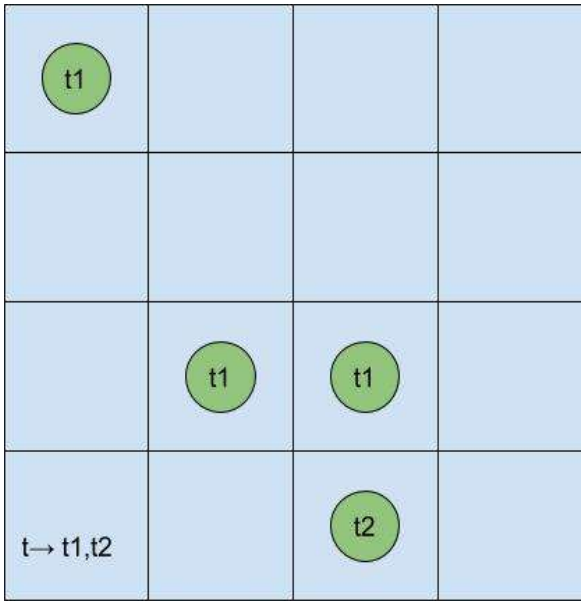
Model: Random Walk



In this simulation there are agents that migrate along a 2-D array that is warped into a torus. The main concern is to allow agents a random migration direction in von Neumann neighborhoods without collisions. Therefore, a node or agent must notify a collision in order to ensure a collision free simulation. Additionally, in this simulation agents do not spawn after the initialization.

Class Cellular Automata

Model: Cellular Automata



In this simulation there are agents that migrate, die, and proliferate upon a 2-D array of nodes. The model on the left signifies time-steps with t1 and t2, but cellular automata is not bound by such simplistic rules. Cellular automata rules could be as complex as shown earlier in the microvascular simulation, where the conditions for growth, death, and migration are defined by complex biological equations and network blood flow analysis.

Class Tuberculosis

Model: Tuberculosis

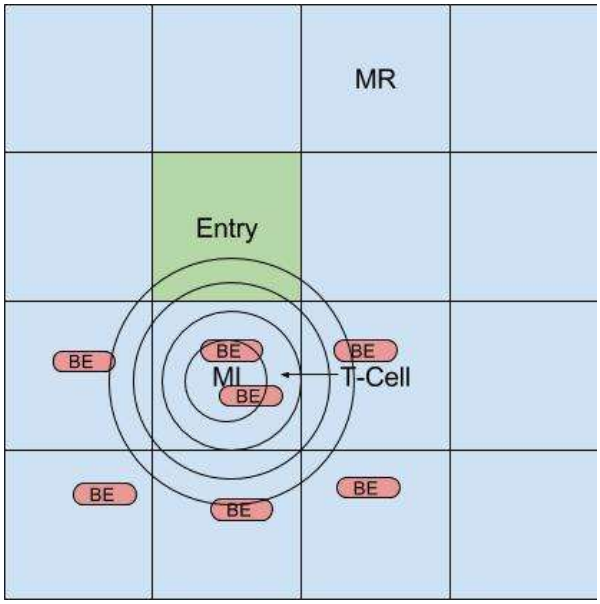
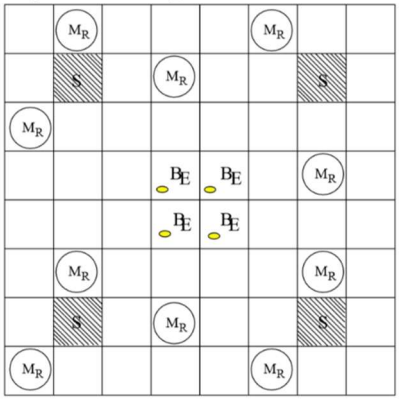


Figure: Sample Initial State [Jose L.]

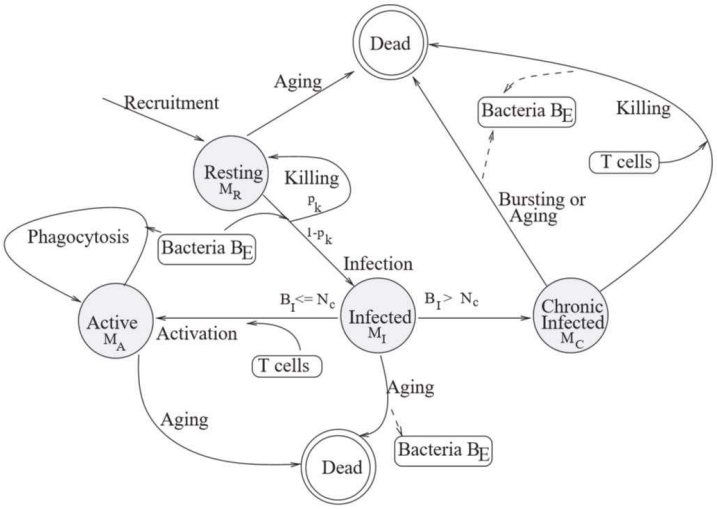
In this simulation there exists two kinds of agents (Macrophage, T-Cell) that operate on a 2-D array torus of nodes. Where a node could be an entry node, or a normal node. Nodes diffuse and decay chemicals that may later be read by agents, as well as proliferate external bacteria. These chemicals diffuse in the von Neumann neighborhoods. When these chemical concentration values reach a certain threshold around entry nodes there is a chance of spawning an agent at that node.



The initial state of the model does not have T-Cell agents and based off of the application's document will include them after 10 time-steps. Additionally, all present Macrophage agents are at rest, and are capable of either being infected or eliminate bacteria and releasing chemokine into the environment.

While extracellular bacteria are not agents, it is important to note that they are instantiated at point around the middle.

Figure: FSM for Macrophage Agents [Jose L.]

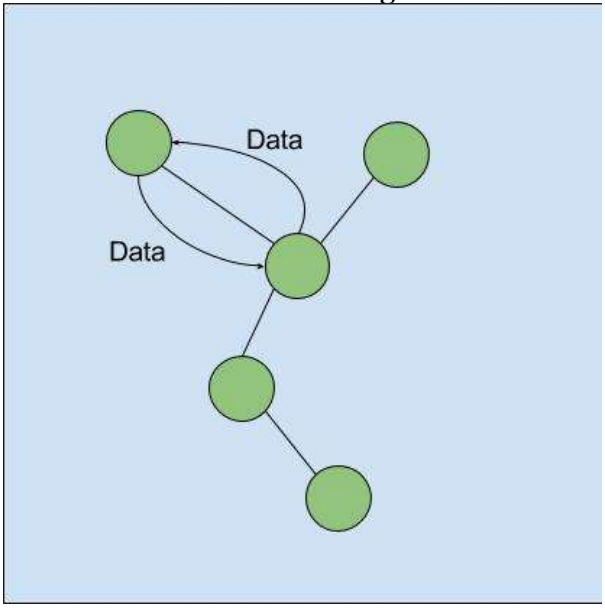


The figure to the left is the finite state machine for macrophage agents. It illustrates the model's agent-based interactions as well as the necessary conditions.

For example, one can see at what state changes release extracellular bacteria back into the environment.

Class Network of Static Agents

Model: Network of Static Agents



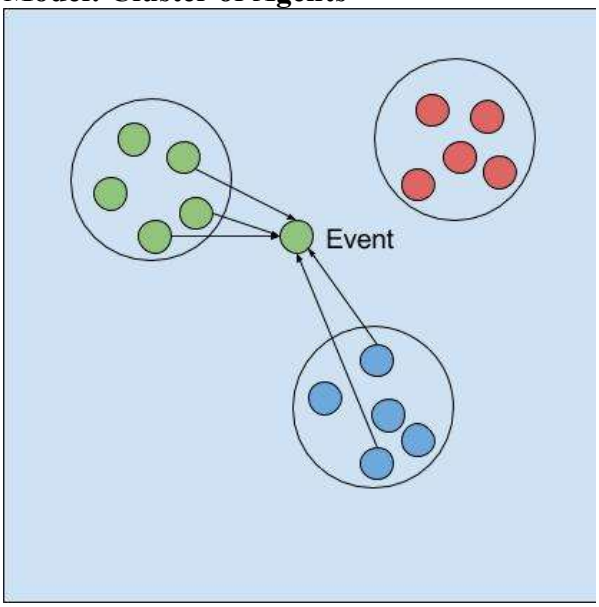
In this simulation there is no specified space or nodes for agents to act upon. According to the social network modeling application this class derives from, agents have defined local friend networks. That is a subnetwork of agents that is comprised of the agent and any other agent that shares an edge with that agent.

While there are interactions on the level of local friend networks there is also a concern over interactions made outside of an agent's local friend network. These external interactions signify an agent's willingness to interact with agents outside of their local friend network. Given that agents will interact with external agents; all agents must be connected in a complete graph in order to allow these kinds of interactions to be monitored.

According to the social network modeling application the state of the model remains, as in there are no new agents added to the system and the social network is an isolated one.

Class Cluster of Agents

Model: Cluster of Agents

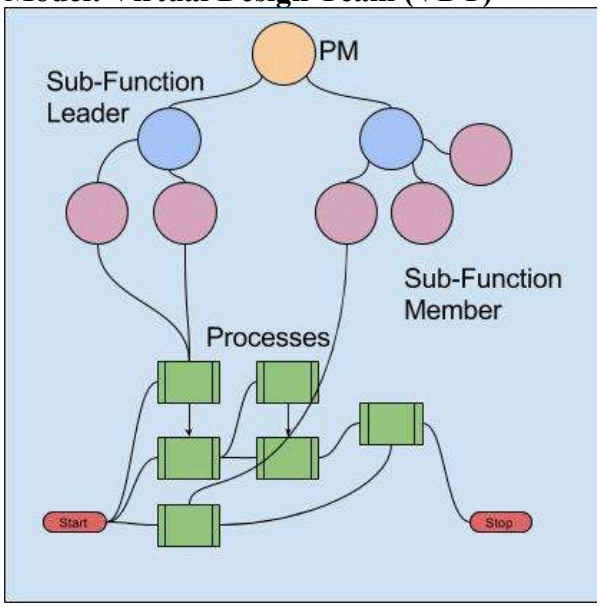


In this simulation there is no specified space or nodes for agents to act upon. Agents are simply classified into types, with interactions defined. Interactions can be between the same type of agents or different, they can even model real-world systems.

Regardless of the interactions imposed or agent types, all agents must be able to communicate with each other in order to for the model to function. Therefore, although one sees clear clustering along the types of agents in the figure to the left, the agents must actually be connected in a complete graph. That is every agent connects to every other agent.

Class Virtual Design Team (VDT)

Model: Virtual Design Team (VDT)



The model to the left is one based off of the SimVision GUI. It seems to connect an agent hierarchy tree with "network planification." At this stage it is uncertain whether or not there are nodes supporting the agents outside of processing tasks. If this model were to be implemented by MASS C++ it would have to utilize places to support agents. These places will be considered the logical in-trays of each agent and assist in informational flow for workers. This informational flow along with the initial task requirements will determine the migration of agents during the simulation.

References

General

Fukuda, M., Stiber, M., Salathe, E., & Kim, W., “CDS&E: Small: Multi-Agent-Based Parallelization of Scientific Data Analysis and Simulation.”, 2013

MASS C++, Parallel-Computing Library for Multi-Agent Spatial Simulation in C++, February 24th, 2015,
< http://depts.washington.edu/dslab/MASS/docs/MassCpp.pdf >

Rebuilding the MOSAIC, Fostering research in the social, behavioral, and economic science at the national science foundation in the next decade, 2011

Biology

Bryan C. Thorne, Alexander M. Bailey, Shayn M. Peirce, "Applying Agent--Based Modeling to Studying Emergent Behaviors of the Immune System Cells", Briefings in Bioinformatics Vol 8 (No. 4) 245-257, 2007

Bryan C. Thorne, Alexander M. Bailey, “Multi-cell Agent-based Simulation of the Microvasculature to Study the Dynamics of Circulating Inflammatory Cell Trafficking, Annals of Biomedical Engineering, 2007

Jose L. Segovia-Juarez, Suman Ganguli, “Identifying control mechanisms of granuloma formation during M. tuberculosis infection using an agent-based model”, Journal of Theoretical Biology, 2004

Luis F.O. Jacintho, Andre F.M. Batista, Terry L. Ruas, Maria G.B. Marietto, Fabio A. Silva, “An Agent-Based Model for the Spread of the Dengue Fever: A Swarm Platform Simulation Approach”, Spring Simulation Multiconference, 2010

Business / Industry

Caroline C. Krejci, Benita M. Bearmon, “Modeling Food Supply Chains Using Multi-Agent Simulation”, Winter Simulation Conference, 2012

Chee Siang Ang, Panayiotis Zaphiris, “Simulating Social Networks of Online Communities: Simulation as a Method for Social Design”, Human-Computer Interaction, 2009

Eric Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems", PNAS, 2002

Kent D. Miller, “Agent-Based Modeling and Organization Studies: A critical realist perspective”, Organization Studies, 2015

M.-H. Chang, J.E. Harrington Jr, "Chapter 26: Agent-Based Models of Organizations", Handbook of Computational Economics, 2006

Olivier Kooy, "Understanding the causal relations in organizational structures of project teams", System Engineering, Policy Analysis and Management, TU Delft, Master Thesis, 2012

Raymond E. Levitt, “VDT Computational Emulation Models of Organizations: State of the Art and Practice”, Stanford University, 2000
< http://web.stanford.edu/group/VDT/VDT.pdf>

Thomsen, Jan, Raymond E. Levitt, John C. Kunz, Clifford I. Nass, Douglas B. Fridsma, “A Trajectory for Validating Computational Emulation Models of Organizations” Journal of Computational & Mathematical Organization Theory, 1999

Economics / Social Sci.

Christian Gloor, Pascal Stucki, Kai Nagel, "Hybrid techniques for pedestrian simulations", 4th Swiss Transport Research Conference, 2004

Duncan Foley, J. Doyne Farmer, "The economy needs agent-based modelling", Nature Vol 460, 2009

Herbert Gintis, "Long--range Research Priorities in Economics, Finance, and the Behavioral Sciences", Santa Fe Institute, 2010

Peter Klimek, Sebastian Poledna, J. Doyne Farmer, Stefan Thurner, "To bail-out or to bail-in? Answers from an agent-based model" Journal of Economics Dynamics & Control Vol 50, 2015

Yasuki Iizuka, Katsuya Kinoshota, Kayo Iizuka, "Agent Based Disaster Evacuation Assistance System", Information Engineering Express International Institute of Applied Informatics Vol 1, 2015