Critical Role of Network & Population Dynamics

• We have introduced the basic mechanisms for
  – Creating populations of pre-specified sized
  – Creating network from a pre-specified set of network categories

• However,
  – Open populations (e.g. with immigration, death, birth) are the norm
  – Research suggests that many types of networks dynamics (serial partnerships, differing contact durations) are important to infection dynamics
AnyLogic’s Support of Network & Population Dynamics

• Fortunately, AnyLogic provides strong support for
  – Adding & removing population members
  – Adding & removing connections

• However, this support does not yet have direct graphical interface support or specification
  – using this support does require that you call “methods” to accomplish this
AnyLogic Support for Changing Populations

• Adding to population
  – add_populationname(parameters)
    • Allow explicit specification of agent parameter values
  – add_populationname()
    • Uses population specification of agent parameter values

• Deleting from population
  – remove_populationname(agentToBeRemoved)
AnyLogic methods for Adding & Deleting Connections

- `agentA.connectTo(agentB)`
  - Connects `agentA` to `agentB`
  - NB: Connections are assumed to be undirected and symmetric (i.e. if `agentA` is considered to be connected to `agentB`, then `agentB` is considered to be connected to `agentA`)

- `agentA.disconnectFrom(agentB)`
  - Disconnects `agentA` and `agentB` from each other

• For more details and additional methods, see the slides for the Networks lecture
Hands on Model Use Ahead

Load Previously Built Model: MinimalistNetworkABMModel

Suggest Saving as “MinimalistNetworkABMModelWithInterfaceDrivenPopulationDynamics”
Set Small Population Size (5)
Set Distance Based Network with High Connection Range Threshold
To Main: Add Button to Request Adding Population Member

```java
add_population(); // add with the population-given parameters
environment.applyNetwork(); // recompute the new network with this person added
```
To Person’s “Oval”, Add a “Handler” to Delete a Person if their Node is Clicked
Hands on Model Use Ahead

Load Provided Shared Model: ABMMModelWithBirthDeath
Adding an Immigrant to the Model Population

```java
addPopulation(uniform(MeanLifespan, Person.RandomEthnicity), Person.RandomSex, uniform()) < prevalenceOfInfectionAmongImmigrants;
```
Add Population Options – Note Customization to Context
Removing a Population Member

FinalizeDeath - Function

```
traceln("Population member " + this + " has died.");
get_Main().remove_Population(this);
```
Establishing Baby’s Connection
Looping over Connections

// now establish links between the baby and all of the mother’s connections

if (mother.getConnections() != null) // guard against a mother with no connections
for (Agent a : mother.getConnections())
|
| Person p = (Person) a;
| offspring.connectTo(p);
|

// Finally, establish a link between the baby and the mother
// (we do this last so we don’t have to worry that one of
// the mother’s connections is to this offspring!
offspring.connectTo(mother);

// note that the “mother” property of the baby has already been set when it was created
Code to Perform Birth

Function body:

```java
Person mother = this;
Person offspring = get_Main().add_Population((double) 0, ethnicity, RandomSex(), this.IsInfected(), mother,
    treeFn("A baby has been born! Baby's id is " + offspring + " while the mother is " + this);
// establish connections of infant
EstablishOffspringConnectionsBasedOnMothersConnections(offspring, mother);
// now position the baby to be close to the mother (otherwise leads to stretching of mother's connections
EstablishOffspringLocationBasedOnMothersLocation(offspring, mother);
```
Establishing Baby’s Connection
Looping over Connections

```java
// now establish links between the baby and all of the mother's connections
if (mother.getConnections() != null) // guard against a mother with no connections
    for (Agent a : mother.getConnections())
        Person p = (Person) a;
        offspring.connectTo(p);

// Finally, establish a link between the baby and the mother
// (we do this last so we don't have to worry that one of
// the mother's connections is to this offspring!
offspring.connectTo(mother);
// note that the "mother" property of the baby has already been set when it was created
```
Setting Offspring Location