# DiscreteSpatialPhyloSimulator

**S. J. Lycett**

**14 October 2013**

# Introduction

The DiscreteSpatialPhyloSimulator is a java program which uses a stochastic individual based model to simulate infection over a structured population. The individuals are organised into “Demes”, and Demes can be connected together in a variety of ways. Within a Deme all individuals have the same parameters, but each Deme could have its own particular parameters. For example, the DiscreteSpatialPhyloSimulator can be configured for:

1. Phylogeography – each Deme is a location, individuals may migrate between locations.
2. Zoonoses – each Deme is a species, infections may be transmitted between Demes, but the hosts themselves do not move.
3. Networks – each Deme contains one host, infections may be transmitted between neighbouring Demes.

# Inputs

The DiscreteSpatialPhyloSimulator is configured via an XML configuration file, which contains four sections. Examples are included with the java code, and in the appendix.

General: Specify the output file names and number of replicates.

Sampler: Specify the type of sampled output; choices are to sample everything just before recovery, or to take a random sample of the infected population at each (specified) time point.

Demes: For each deme specify the number of individuals, and connectivity to other demes.

Population: Specify population level parameters (that apply to all demes), including infection model type (e.g. SIR), infection parameters (appropriate to the infection model type) and Deme type (migration between demes or infection over the network).

# Outputs

For each replicate run the outputs are:

### Population Log:

A \*.csv file containing the simulation time and number of individuals in each state for each deme. (There are also two diagnostic columns, one for the number of events in the queue and the other containing the number of active hosts – i.e. hosts able to generate events.)

### Events Log:

By default this \*.csv file contains only the infection events, and has the headers: EventType, CreationTime, ActionTime, FromDeme-FromHost, ToDeme-ToHost. EventType will usually refer to the infection event; creation time is the time at which the event was created by a host; action time is the time at which the event occurred; and the from and to columns detail which individual infected which. Consequently to plot the true transmission network (including “un-sampled” cases), the infection events log can be loaded into e.g. Cytoscape and a directed network created from the FromDeme-FromHost and ToDeme-ToHost columns.

### Full Tree:

The full tree, detailing who infected who (including “un-sampled” cases) is output in newick format. The internal nodes of the tree represent events, and the tips of the tree represent either samples or “stubs” – currently infected individuals which may cause future infection events. The tree node names have the format : (HostID)\_(DemeName)\_(EventNumber); if the node is sampled (samples are always tips) then the format is: (HostID)\_(DemeName)\_(EventNumber)\_sampled.

### Pruned Tree:

The pruned tree is the full tree with all the branches that do not end with a sample pruned away. Note that it contains internal nodes with only one child – these single child nodes are just before the sampled tips.

### Binary Tree:

The binary pruned tree is the pruned tree but with the single child nodes removed; and is the time resolved phylogenetic tree suitable for comparison with other phylogenetic trees or for use in a sequence simulator.

# Current capabilities

Infection Model Types: SI, SIR, SEIR

Deme Types: Migration of Infecteds, Infection over Network

In Built Deme connectivity: Full, Random, Star, Line

Network Deme connectivity: Supplied via XML configuration file

Sampling schemes: Just Before Recovery Sampler (sample everything), Random Sampler (sample a proportion of the infected population at specified times)

# In Development & Extensions

Migration: Migration of infecteds is only partly implemented and not tested. Also Infection over Network requires further work for networks with demes of more than one host.

Sampling schemes: Implement more sampling schemes – perhaps a “cull” when an infected node is sampled. Note that the Random Sampler is not tested yet.

Host states: At present a host can be Susceptible, Exposed, Infected, Recovered, Immune – although note that only SI, SIR, SEIR models are implemented, and only SI and SIR have been tested. However, it would be possible to extend the host class to allow the addition of one or more virus objects to the host. This would allow the direct simulation of heritability (virus object has a continuous value, e.g. Double) or co-infection (virus object has a name). If the virus object state was included in the host name then the output trees would contain all the relevant information.

Parallelisation: The simulator is composed of demes which contain one or more hosts. An option for parallelisation would be to put sets of demes on different processors (might want to consider some smart way to do this). It would also be possible to extend the Deme class to allow the hosts within a deme to be connected via a network – so here a deme would be a cluster / risk group with fewer connections to other demes. However, since it is typical that many replicate runs are performed, “embarrasing” parallelisation over different runs would be the first thing to do; this could be achieved in a shell script since the DiscreteSpatialPhyloSimulator is a command line jar file, but it might be worth developing a java controller which task farms to different DiscreteSpatialPhyloSimulators and collects the outputs to one set of files.

# Example Cases

## SimpleSI

One Deme of 1000 individuals, infection model is Susceptible -> Infected.

Figure 1: Number of Infected over time for simple SI model. Red = DiscreteSpatialPhyloSimulator stochastic simulation, Grey Dash = Deterministic results from R.

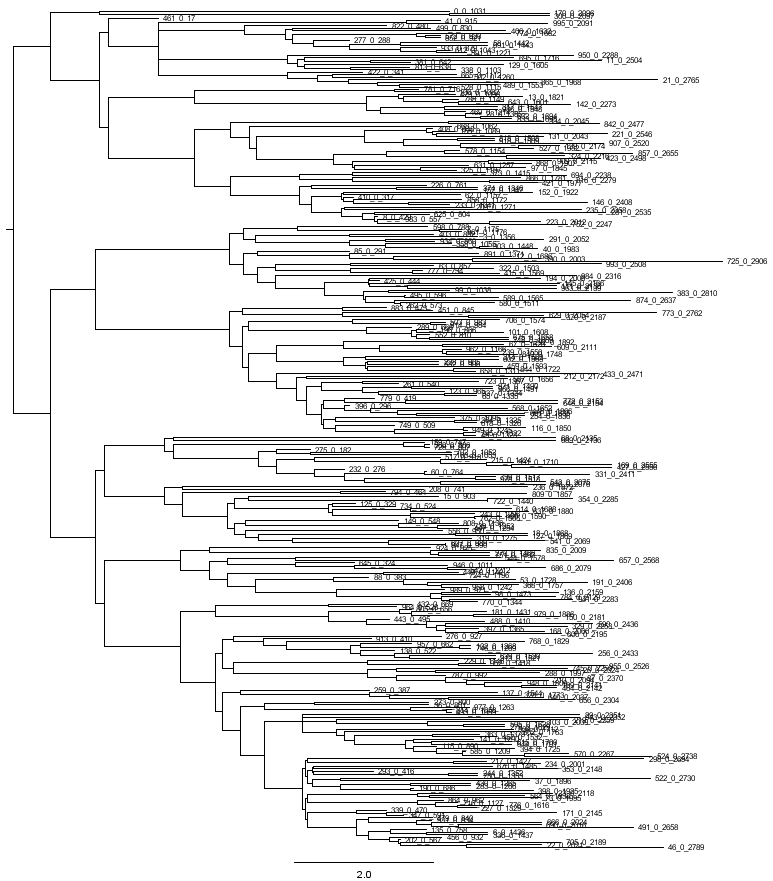


Figure 2: Phylogenetic Tree from simple SI model (Binary Pruned tree)

## SimpleSIR

One Deme of 1000 individuals, infection model is Susceptible -> Infected -> Recovered.

Figure 3: Number of Infected over time for simple SIR model. Red = DiscreteSpatialPhyloSimulator stochastic simulation, Grey Dash = Deterministic results from R.

Figure 4: Number of Susceptible over time for simple SIR model. Red = DiscreteSpatialPhyloSimulator stochastic simulation, Grey Dash = Deterministic results from R.

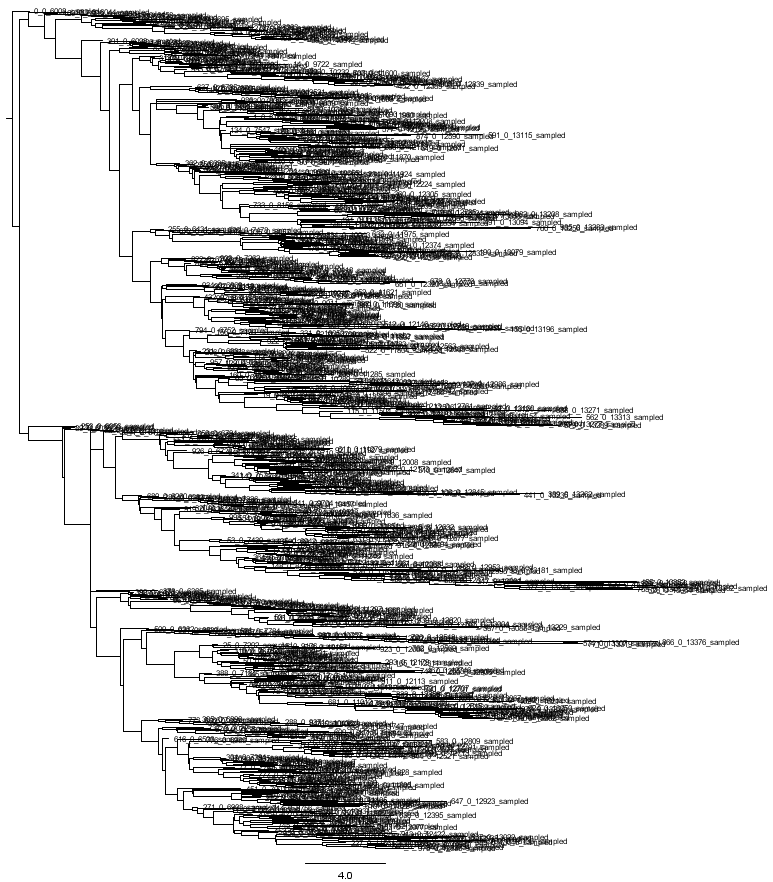


Figure 5: Phylogenetic Tree from simple SIR model (Binary Pruned tree)

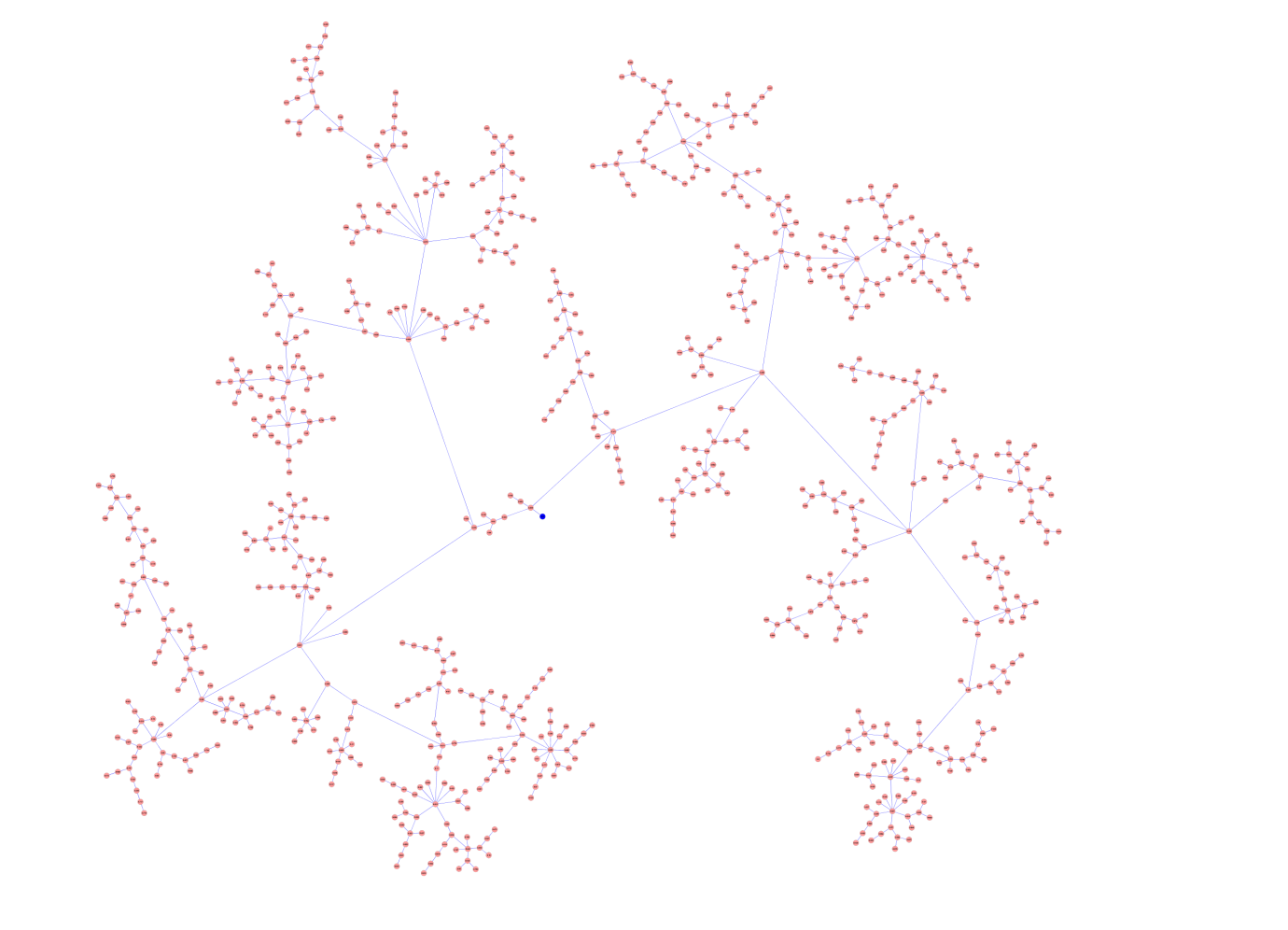


Figure 6: Transmission network from simple SIR model – initial infection is coloured blue.

## Two Demes SIR

Two deme of 500 individuals each, infection model is Susceptible -> Infected -> Recovered. Equal probability of a host infecting in own deme compared to the other.

Figure 7: Number of Infected over time for two Demes SIR model. Blue = deme 0, Red =Deme 1.

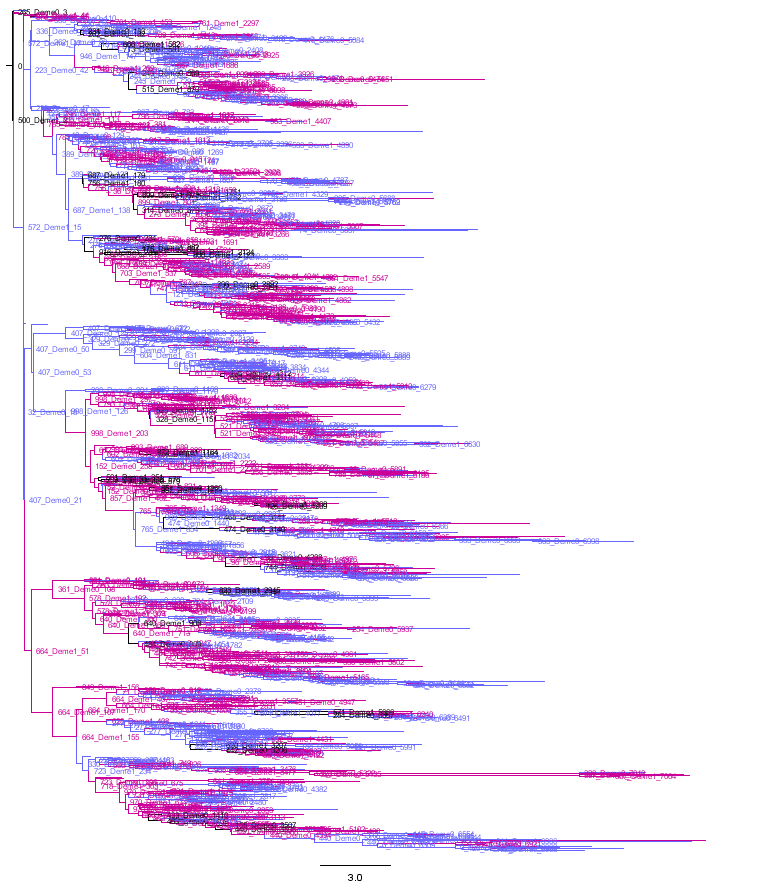


Figure 8: Phylogenetic Tree from two Demes SIR model (Binary Pruned tree). Tips and branches coloured by Deme (blue = Deme 0, pink = Deme 1); internal nodes shown.

## Network

Demes connected in a network, with 1 host per deme and SIR infection mode.

Network From Statnet – Faux Magnolia High Largest Connected Component

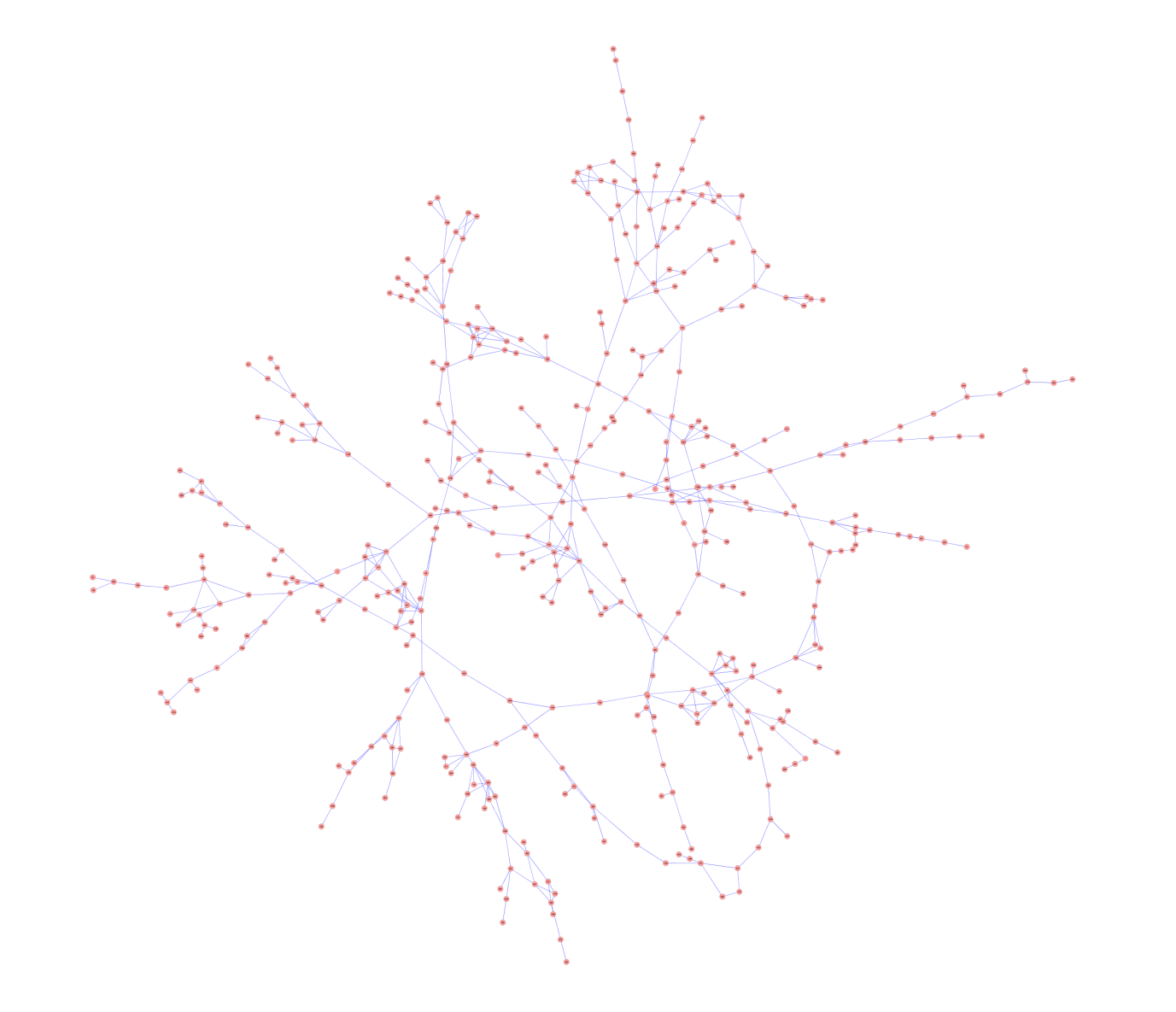


Figure 9 : Connection between Demes; network taken from Faux Magnolia High School Data in Statnet (largest connected component).

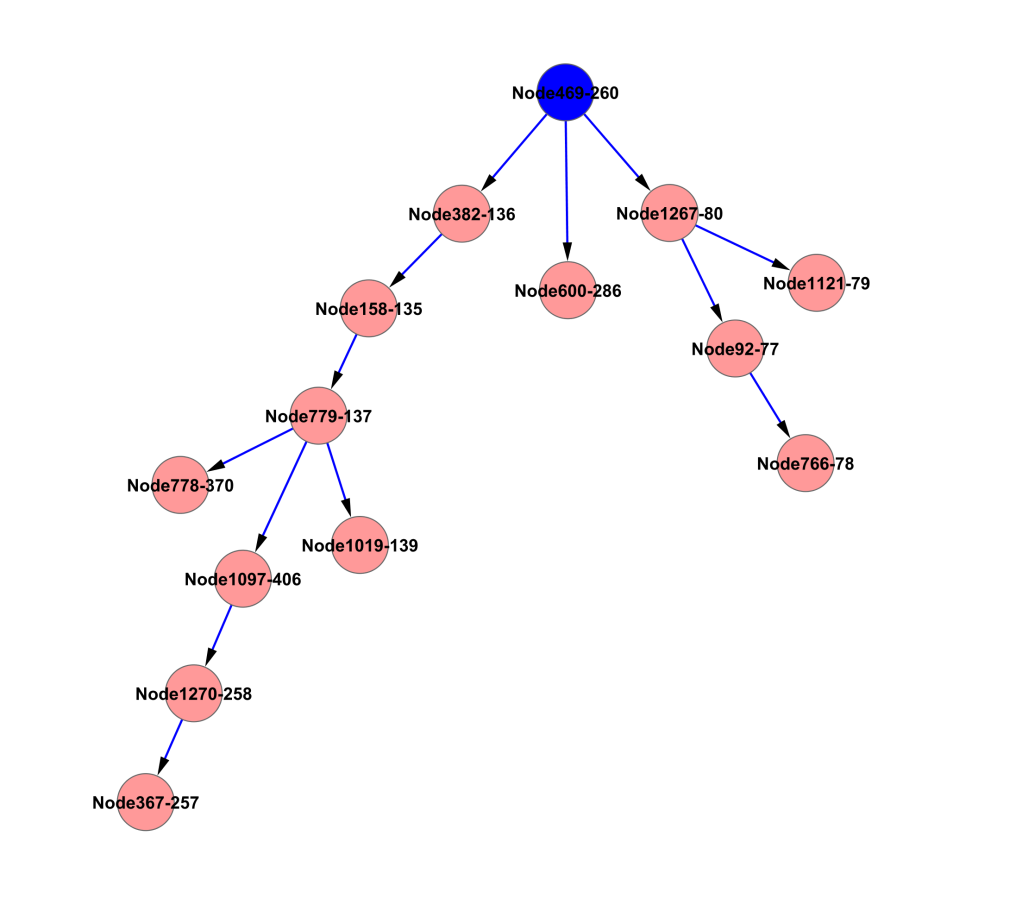


Figure 10: Example of transmission tree over the Faux Magnolia High Network. The blue node is the index case.

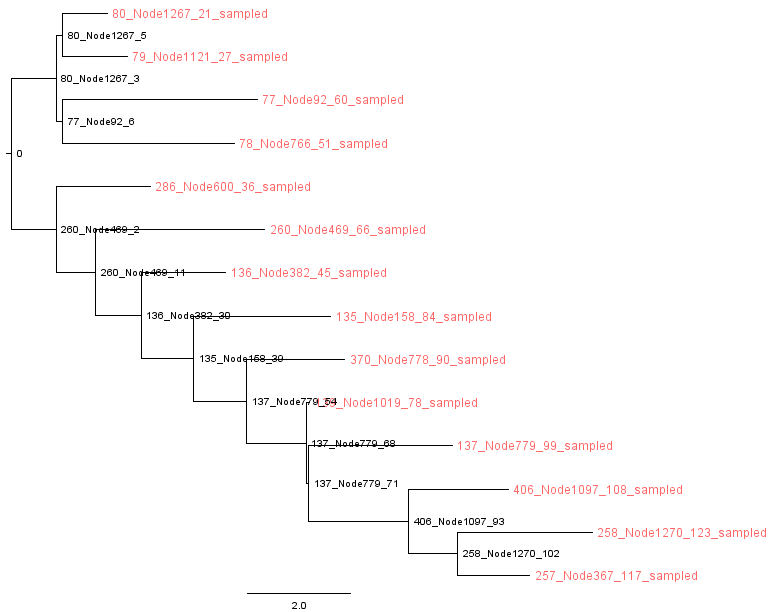


Figure 11: Phylogenetic tree of an infection over the Faux Magnolia High Network.

# Example XMLs

## simpleSI\_params.xml

<?xml version="1.0" standalone="yes"?>

<DSPS>

<General>

<parameter id="Seed" value="12345"/>

<parameter id="Path" value="test//"/>

<parameter id="Rootname" value="simpleSI"/>

<parameter id="Nreps" value="2"/>

<parameter id="Tauleap" value="0"/>

</General>

<Sampler>

<parameter id="SamplerType" value="individualBasedModel.JustBeforeRecoverySampler"/>

<parameter id="justBefore" value="1.0E-16"/>

</Sampler>

<Demes>

<Deme>

<parameter id="DemeUID" value="0"/>

<parameter id="DemeName" value="0"/>

</Deme>

</Demes>

<PopulationStructure>

<parameter id="NetworkType" value="FULL"/>

<parameter id="NumberOfHostsPerDeme" value="1000"/>

<parameter id="ModelType" value="SI"/>

<parameter id="InfectionParameters" value="1"/>

</PopulationStructure>

</DSPS>

## simpleSIR\_params.xml

<?xml version="1.0" standalone="yes"?>

<DSPS>

<General>

<parameter id="Seed" value="12345"/>

<parameter id="Path" value="test//"/>

<parameter id="Rootname" value="simpleSIR"/>

<parameter id="Nreps" value="1"/>

<parameter id="Tauleap" value="0"/>

</General>

<Sampler>

<parameter id="SamplerType" value="individualBasedModel.JustBeforeRecoverySampler"/>

<parameter id="justBefore" value="1.0E-16"/>

</Sampler>

<Demes>

<Deme>

<parameter id="DemeUID" value="0"/>

<parameter id="DemeName" value="0"/>

</Deme>

</Demes>

<PopulationStructure>

<parameter id="NetworkType" value="FULL"/>

<parameter id="NumberOfHostsPerDeme" value="1000"/>

<parameter id="ModelType" value="SIR"/>

<parameter id="InfectionParameters" value="1,0.5"/>

</PopulationStructure>

</DSPS>

## example\_structure\_params.xml

<?xml version="1.0" standalone="yes"?>

<DSPS>

<General>

<parameter id="Seed" value="12345"/>

<parameter id="Path" value="test//"/>

<parameter id="Rootname" value="twoDemes500"/>

<parameter id="Nreps" value="1"/>

<parameter id="Tauleap" value="0"/>

</General>

<Sampler>

<parameter id="SamplerType" value="individualBasedModel.JustBeforeRecoverySampler"/>

<parameter id="justBefore" value="1.0E-16"/>

</Sampler>

<Demes>

<Deme>

<parameter id="DemeUID" value="0"/>

<parameter id="DemeName" value="Deme0"/>

</Deme>

<Deme>

<parameter id="DemeUID" value="1"/>

<parameter id="DemeName" value="Deme1"/>

</Deme>

</Demes>

<PopulationStructure>

<parameter id="NetworkType" value="FULL"/>

<parameter id="NumberOfHostsPerDeme" value="500"/>

<parameter id="ModelType" value="SIR"/>

<parameter id="InfectionParameters" value="1,0.5"/>

<parameter id="DemeType" value="INFECTION\_OVER\_NETWORK"/>

<parameter id="ProbabilityInfectionAnyOtherDeme" value="0.5"/>

</PopulationStructure>

</DSPS>

## example\_network\_params.xml

<DSPS>

<General>

<parameter id="Path" value="test//"/>

<parameter id="Rootname" value="example\_network"/>

<parameter id="Nreps" value="1"/>

<parameter id="Tauleap" value="0.0"/>

</General>

<Sampler>

<parameter id="SamplerType" value="individualBasedModel.JustBeforeRecoverySampler"/>

<parameter id="justBefore" value="1.0E-16"/>

</Sampler>

<Demes>

<Deme>

<parameter id="DemeUID" value="0"/>

<parameter id="DemeName" value="Node2"/>

<parameter id="Neighbours" value="Node751,Node912,Node963"/>

<parameter id="MigrationParameters" value="0.3333333333333333,0.3333333333333333,0.3333333333333333"/>

</Deme>

<Deme>

<parameter id="DemeUID" value="1"/>

<parameter id="DemeName" value="Node751"/>

<parameter id="Neighbours" value="Node2,Node209,Node481,Node492,Node550"/>

<parameter id="MigrationParameters" value="0.2,0.2,0.2,0.2,0.2"/>

</Deme>

<Deme>

<parameter id="DemeUID" value="2"/>

<parameter id="DemeName" value="Node912"/>

<parameter id="Neighbours" value="Node2,Node158,Node308"/>

<parameter id="MigrationParameters" value="0.3333333333333333,0.3333333333333333,0.3333333333333333"/>

</Deme>

(more demes)

</Demes>

<PopulationStructure>

<parameter id="NumberOfHostsPerDeme" value="1"/>

<parameter id="ModelType" value="SIR"/>

<parameter id="InfectionParameters" value="1,0.5"/>

<parameter id="DemeType" value="INFECTION\_OVER\_NETWORK"/>

</PopulationStructure>

</DSPS>

## R Script to make edge list for Faux Magnolia High Largest Component

# example networks with statnet

# S. J. Lycett

# 10 Oct 2013

library(statnet)

#######################################

# load faux.magnolia.high friendship network

# http://127.0.0.1:10907/library/ergm/html/faux.magnolia.high.html

# load data

data(faux.magnolia.high)

fmh <- faux.magnolia.high

# plot data

plot(fmh, displayisolates=FALSE, vertex.cex=0.7)

# cluster sizes

clusts <- component.dist(fmh)

cs <- table(clusts$csize)

# degree distribution

dd <- table(degree(fmh, cmode="indegree"))

# formats

is.network(fmh)

# to edge list

edgeList <- as.matrix.network.edgelist(fmh)

# write to file

fname <- "C://test//faux\_magnolia\_high\_edgeList.csv"

write.table( edgeList, file=fname, sep=",", col.names=FALSE, row.names=FALSE)

# get largest connected component

clustNum <- which.max(clusts$csize)

ids <- which(clusts$membership==clustNum)

einds1 <- match(edgeList[,1], ids)

jj1 <- which(is.finite(einds1))

einds2 <- match(edgeList[,2], ids)

jj2 <- which(is.finite(einds2))

einds <- sort(unique(c(jj1,jj2)))

edgeList2<- edgeList[einds,]

# write largest component to file

fname <- "C://test//faux\_magnolia\_high\_edgeList\_largest\_connected\_component.csv"

write.table( edgeList2, file=fname, sep=",", col.names=FALSE, row.names=FALSE)