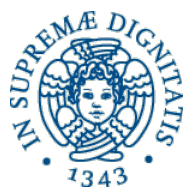
Endosymbiotic Particle System in NetLogo

Antonio Di Mauro

Department of Computer Science
University of Pisa

Project discussion of Computational Models for Complex Systems Course



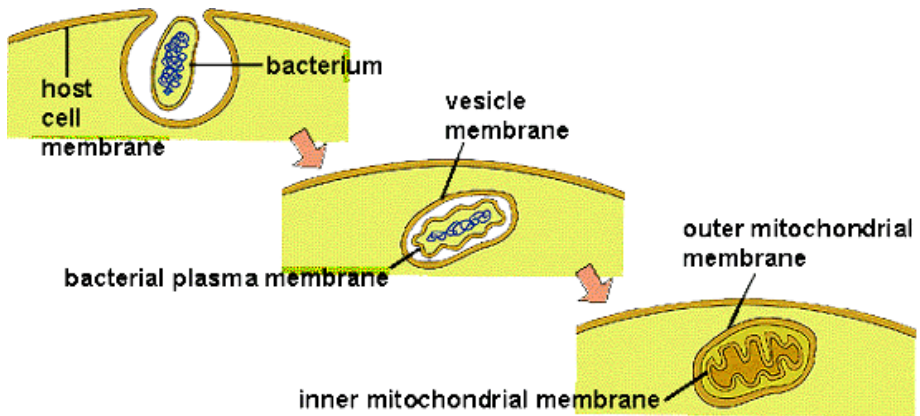
Endosymbiotic Theory (Symbiogenesis)

Endocytosis = (cyto = cell) a process of "cell eating", cells are engulfed and digested as food

Endosymbiosis = cells are engulfed but not digested, cells live together in a mutually benefitting relationship, or symbiosis

Endosymbiotic Theory (cont.)

1. Evolutionary theory of the origin of eukaryotic cells from prokaryotic organisms
2. Proposed by Konstantin Mereschkowski in 1905 and advanced with microbiological evidence by Lynn Margulis in 1967
3. Original hypothesis proposed that aerobic bacteria (that require oxygen) were ingested by anaerobic bacteria, and each have a survival advantage



The Particle System

The approach to the endosymbiotic system is a particle system with attractions and repulsions among entities (without phagocytosis).
Below the steps in a single unit of time made by a particle:

1. Find affine: the particle find the neighbour with the maximum quantity of molecule of interest
2. Find not affine: the particle find the neighbour with the minimum quantity of molecule of interest
3. Move away: the particle rejects the not affine and moves away from it
4. Move towards: the particle is attracted by the affine and moves towards it
5. Consume: the particle consumes the quantity of the molecule of interest of the affine
6. Produce: the particle increases the quantities of its molecules

NetLogo Functions

Find affine and not affine

```
to find-affine
  let idx species
  set affine max-one-of neighbours [ item idx species-molecules ]
  set not-affine min-one-of neighbours [ item idx species-molecules ]
  if affine = not-affine
    [ set not-affine no-turtles ]
end
```

Move away

```
to move-away-not-affine
  if not-affine != no-turtles [
    face not-affine
    rt 180
    fd 1 ]
end
```

Move towards

```
to move-to-affine
  if affine != no-turtles [ move-to affine ]
end
```

NetLogo Functions (cont.)

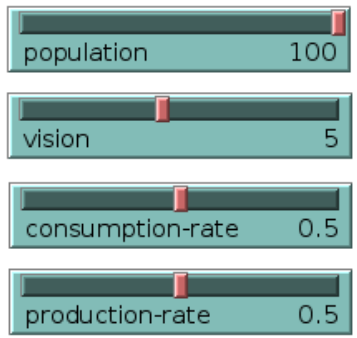
Consume

```
to consume
  let idx species
  ask affine [
    if item idx species-molecules > 0
      [ set species-molecules replace-item idx species-molecules
        (item idx species-molecules - (consumption-rate * item idx species-molecules)) ]
  ]
end
```

Produce

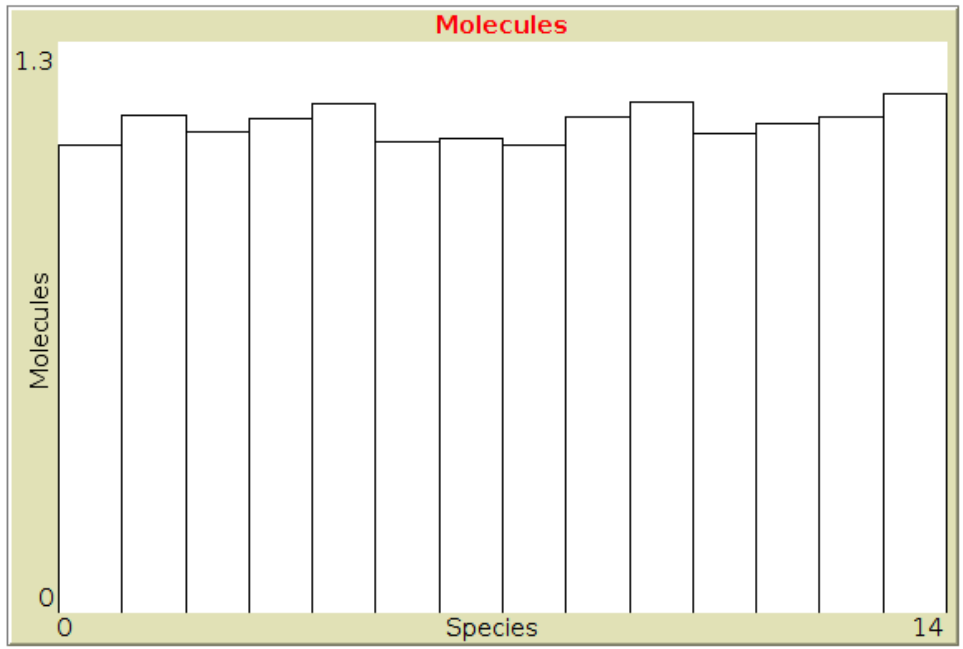
```
to produce
  let indexer ( range 0 length species-molecules )
  foreach indexer [ idx ->
    if (item idx species-molecules) < 1
    [ set species-molecules replace-item idx species-molecules
      (item idx species-molecules + (production-rate * item idx species-molecules)) ]
  ]
end
```

NetLogo Parameters



1. population: the number of particles
2. vision: number of patches visible by a particle, used to find neighbours
3. consumption-rate: the rate (between 0 and 1) of comsumption of molecules
4. production-rate: the rate (between 0 and 1) of production of molecules

NetLogo Plot



- Monitoring the quantities of molecules of each species in the system

Empirical facts

1. Increasing/decreasing the vision leads to decreasing/increasing of clusters
2. increasing or decreasing the reproduction and consumption rates causes a temporary instability of the quantities of molecules, but its stabilizes quickly

Lets see the code...