

Using Java and Swarm to Build a Stochastic Simulation of Natural Organic Matter (NOM) and Microbe Interactions in the Soil

Greg Madey¹

Yingping Huang¹

Steve Cabaniss²

Patricia Maurice¹

(1)University of Notre Dame

(2) Kent State University

Background

- NSF - ITR - DEB
- Interdisciplinary project
 - Chemist - Steve Cabaniss, responsible for the modeling ideas
 - Biologists
 - Ecologists
 - Geologists
 - Computer Scientist
- Stochastic Simulation of Environmental Transformations of Natural Organic Matter
 - In soil
 - In solution

Natural Organic Matter

- Ubiquitous in terrestrial, aquatic and marine ecosystems
- Important role in compositional evolution and fertility of soil
- Impacts mobility and transport of pollutants, e.g., trace metals, radionuclides and hydrophobic organic compounds
- Impacts availability of nutrients for microorganisms and plant communities
- Impacts growth and dissolution of minerals

Natural Organic Matter (cont)



Hardwood
Swamp
(HS)
Upstream
GW Recharge
Area

Natural Organic Matter (cont)



Open
Channel
(OC)
Variable
Recharge

Natural Organic Matter (cont)



Cedar
Swamp
(CS)
Downstream
GW Discharge
Area

Background

- Compositional evolution of NOM is an interesting problem
- Important aspect of predictive environmental modeling
- Prior modeling work is often too simplistic to represent the heterogeneous structure of NOM and its complex behaviors in ecosystems (e.g., carbon cycling models), also ...
- Prior modeling work is often too compute-intensive to be useful for large-scale environmental simulations (e.g., molecular models employing connectivity maps or electron densities)

Project Goals

- Stochastic model of NOM evolution — middle computational approach
 - Algorithms
 - Parameters
- Represent individual molecules as discrete objects with
 - Specified elemental and functional group composition
 - Size/weight
 - Reactivity
- Model the evolution of NOM from biological precursor compounds
 - Lignin
 - Polysaccharides
 - Proteins
- Distribute simulation for testing and adoption (web-based?)
- Generate experimentally testable predictions about NOM evolution and properties

Modeling

- Molecules and microbes are objects
- Molecules and microbes have attributes (parameters)
 - Heterogeneous, distributions
 - At least 1,000 objects, preferably 10,000 or more
- Molecules have behaviors (reactions)
 - Behaviors are stochastically determined
 - Dependent on the:
 - Attributes (intrinsic parameters)
 - Reaction rates
 - Environment (extrinsic parameters)

Modeling (cont)

- Objects of interest
 - Macromolecular precursors
 - Polysaccharides
 - Proteins
 - Polynucleotide, tannin, lignin, polyterpene, cutin
 - Smaller molecules
 - Phospholipids
 - Sugars
 - Amino acids
 - Flavonoids
 - Quinones
 - Microbes

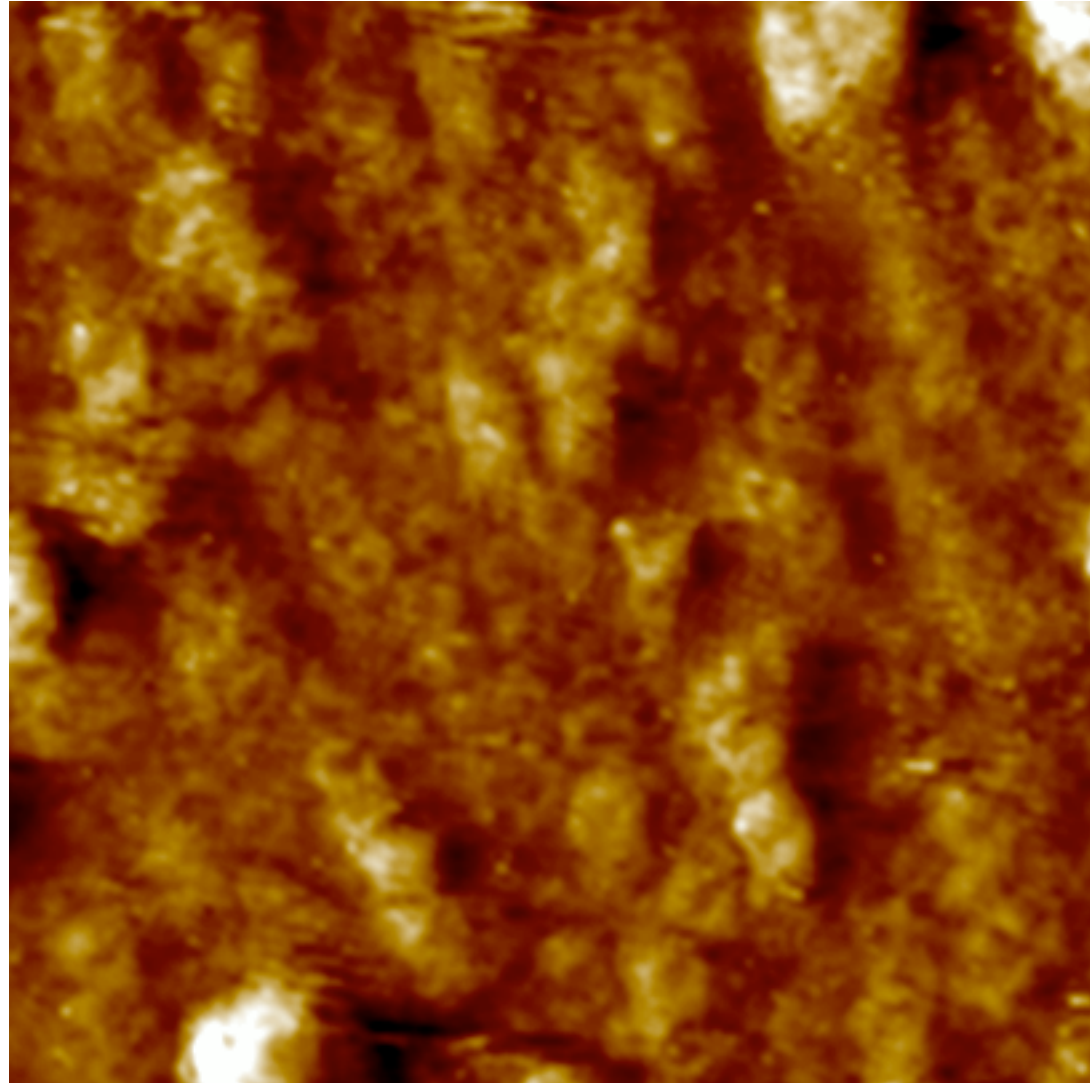
Modeling (cont)

- Attributes
 - More specific than percent carbon” but less detailed than a molecular connectivity map
 - Elemental composition
 - Number of C, H, O, N, S and P atoms in molecule
 - Functional group counts
 - Double-bonds
 - Ring structures
 - Phenyl groups
 - Alcohols
 - Phenols, ethers, esters, ketones, aldehydes, acids, aryl acids, amines, amides, thioethers, thiols, phosphoesters, phosphates
 - Time molecule entered the system
 - Origin type of molecule

Modeling (cont)

- Behaviors (reactions and processes)
 - Physical reactions
 - Adsorption to mineral surfaces
 - Initial adsorption
 - Surface migration to high-energy sites
 - Hemi-micelle formation at high coverage (cooperative, hydrophobicity dependent)
 - Aggregation/micelle formation (e.g., metal cation-induced aggregation) - flocs
 - Transport downstream (surface water)
 - Transport through porous media
 - Volatilization

AFM Image of NOM Adsorption



NOM
Rings

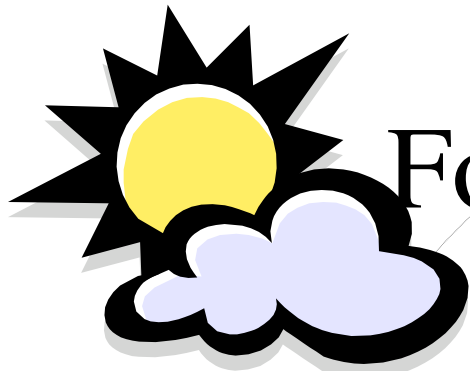
Maurice, 1999

Modeling (cont)

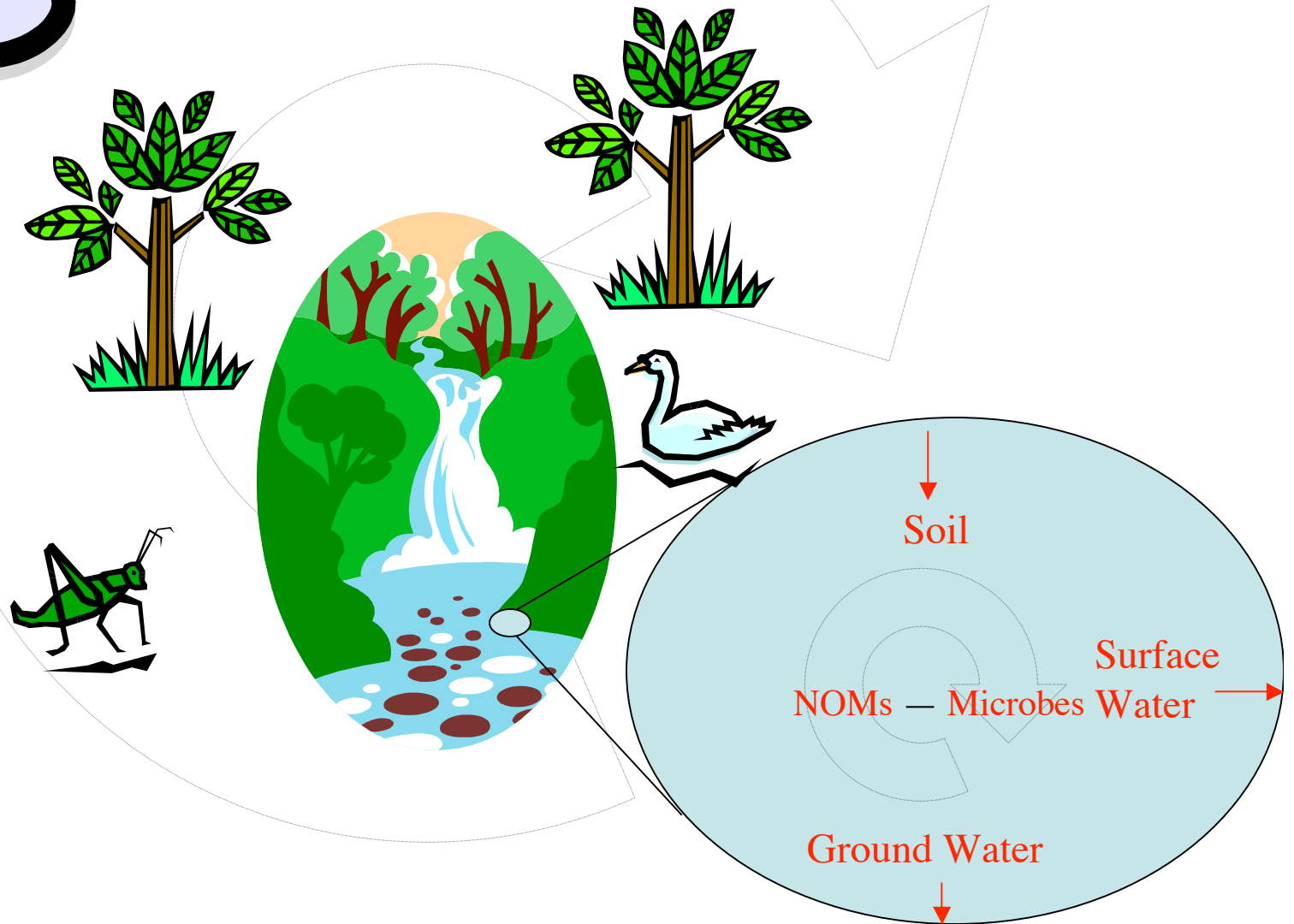
- Behaviors (reactions and processes)
 - Chemical reactions
 - Abiotic bulk reactions
 - Hydrolysis
 - Hydration
 - Ester condensation
 - Thermal decarboxylation
 - Abiotic surface reactions
 - Direct photochemical reactions
 - Indirect photochemical reactions
 - Extracellular enzyme reactions on large molecules
 - Bacteria
 - Fungi
 - Algae
 - Microbial uptake of small molecules

Modeling (cont)

- Environmental parameters
 - Temperature
 - pH
 - Light intensity
 - Metal concentrations (e.g., Al(III) and Fe)
 - Bacterial activity
 - Water flow rate/pressure gradient
 - Surface area



Focus of the Modeling



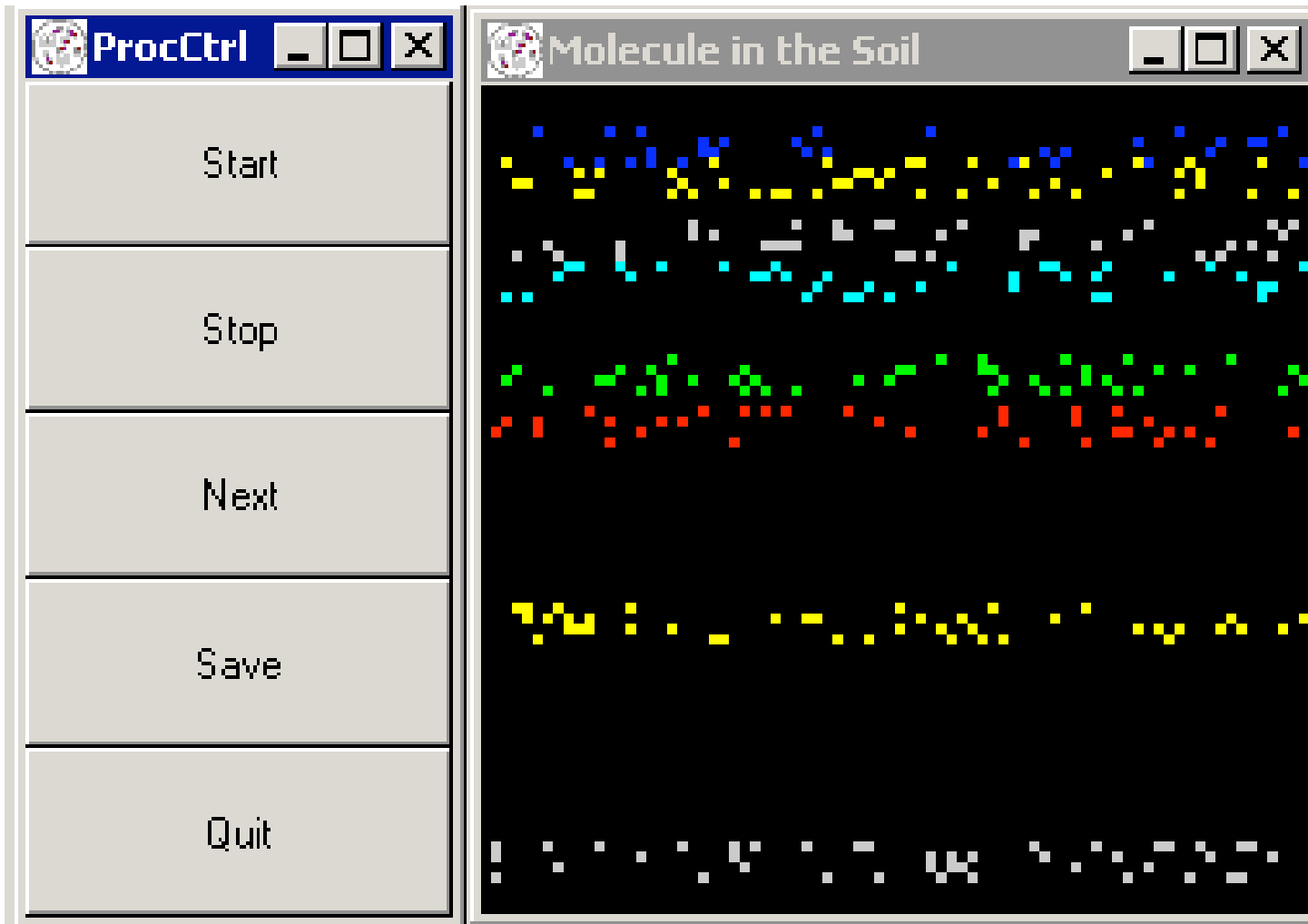
Swarm Modeling

- Evaluation of Java/Swarm
 - Why Java?
 - Java/Swarm resources? (emphasis on Java)
 - jSimpleBug - C. Staelin
 - SDG 2000 Complexity School Tutorial - M. Daniels
 - Swarm User Guide (Part 1) - Johnson & Lancaster
 - Swarm Java API
 - <http://www.santafe.edu/projects/swarm/swarmdocs/refbook-java/index.html>
 - Swarm FAQ - P. Johnson
 - Code examples
 - jSimplebug
 - SDG
 - Heatbugs

Swarm Modeling (cont)

- Metaphors
 - Molecules and microbes as bugs
 - Forager-food relationship
 - Soil as environment (food space)
 - Two dimensional grid models
 - Heatbug
 - jSimpleBug
- Transport through soil
 - One, two, three dimensional spaces
 - Lower molecular weight molecules are food for microbes
 - Larger molecular weight molecules adsorb to soil minerals
- Class molecule
- Class soil

Swarm Modeling (cont)



Swarm Modeling (cont)

- Questions
 - Is Java/Swarm the right toolkit?
 - Scalability
 - Performance
 - Web deployment
- BTW ... one of my hobbies is ...

Make File

```
JAVA_SRC=Soil.java Molecule.java ModelSwarm.java  
ObserverSwarm.java
```

```
all: $(JAVA_SRC)
```

```
    $(SWARMHOME)/bin/javacswarm $(JAVA_SRC)
```

```
clean:
```

```
    rm -f *.class
```

```

public class Molecule extends SwarmImpl{
    public double getStuckProbability; //probability that the molecule get stuck
    public int x, y; //origin
    public Grid2d world; //the world the molecule lives in
    public byte color; //color of the molecule
    public byte cnum; //number of C's in the molecule
    public int num; //the number of the same kind of molecules

    Schedule schedule;

    public Molecule(Zone aZone, Soil soil, int x, int y, byte cnum, double prob,
        int num){
        super(aZone);
        this.world=soil.getWorld();
        this.x=x;
        this.y=y;
        this.cnum=cnum;
        getStuckProbability=prob;
        this.num=num;
        world.putObject$atX$Y(this, x, y);

        schedule=new ScheduleImpl(aZone, 10*cnum);

        try{
            schedule.at$createActionTo$message(0, this,
                new Selector(getClass(), "moveMolecule", false));
        }catch(Exception e){
            e.printStackTrace(System.err);
            System.exit(1);
        }
    }
}

```

```
//Soil.java
//The soil that molecules live in

import swarm.objectbase.SwarmImpl;
import swarm.defobj.Zone;
import swarm.space.Grid2d;
import swarm.space.Grid2dImpl;

public abstract class Soil extends SwarmImpl{
    Grid2d world;
    int xSize, ySize;

    public Soil(Zone aZone, int xSize, int ySize){
        super(aZone);
        this.xSize=xSize;
        this.ySize=ySize;
        world=new Grid2dImpl(getZone(), xSize, ySize);
    }

    public Grid2d getWorld(){
        return world;
    }
}
```

```
public class ObserverSwarm extends GUISwarmImpl{  
    public final static byte c3color=0;  
    public final static byte c4color=1;  
    public final static byte c5color=2;  
    public final static byte c6color=3;  
    public final static byte c7color=4;  
    public final static byte c8color=5;  
    public final static byte c9color=6;  
    public final static byte c10color=7;
```

```
    Object2dDisplay display;  
    ZoomRaster raster;  
    ModelSwarm model;  
    Colormap colormap;  
    Schedule displaySchedule;
```

```
    public ObserverSwarm(Zone aZone){  
        super(aZone);  
        model=new ModelSwarm(aZone, 80, 80);  
    }
```

```
//ModelSwarm.java
```

```
import swarm.activity.Activity;  
import swarm.objectbase.Swarm;  
import swarm.Globals;  
import swarm.defobj.Zone;
```

```
import java.util.List;  
import java.util.LinkedList;  
import java.util.Iterator;
```

```
public class ModelSwarm extends Soil{  
    final static int numMolecules=320;  
    final static int kinds=8;  
    Molecule mol;  
    List moleculeList;  
  
    public ModelSwarm(Zone aZone, int xsize, int ysize){  
        super(aZone, xsize, ysize);  
    }  
  
    public Object buildObjects(){  
        super.buildObjects();  
  
        moleculeList=new LinkedList();
```