

# Performance & Computational Resource Considerations

Nathaniel Osgood

Using Modeling to Prepare for Changing  
Healthcare Needs

Duke-NUS

April 16, 2014

# The Computational Burden of ABM

- Agent-Based models impose large computational burden
- Key factors:
  - “More moving parts”: Lots of values to calculate and manipulate
  - Requirement of running multiple realizations

# ABM and Computational Resource Use

- The computational burden of Agent-Based Models limits value delivered
  - Opportunity cost: Reduces time spent in exploration of model results & insights gained
  - Limited time => Less thorough exploration of parameter space => Reduced quality of calibration
  - Inhibits adoption

# Significant Computational Factors

- Event-limited performance
- Statistics
- Visualization
- Network Construction
- Output of data

# Significant Computational Factors

- Event-limited performance
- Statistics
- Visualization
- Network Construction
- Output of data

# Event-Limited Performance

- AnyLogic utilizes an “event driven” scheduler
  - The more events, the more the scheduler has to “wake up” to do things
  - In addition to the work to be done, there is some bookkeeping involved in the occurrence of an event
- One important performance saving: When sending messages, Replacing `a.send` by `a.deliver`
  - Note that this is not possible in all cases, and can lead to infinite loops when agent A `delivers` to B, and B `delivers` to A
- If there is greater occurrence of events (either explicit or implicit in e.g. transitions or messages), this will generally adversely affect performance



# Hands on Model Use Ahead



Load Sample Model:

**SIR Agent Based**

(Via “Example Models” and then “How To Models” under “Help” Menu)

# Suggestions

- Permit disabling of visual elements
  - Use parameter variation experiment or set update freq
- Lower event frequency
  - Use dynamic events
  - Do more on firing of each event
  - Disable when not required
- Where possible, use “bookkeeping” on transitions (increase/decrease counts) rather than statistics
- Use a profiler to find where spending time
- Send events only for “infecting dose” (rather than exposure), where possible
- Use deliver rather than send

# Example of Reducing Events: Messages

- Sometimes there are simple ways to reduce event occurrence
- Example replacement
  - Worse performance: Sending “exposure” messages with rate  $\alpha$ , each having a likelihood  $\beta$  of infection upon receipt
  - Better performance: Send “infect” messages with rate  $\alpha\beta$
- Such simplifications are context-specific
  - For example, this transformation is much harder if the likelihood of infection given exposure varies by individual

# Significant Computational Factors

- Event-limited performance
- **Statistics**
- Visualization
- Network Construction
- Output of data

# Statistics

- AnyLogic's capacity to define "Statistics" over a population provides an easy way to compute population statistics
- Downside: Each computed statistic requires a full iteration through each member of the population
- Example: Classifying people into each of 17 age categories using "Statistics" requires 17 passes through the population!
  - This could in principle be done in a single pass – with each individual just incrementing different bins of a histogram

# Significant Computational Factors

- Event-limited performance
- Statistics
- **Visualization**
- Network Construction
- Output of data

# Visualization

- The visual presentations of elements of a model takes considerable
  - Time
  - Memory
- Example: Dynamic properties
- Disabling visualization can lead to much faster operation
- Options
  - Creating a model without “presentation” properties
  - Set running settings so that infrequent updates
  - Running the model using “runFast()”

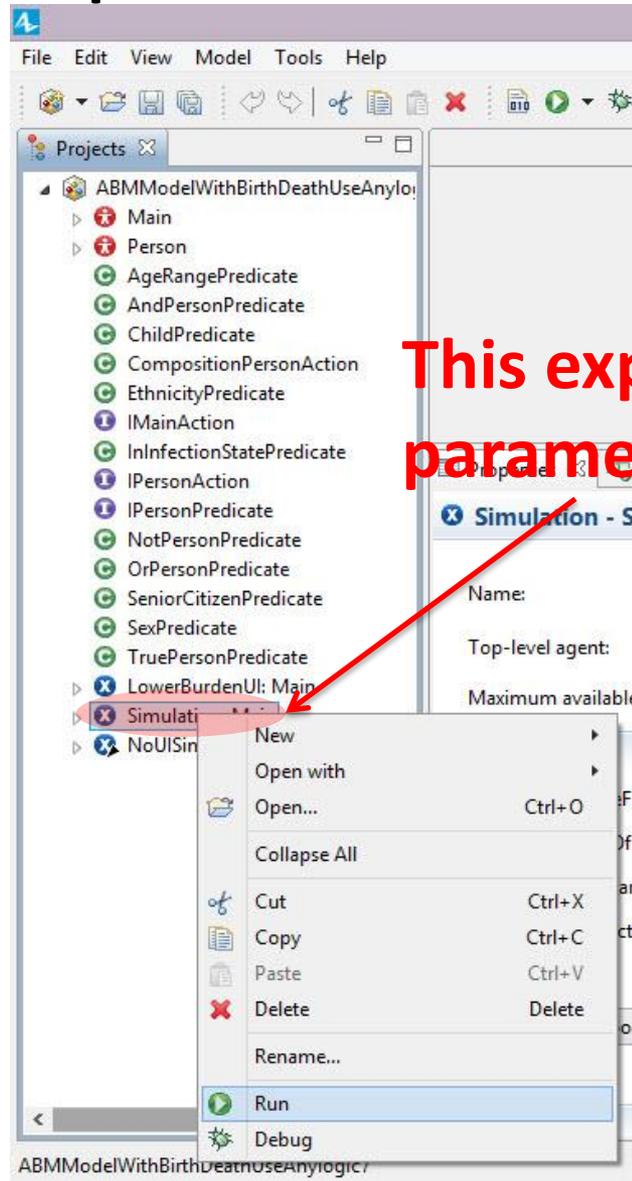


Hands on Model Use Ahead



Load ***Agent-Based Model with Birth Death***

# Run the Experiment “Simulation” & Time Elapsed Duration until Completes



**This experiment uses the default parameter values**

# Running the Model using RunFast

- Create a Parameter variation experiment with 1 realization (no exploration of parameter space)
- Either
  - Run from the “Run” Menu item
  - Create a button to run
- Advantages
  - Simple (no custom coding)
  - Fastest
  - Model can still contain dynamic presentation properties & display visualizations in other experiments
- Disadvantages: No option to visualize

# Enabling Faster Running by Settings & Displaying only when Required

- Create a simulation with slow update & don't display by default
- Advantages: Retain option of visualization where desired
- Disadvantages:
  - A bit of custom coding required
  - Slower
  - Memory is still allocated for presentation elements

# Method 1: Adding an Experiment without Visualization via Param Variation

**New Experiment**

Select an experiment type, specify a name and choose a root (top-level) active object.

Name:

Main active object class (root):

Experiment Type:

- Simulation
- Optimization
- Parameter Variation**
- Compare Runs
- Monte Carlo
- Sensitivity Analysis
- Calibration
- Custom

Performs multiple model runs varying one or more parameters, optionally using replications. You can later add arbitrary UI to this experiment.

Copy model time settings from:

< Back    Next >    **Finish**    Cancel

# New Experiment

The screenshot displays the AnyLogic Professional software interface. The main window is titled "NoUISimulation" and shows a large grid area with a blue line graph. The left sidebar contains a "Projects" tree view with the following structure:

- ABMModelWithBirthDeathUseAnylo...
  - Main
  - Person
    - AgeRangePredicate
    - AndPersonPredicate
    - ChildPredicate
    - CompositionPersonAction
    - EthnicityPredicate
    - IMainAction
    - InInfectionStatePredicate
    - IPersonAction
    - IPersonPredicate
    - NotPersonPredicate
    - OrPersonPredicate
    - SeniorCitizenPredicate
    - SexPredicate
    - TruePersonPredicate
  - Simulation: Main
  - NoUISimulation: Main**

The right sidebar is the "Palette" window, showing a "Presentation" section with various drawing tools:

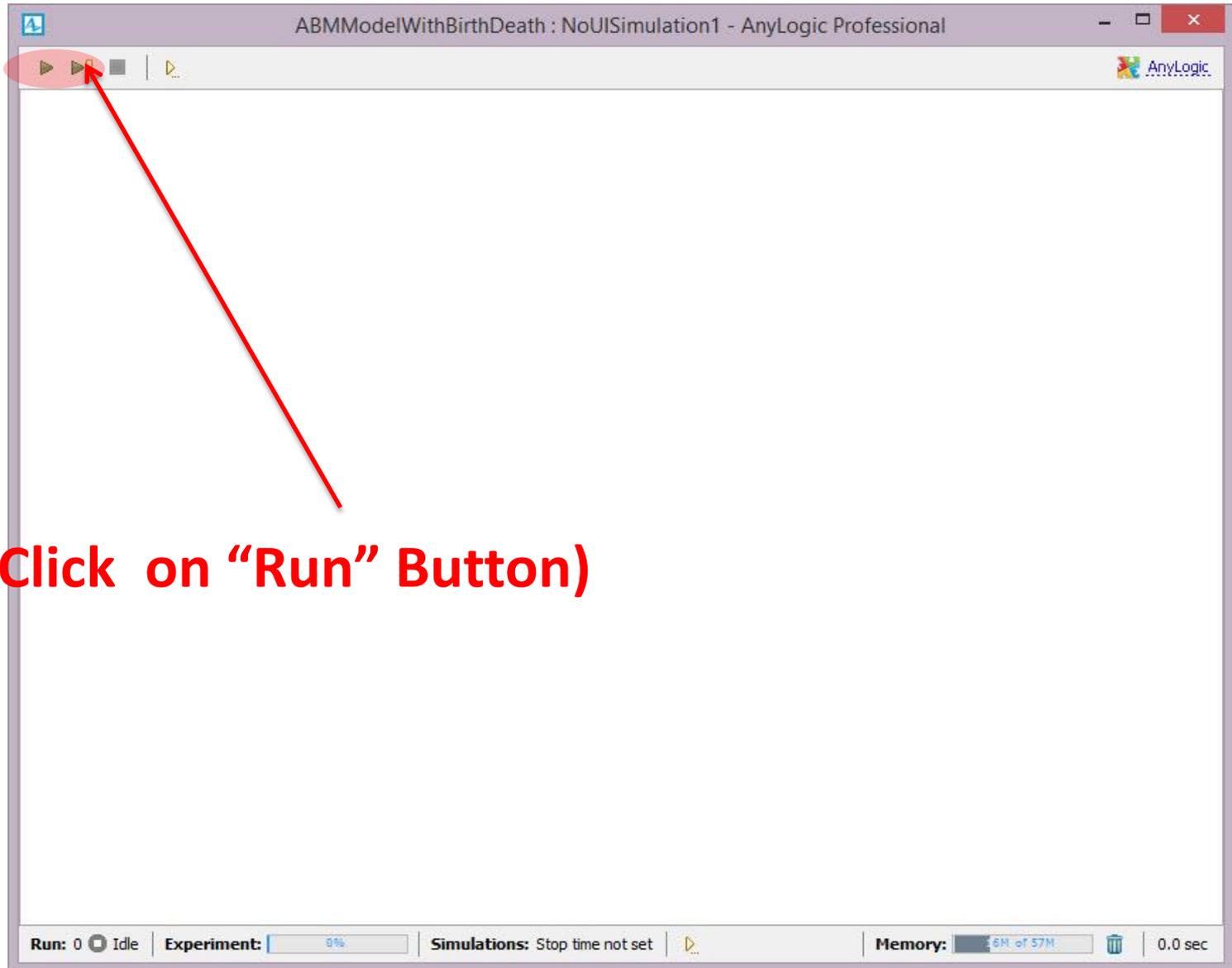
- Line
- Polyline
- Curve
- Rectangle
- Rounded Rect...
- Oval
- Arc
- Pixel
- Aa Text
- Image
- Group
- View Area
- 3D
  - 3D Window
  - 3D Object

The bottom panel is titled "NoUISimulation - Parameter Variation Experiment" and contains the following configuration options:

- Name: NoUISimulation  Ignore
- Top-level agent: Main
- Maximum available memory: 64 Mb
- Create default UI

The bottom status bar shows "Time units: days" and "X=...-2".

# Running & Timing the Experiment (b/c parameter variation exp., uses runFast())



Click on "Run" Button)

# Comparing Timings

- The time required to run the model in the experiment that avoids a UI should be a fraction of that required with the UI

# A New Experiment

**Experiment**  
Select an experiment type, specify a name and choose a root (top-level) active object.

Name:

Main active object class (root):

Experiment Type:

- Simulation
- Optimization
- Parameter Variation
- Compare Runs
- Monte Carlo
- Sensitivity Analysis
- Calibration
- Custom

Performs model runs with specified parameters, supports virtual and real-time modes, animation, and model debugging

Copy model time settings from:

< Back   Next >   **Finish**   Cancel

# Insert Code Into “Before Each Experiment Run” in “Advanced” tab for Experiment

The screenshot displays the AnyLogic Professional software interface. The main window shows a diagram titled "ABM Model With Birth Death Large Burden UI" with the subtitle "Experiment setup page". The diagram is a simple rectangle on a grid. The left sidebar shows a project tree with the following structure:

- ABMModelWithBirthDeathUseAnylo...
  - Main
  - Person
    - AgeRangePredicate
    - AndPersonPredicate
    - ChildPredicate
    - CompositionPersonAction
    - EthnicityPredicate
    - IMainAction
    - InInfectionStatePredicate
    - IPersonAction
    - IPersonPredicate
    - NotPersonPredicate
    - OrPersonPredicate
    - SeniorCitizenPredicate
    - SexPredicate
    - TruePersonPredicate
  - LowerBurdenUI: Main**
  - Simulation: Main
  - NoUISimulation: Main

The bottom panel shows the configuration for the "LowerBurdenUI - Simulation Experiment". The "Advanced Java" tab is selected, and the "Before each experiment run:" section contains the following code:

```
getEngine().setRealTimeMode(false);
getPresentation().getPanel().setFrameManagementAdaptive(false);
getPresentation().getPanel().setFrameRate(0.05);
```

The "Initial experiment setup:", "Before simulation run:", and "After simulation run:" sections are currently empty. The "Imports section:" and "Additional class code:" sections are also empty.

The status bar at the bottom indicates "Time units: days" and "X=...17".

# Code to Insert

Before each experiment run:

```
getEngine().setRealTimeMode(false);  
getPresentation().getPanel().setFrameManagementAdaptive(false);  
getPresentation().getPanel().setFrameRate(0.05);
```

# Run the Experiment

The screenshot displays the AnyLogic Professional software interface. The main window is titled "LowerBurdenUI - Simulation Experiment". The interface includes a menu bar (File, Edit, View, Model, Tools, Help), a toolbar, and a "Projects" pane on the left. The "Projects" pane shows a tree view of the simulation model, with "LowerBurdenUI: M" selected. A context menu is open over this selection, with the "Run" option highlighted. The "Properties" pane shows the following settings:

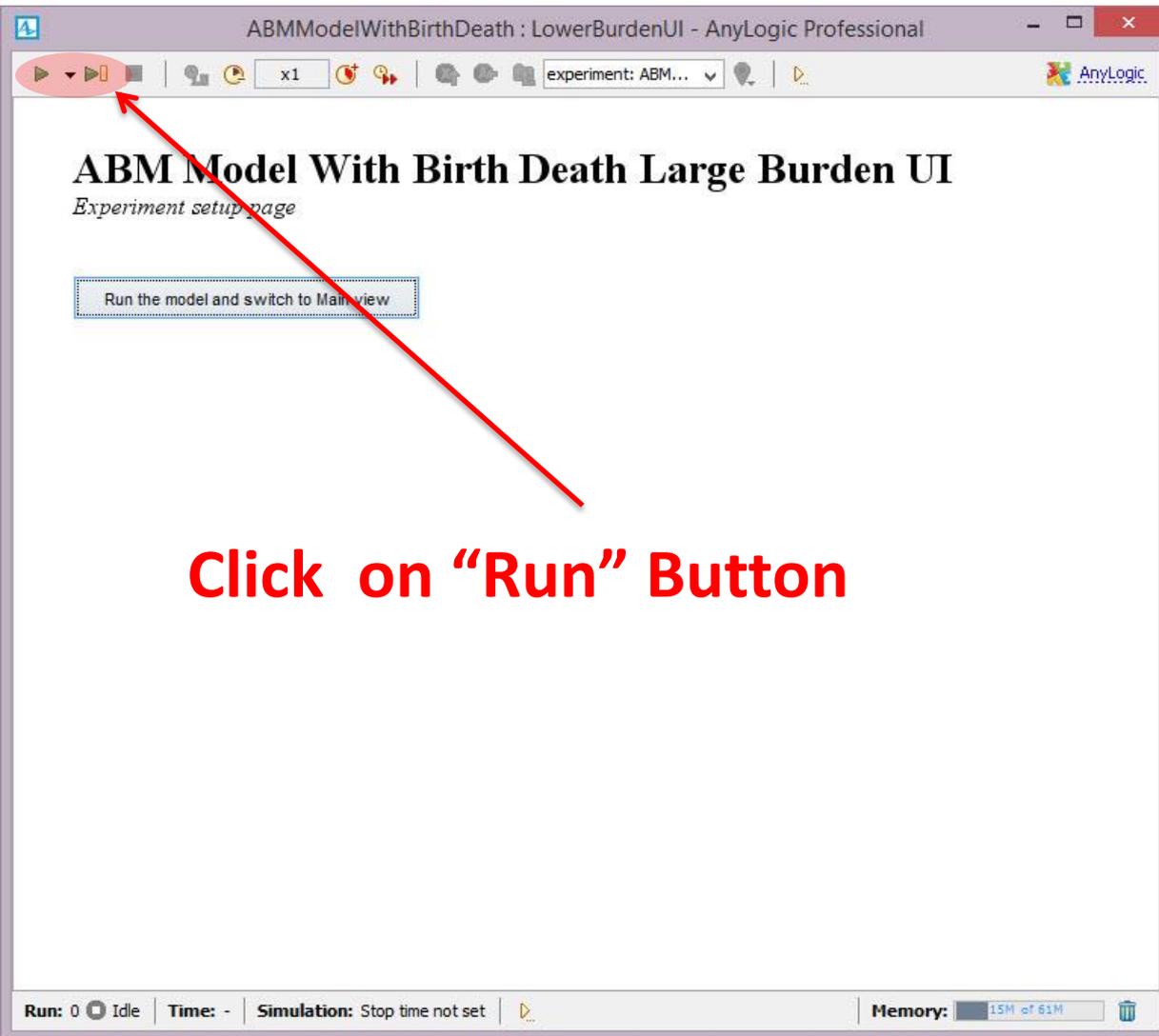
- Name: LowerBurdenUI  Ignore
- Top-level agent: Main
- Memory: 64 Mb

The "Properties" pane also displays a list of parameters for the simulation:

Parameter	Value
15 /* half a distance outside of perimeter */	15
0.01	0.01
100	100
0.10	0.10
80.0	80.0

The status bar at the bottom indicates "Time units: days".

# Run & Time Model without Displaying UI



**Click on “Run” Button**

ide of perimeter \*/

# Without the UI, Model should be Fast!

- Note that the inserted code allows this trick to work
  - Not viewing the UI in the original simulation experiment (which is missing this inserted code) will not similarly shorten the running time!

# Run the Experiment

The screenshot displays the AnyLogic Professional software interface. The main window is titled "AnyLogic Professional" and contains a menu bar (File, Edit, View, Model, Tools, Help) and a toolbar with various icons. On the left, a "Projects" pane shows a tree view of the model structure, including "ABMModelWithBirthDeathUseAnylo...", "Main", "Person", and several predicates and actions. A context menu is open over the "LowerBurdenUI: M" element, with the "Run" option highlighted. The "Properties" pane on the right shows the configuration for "LowerBurdenUI - Simulation Experiment", including fields for Name, Top-level agent, and a list of parameters with values.

**Properties: LowerBurdenUI - Simulation Experiment**

Name: LowerBurdenUI  Ignore

Top-level agent: Main

Memory: 64 Mb

Parameters:

- Parameter 1: 15 /\* half a distance outside of perimeter \*/
- Parameter 2: 0.01
- Parameter 3: 100
- Parameter 4: 0.10
- Parameter 5: 80.0

Time units: days

# Run & Time the “LowerBurdenUI” Model *with* the Visualization

The screenshot displays the AnyLogic Professional interface for an experiment titled "ABM Model With Birth Death Large Burden UI". The window title is "ABMModelWithBirthDeath : LowerBurdenUI - AnyLogic Professional". The interface includes a toolbar with various simulation controls, a main workspace area, and a right-hand palette. The main workspace contains the text "ABM Model With Birth Death Large Burden UI" and "Experiment setup page". A button labeled "Run the model and switch to Main view" is highlighted with a red oval, and a red arrow points to it from the bottom. At the bottom of the window, a status bar shows "Run: 0", "Idle", "Time: -", "Simulation. Stop time not set", and "Memory: 15M of 61M".

**Click on this button to run the model with a visualization**

# Visualization with New Settings

- The custom settings should significantly lower the time required to run the simulation when compared to the default settings

# Significant Computational Factors

- Event-limited performance
- Statistics
- Visualization
- **Network Construction**
- Output of data

# Network Construction

- AnyLogic's Scale Free network requires a long time to run
- We have found gains by implementing the Barabasi-Albert algorithm ourselves

# Significant Computational Factors

- Event-limited performance
- Statistics
- Visualization
- Network Construction
- **Output of data**

# Database Output

- Batch up data to send to the database
  - Send in one big call to database, rather than multiple calls
- Use local database
- Record smaller subset of data
- Record less frequently
- Record fewer types of data

# Model Space Demands

- Models with large populations can require much space
- I believe that the space demands can be particularly large when visualization is enabled
- You can enable space available for models in the “experiment” area
- Ways to reduce space demands
  - Accumulate less data (less frequently/fewer data items)
  - Write data out rather than accumulating in datasets

# Exploiting Opportunities for Concurrency

## Using Distributed Processing

- ABM offers opportunities for parallel processing
- Two particularly manifest opportunities for concurrency require different levels of sophistication to exploit
  - “Embarassingly parallel” & easy to exploit: Concurrency between model realizations. One can readily run different realizations of a model in parallel (e.g. on different machines) & harvest results
  - Also parallelizable, but harder to exploit: Concurrency opportunities between distinct agents. While agent processing could in principle be parallelized, dependencies between agents (e.g. via message sending & joined flows) makes this more challenging to exploit.