Modeling Contagion

Scalable Game Design Project

Thought Amplifier

Funded in part by AgentSheets®
STEM Modeling - Overview

- Why model?
  - Motivational examples
    - Contagion
    - GleamViz
  - Main purpose of modeling: Prediction
  - Make a model then map it to ‘reality’

- STEM model vs. game
  - STEM model asks a question
  - STEM model collects data for analysis
  - STEM model is ‘hands off’ the keyboard/mouse
STEM Modeling - Agenda

- Basic Contagion model – 3 phases
- Generalization of the model to other areas
- Example extensions to the model
- Other topics (as desired)
- Make a model [extension]
Stage 1: A Very Basic Model

- **Specification**: Persons move randomly in (on) the City
- Create Person and City agents
- Create world: “World 1”
- Populate World 1 with City agents (completely).
- Place one Person agent in World 1
- One rule: Person agent: move random in City
- Run the simulation and observe
- Create world: “World 2”
- Place an array (several rows and columns) of Person agents in World 2
- Run the simulation and observe.
Stage 2: Healthy & Sick -1

- **Specification**: Healthy and sick persons. Healthy persons move randomly in the City. A healthy person next to a sick person has 50% chance to get sick. A sick person has 50% chance to recover.

- New shapes of Person: Healthy_Person, Sick_Person

- New behavior
  - (1st rule) If Healthy_Person is next to Sick_Person, with 50% chance, get sick
  - Otherwise, if Healthy_Person, move randomly in the City
  - If Sick_Person, with 50% chance, recover.

- Run simulation and observe
- Change probability to recover to 0% and re-run
- Reload world and single step the simulation.
Stage 2: Healthy & Sick - 2

**Observations:**
- Healthy_Persons move; Sick_Persons do not
- Eventually (usually), all Sick_Persons recover
- When probability to recover is 0%, eventually no Healthy_Persons remain
- When simulation is single-stepped, Healthy_Person agents not next to a Sick_Person get sick. Why?
- It is difficult to keep track of the number of Healthy_Person and Sick_Person agents.
Conclusions:

- Make variables of probabilities for ease of control
- Maintain statistics and plots for visualization ease
- Make simulation terminate when no more sick persons
- Understand why the simulation does not match the specification in terms of Healthy_Person agents getting sick only when next to a Sick_Person; resolving requires behavior (programming) change
Stage 3: Healthy & Sick - 1

- **Specification changes:**
  - Probabilities for getting sick and recovering will be simulation properties (variables): @Get_Sick, @Recover
  - Add Monitor agent to maintain statistics (simulation properties): @Cycles, @Healthy_Persons, @Sick_Persons, @Total_Persons
  - Separate Person behavior into two basic activities, Perceive and Act
  - Rules in Perceive detect when a change in state – healthy vs. sick – is required based on conditions
  - Rules in Act cause state changes to take effect or continue behavior in the current state
  - Use agent attributes in Person agent to indicate when state change should occur
  - Add scripting rule in Monitor to synchronize the Perceive and Act behavior of all Persons
Stage 3: Healthy & Sick - 2

- Run the simulation:
  - Set probabilities to get sick and to recover to 50% each
  - Run the simulation with different probabilities to recover, including 0%
  - Record observations
Extensions to Contagion Model

- Minimum time for sickness to last
- Fatality (Sick_Person does not recover)
- Immunity (after recovering from sickness)
- Sick_Person movement
- Hospital (Sick_Person seeking)
- Doctor (seeking Sick_Person)
- Other “real-life” factors in Contagion
- Generalization to other models (e.g., crystallization)