Agent-based Scientific Applications and Collaboration Using Java

Xiaorong Xiang

Advisor: Dr. Kevin Bowyer

Department of Computer Science and Engineering
University of Notre Dame

Sponsored by NSF/ITR-DEB

Objectives

- New approach for NOM modeling
  - Agent-based modeling
- E-Science on the Web
- Intelligent interface components
- Build the NOM Collaboratory
- Performance analysis for scientific applications

Outline

- Introduction
- Modeling
- Core simulation engine
- Intelligent Web-based interface
- The NOM collaboratory
- Java performance analysis
- Conclusion
- Future work

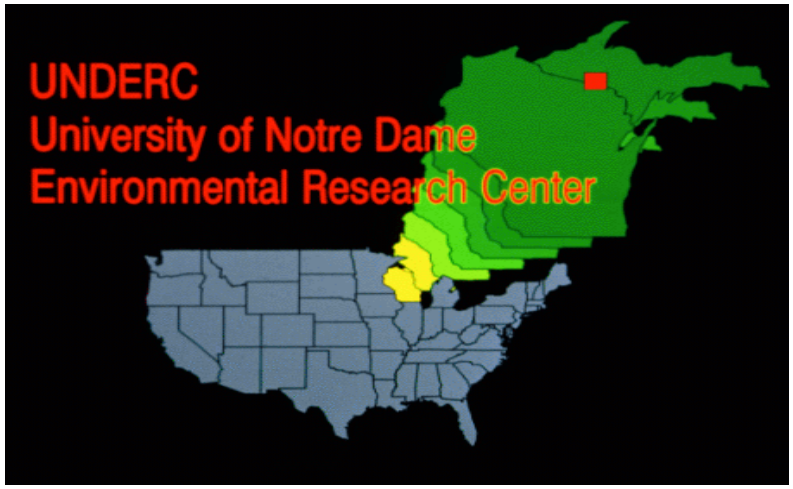
Introduction

- What is Natural Organic Matter (NOM)?
- Role of NOM in various science disciplines
  - Mobility and transport of pollutants
  - Availability of nutrients for microorganisms and plant communities
  - Affects quality of drinking water
- Need to understand the evolution and heterogeneous structure of NOM

Forest Service Bog [DOC] 7 MW 2200

Twomile Creek [DOC] 17 MW 1500

Nelson Creek [DOC] 79 MW 900

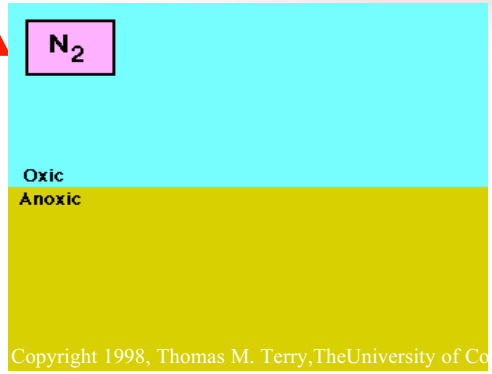


Previous models

- Two examples:
  - Daisy (S. Hansen, H. E. Jensen, and N. E. Nielsen 1990-present) : a soil plant atmosphere system model
  - StochSim (C. J. Morton-Firth 1998-present): Stochastic simulation of cell signaling pathways

large

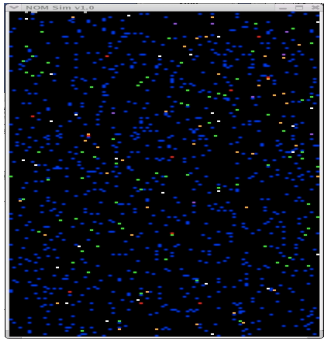
(Large ecosystem)
(Years)



Elemental Cycling

Daisy

NOM1.0

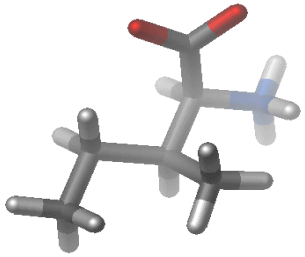


Scale

(size, temporal)

StochSim

Connectivity Maps



(One molecule)
(nanoseconds)

small

low

(Atoms number
Percentage)

Detail

(structure)

high

(Forces between atoms
Electron density)

Our model

- Agent-based modeling (Individual-based modeling)
  - Agent-based model
    - Reynolds (1987): Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*
  - Each molecule as an individual object with spatial properties
  - Bottom-up approach
  - Stochastic model
  - Trace changes of the system → Database and data mining technologies

Our model (cont.)

- Web-based scientific application
  - Serve as an example for E-Science
    - G. Fox (2002): E-science meets computational science and information technology. *Computing & Engineering*
    - R. M. Jakobovits, J. F. Brinkley, C. Rosse, and E.Weinberger (1998): Enabling clinicians, researchers, and eductors to build custom Web-based biomedical information system
  - Support the collaborations, data and information sharing between scientists
  - No installation, expensive computation resources needed by scientists

Outline

- Introduction
- Modeling
- Core simulation engine
- Intelligent Web-based interface
- The NOM collaboratory
- Java performance analysis
- Conclusion
- Future work

Modeling

- A complex system
  - Consists of a large number of objects
    - Molecules, Microbes
      - **Heterogeneous properties**
      - **Individual behaviors**
      - **Interaction between objects**
      - **Objects behavior and interaction are stochastically determined by:**
        - **Attributes**
        - **Reactions rates**
        - **Environment condition**
  - No central control
  - Emergent properties

Modeling (cont)

- Agent Attributes
  - Elemental composition (C, H, O, N, S, P)
  - Functional groups (double bonds, ring structure, alcohols …)
  - The origin of objects (spatial position in the system, parents of the objects)
  - Probability table of physical and chemical reactions
  - Molecule weight

Modeling (cont.)

- Agent Behaviors:
  - Transport through soil pores by water (spatial mobility)
  - Adsorb onto or desorbed from mineral surfaces
  - Chemical reactions
    - Total 10 types in current model
    - First order
    - Second order
  - Stochastically determined
- Space:
  - 2-D grid

Outline

- Introduction
- Modeling
- Core simulation engine
- Intelligent Web-based interface
- The NOM collaboratory
- Java performance analysis
- Conclusion
- Future work

Core simulation engine

- Implementation
  - Swarm toolkit
  - Java programming language (JDK 1.4.1_01)
- GUI version
  - View the animation of molecules
  - Easy for debugging and modeling
- Web-based version, the NOM simulation model

Core simulation engine (cont.)

- Read simulation parameter from the database (JDBC)
  - Environmental parameters (pH, temperature, light intensity, and so on)
  - Molecule types and distributions
- User defined time has been separated to a large number of equal size time steps
- Write relevant data into the database every time step (JDBC)
  - Trace the dynamic properties of individuals and the system over time

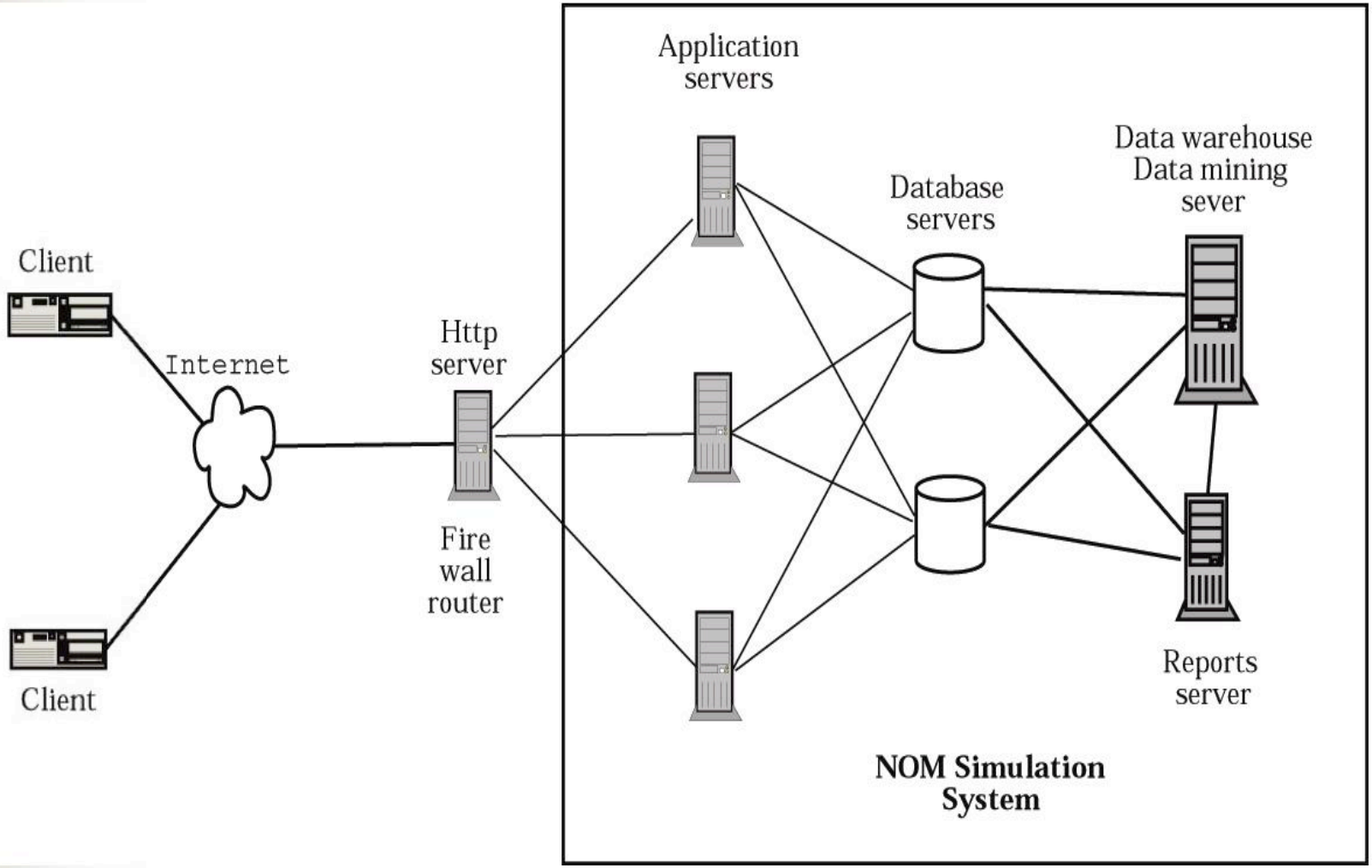
Outline

- Introduction
- Modeling
- Core simulation engine
- Intelligent Web-based interface
- The NOM collaboratory
- Java performance analysis
- Conclusion
- Future work

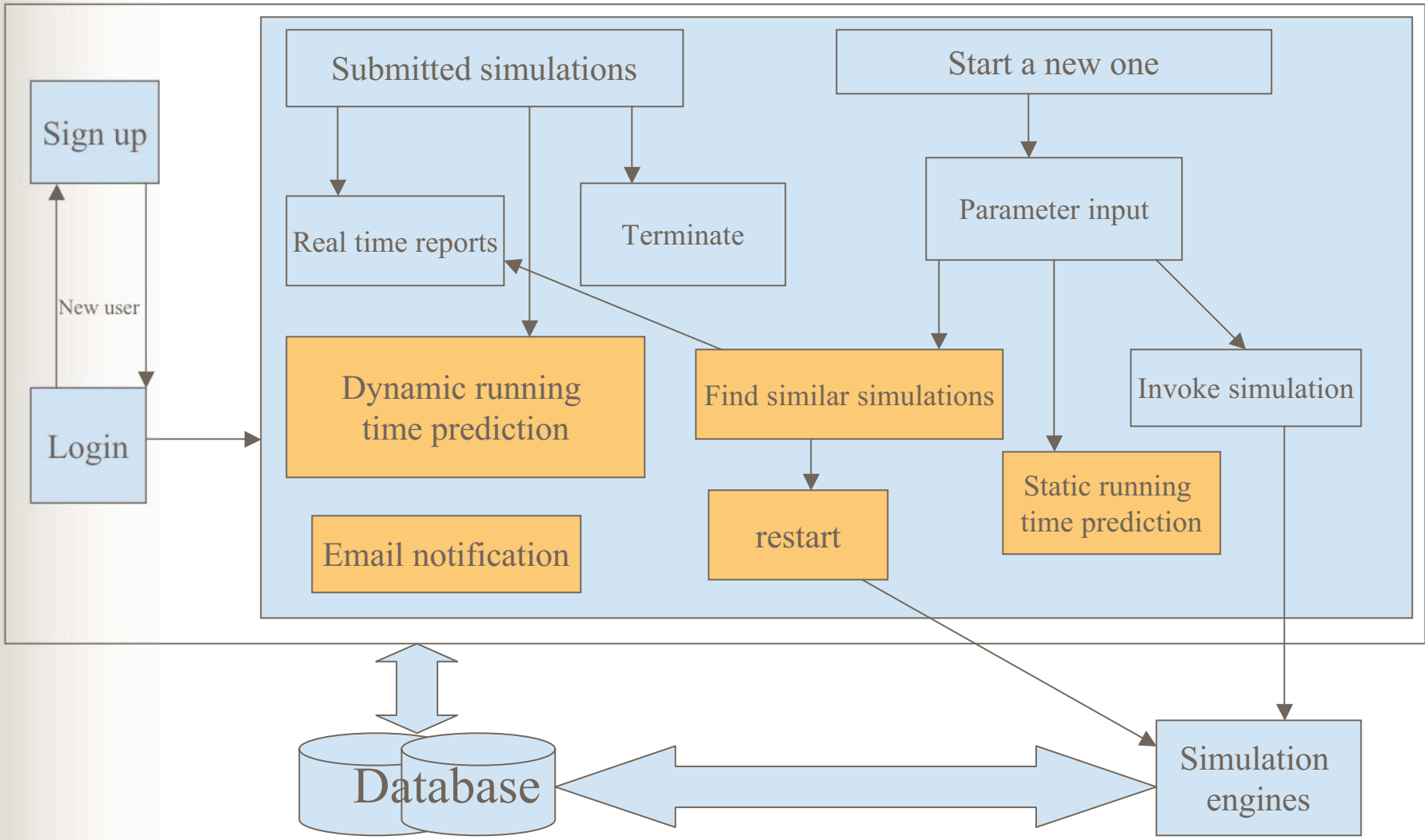
Web-based model

- Distributed, Web-based scientific application model
- Based on Java 2 Enterprise Edition (J2EE)
  - Standard HTML Forms / JSP
  - Java Servlets, Java Beans
  - JDBC - Oracle
  - Oracle Forms and Reports
- Three parts:
  - Intelligent Web-based interface
  - Core simulation engine
  - Data analysis, Data mining

Access NOM simulation from Web



Web-based interface



Intelligent components

- Components:
  - Email notification
  - Running time prediction
    - Static: number of molecules, number of time steps
    - Dynamic: current time step, current wall clock time

Intelligent components (cont.)

- Find similar simulations
  - Environment parameters
  - Molecule types and distributions
  - Retrieve the data sets from database
  - Points on a high dimension space
  - Euclidean distance
  - Ordered list
  - Review the simulation results or restart

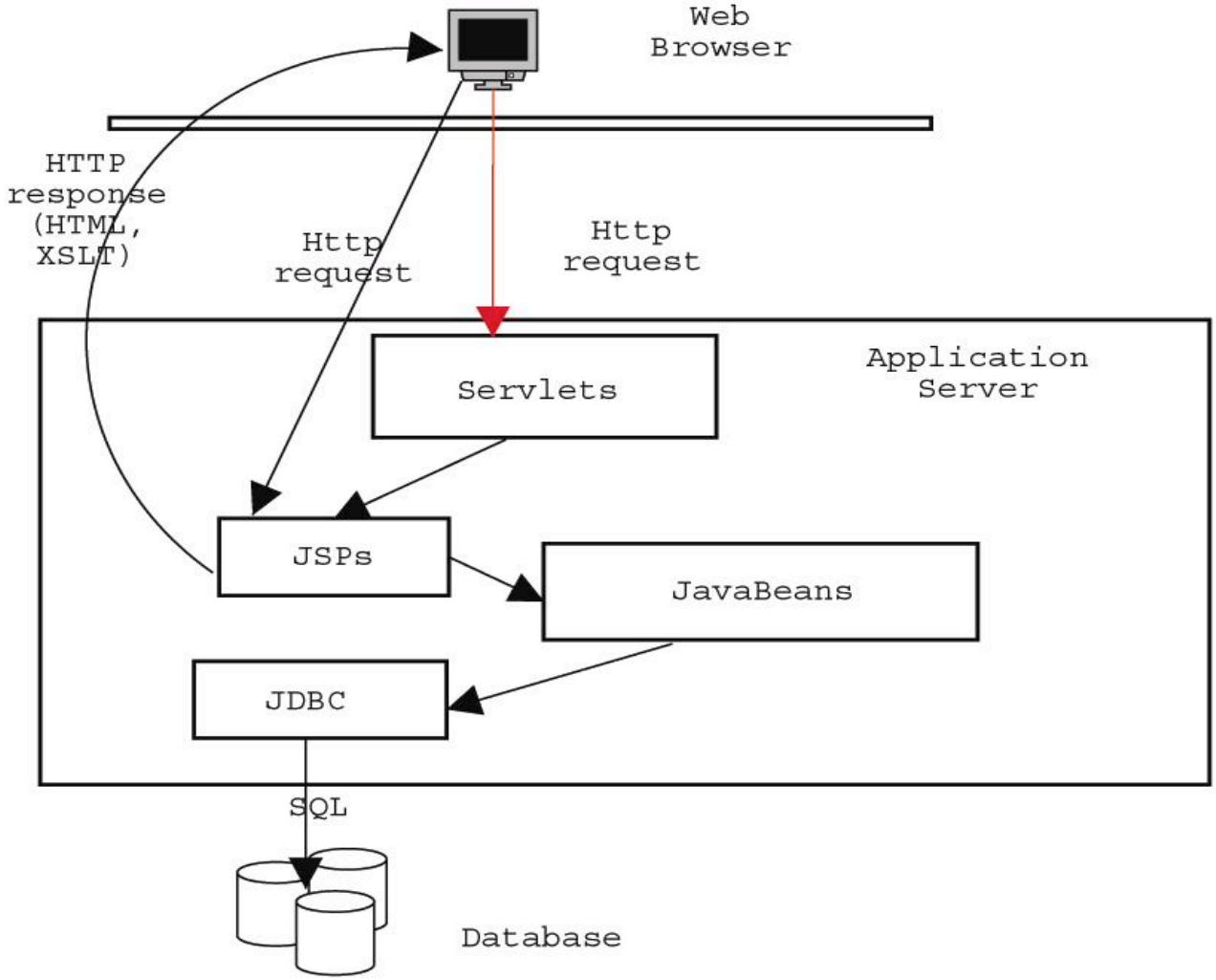
Intelligent components (cont.)

- Automatic restarter
  - Save the state of each objects in the system to database every check point
  - Load the state to the core simulation engine

Intelligent interface design

- Model-View-Controller (MVC) design pattern
  - Model → Application logic
  - View → Presentation logic
  - Controller →Session management
- Separate the design task, centralized control
- Code reuse
- Make the application more easily maintainable
- Well-suited for round-trips of requesting and displaying data

Web interface implementation



Outline

- Introduction
- Modeling
- Core simulation engine
- Intelligent Web-based interface
- The NOM collaboratory
- Java performance analysis
- Conclusion
- Future work

Previous work

- Combination of words "collaboration" and "laboratory" first coined by William Wulf (1996): Richard T. Kouzes, James D. Myers, and William A. Wulf. Collaboratories: Doing science on the internet. *IEEE Computer, 1996*
- Diesel Collaboratory: C. M. Pancerella, L. A. Rahn, and C. L. Yang: The diesel combustion collaboratory: combustion researchers collaborating over the internet. In *Proceedings of the 1999 ACM/IEEE conference on Supercomputing*
- BioCoRE: *http://ks.uiuc.edu/Research/biocore*
- EMSL Collaboratory: *http://www.emsl.pnl.gov:2080/docs/collab*

The NOM Collaboratory

- Interdisciplinary project
  - Chemist
  - Biologist
  - Ecologist
  - Computer Scientist
- Build and integrate software using J2EE
  - NOM modeling & simulation software
  - Standard data format definitions
  - Data querying and manipulation tools
  - Electronic communication tools

NOM Collaboratory - Mozilla

File Edit View Go Bookmarks Tools Window Help

Back Forward Reload Stop http://localhost:8888/interface_COLLABORATION/Collaboratory.html Search Print

NOM Collaboratory

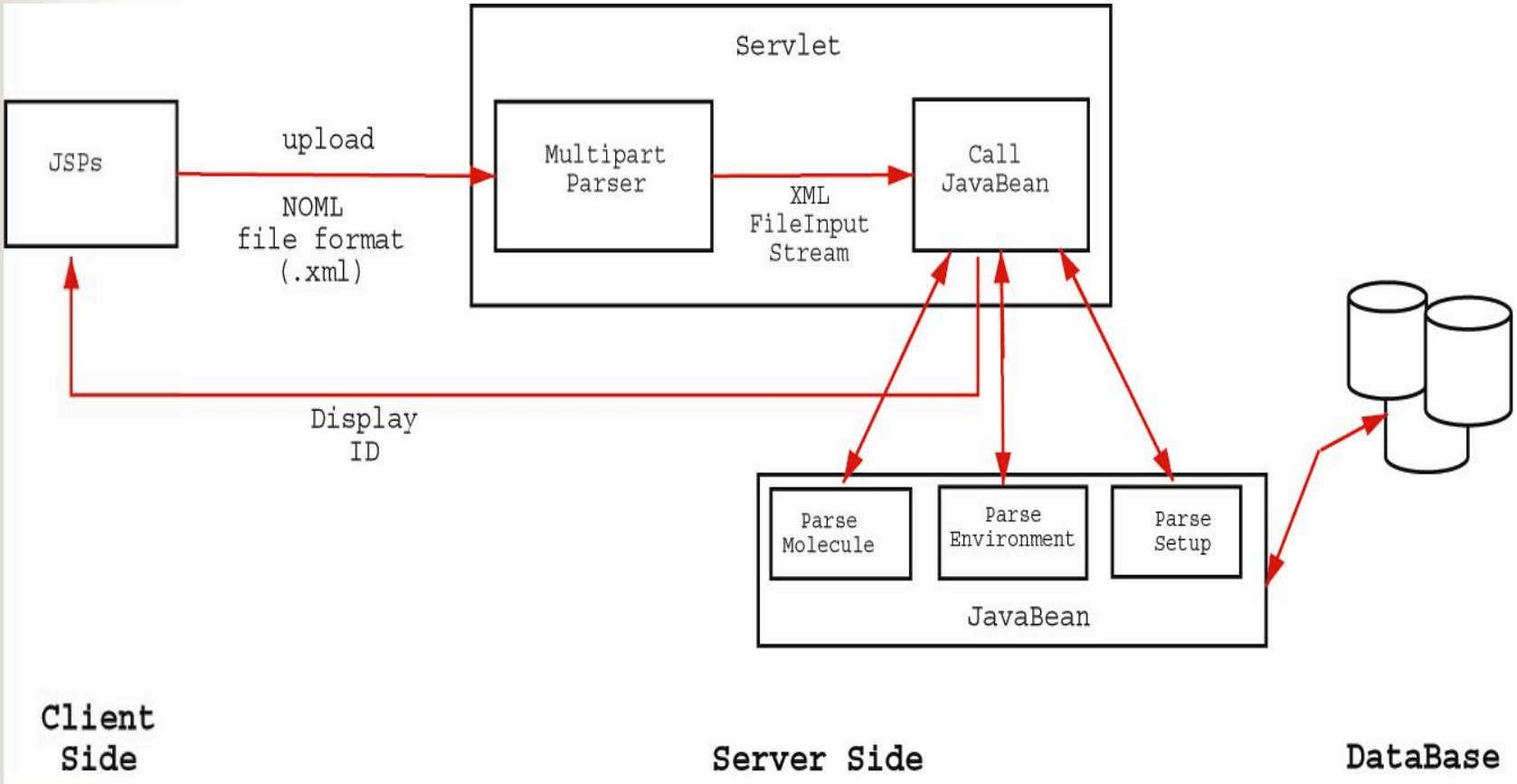
NOM Simulator	Provide an intelligent interface to facilitate using the NOM simulator	Provide an Inteface to define new molecule type	Molecule Editor
Search Engine	Provide an interface to search simulation information	Provide an administration role to validate the newly added molecule for public usage	Molecule Validator
Discussion Board	Provide a threaded discussion board	Provide a XML-based Markup Language definition	NOML
Chat Room	Provide a real time chat box	Provide an Interface to upload publications	File Upload

Document: Done (0.154 secs)

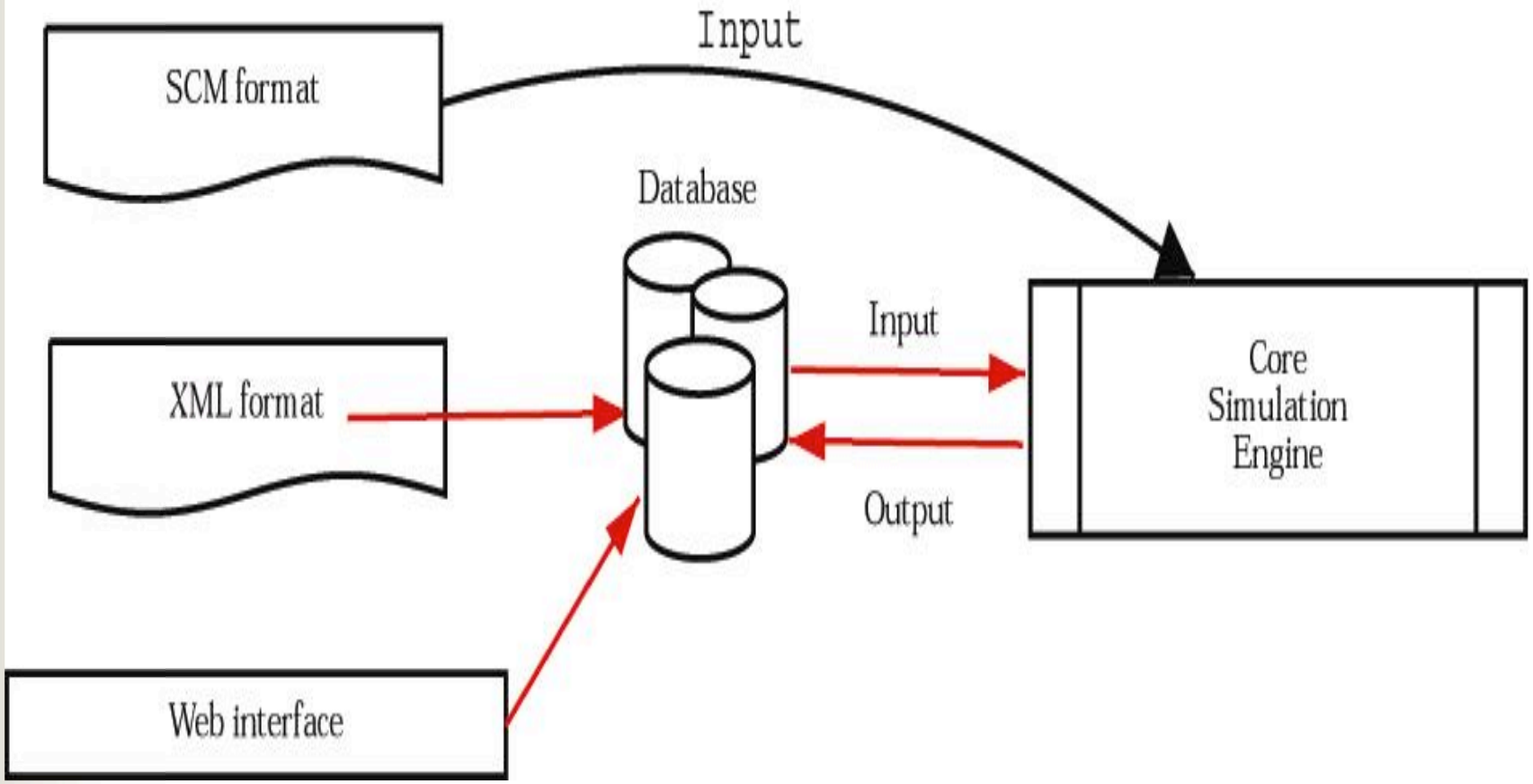
XML-based NOM Markup Language (NOML)

- NOML:
  - Standard data format
    - Environment.dtd, Molecules.dtd, Setup.dtd
- Facilitate communication
  - User ==== User
  - Application ==== Application
- Extension

NOML uploader



Data input options



Other tools

- Molecule editor
  - Define new molecule type
- Molecule validator
  - Authorized persons (Chemists) to validate data
  - Share the molecule type
- Search engine
  - Ad-hoc query
  - View results of the completed simulations
  - Restart some simulations

Outline

- Introduction
- Modeling
- Core simulation engine
- Intelligent Web-based interface
- The NOM collaboratory
- Java performance analysis
- Conclusion
- Future work

Java for Scientific Applications

- Advantages
  - Portability, automatic memory management
  - Java built-in threads, Java RMI
- Disadvantage
  - Performance
- M. Ashworth: The potential of Java for high performance applications. In *the International Conference on the Practical Application of Java, 1999*
- Java Grande benchmark suite
  http://www.epcc.ed.ac.uk/javagrande

Previous work

- Runtime environment optimizations
  - Just-In-Time compilers
  - Bytecode Optimizer
  - Adaptive compilers
  - Native code compilers
- J. M. Bull *et al.*: Benchmarking Java against C and FORTRAN for scientific applications. *In proceedings of the 2001 joint ACM-ISCOPE conference on Java Grande*
- V. Getov *et al.*: Multiparadigm communications in Java for gird computing. *Communication of the ACM, 2001*

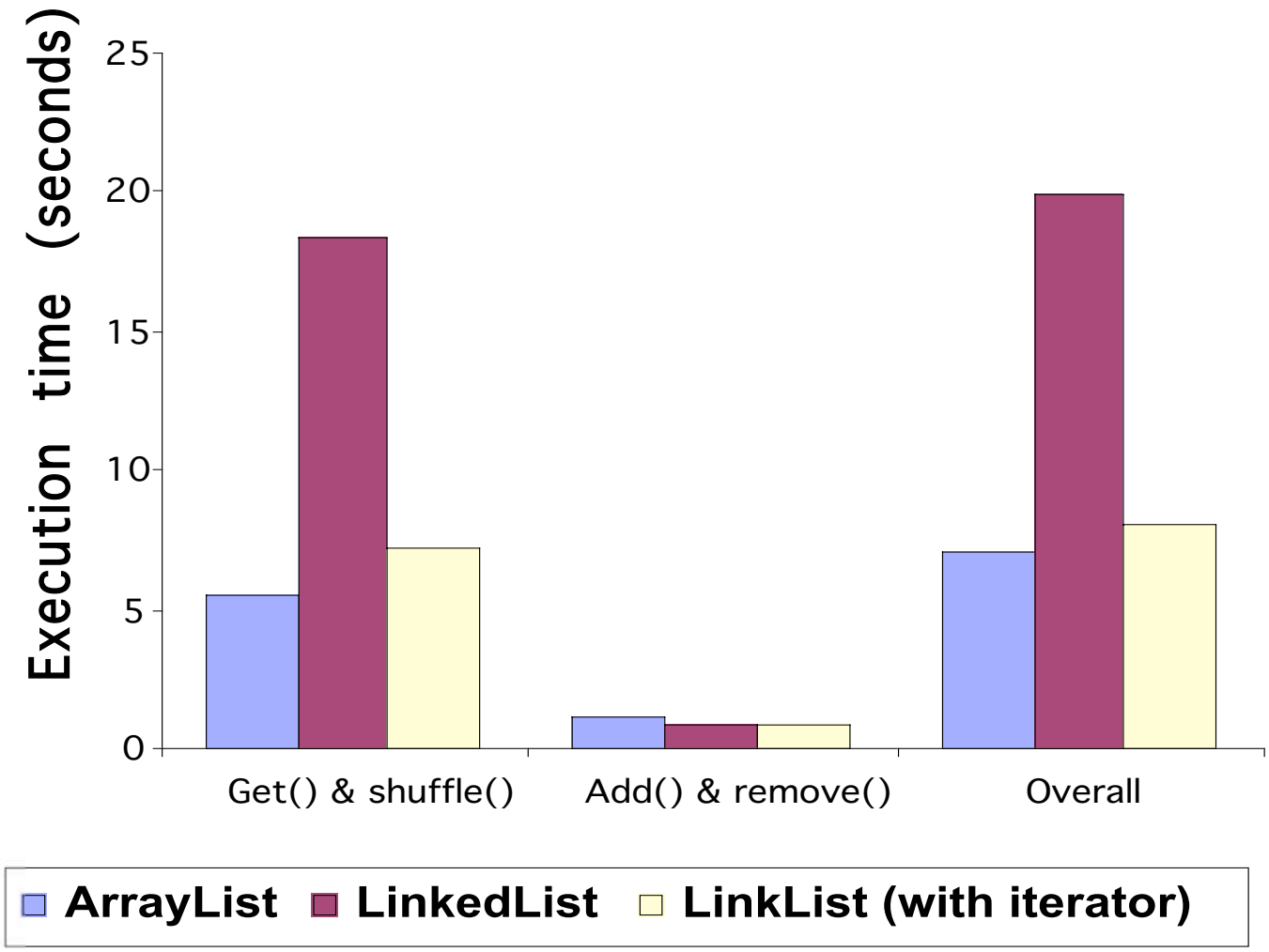
Software engineering perspective

- Profiling the program
  - Identify the bottlenecks
  - Determine the factors that affect performance
- Proper design
  - Eliminate bottlenecks
  - Improve scalability

Motivation

- The NOM simulation model is a typical large scale scientific application model
  - Long running time
  - Large amount of data output
  - Computation and I/O intensive
- Expect that our experiences can help other scientific applications developers using Java

Choice of data structure

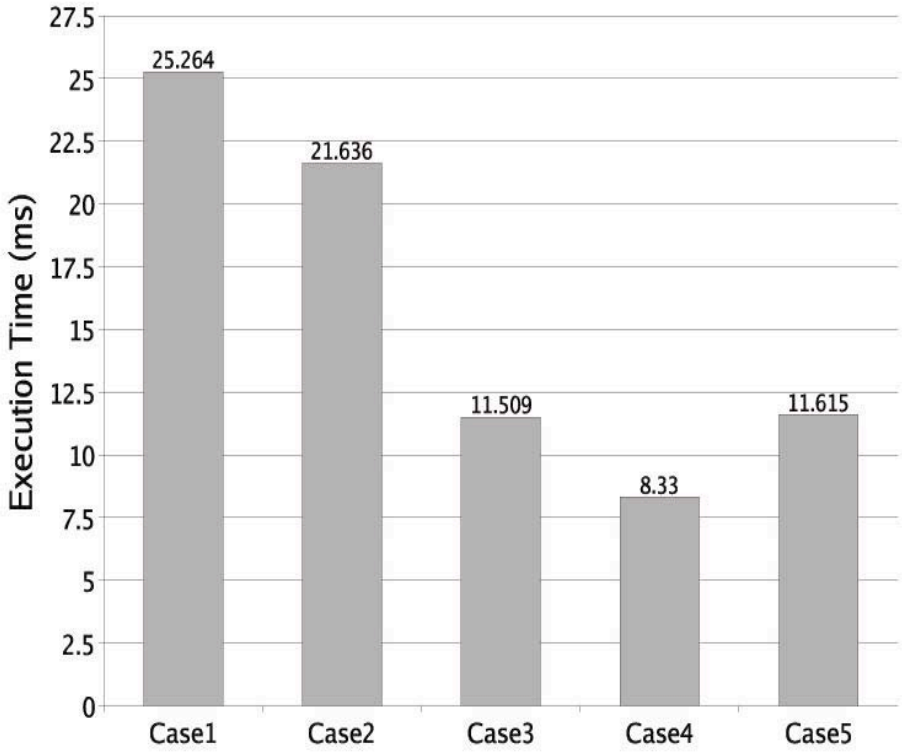


Object reuse

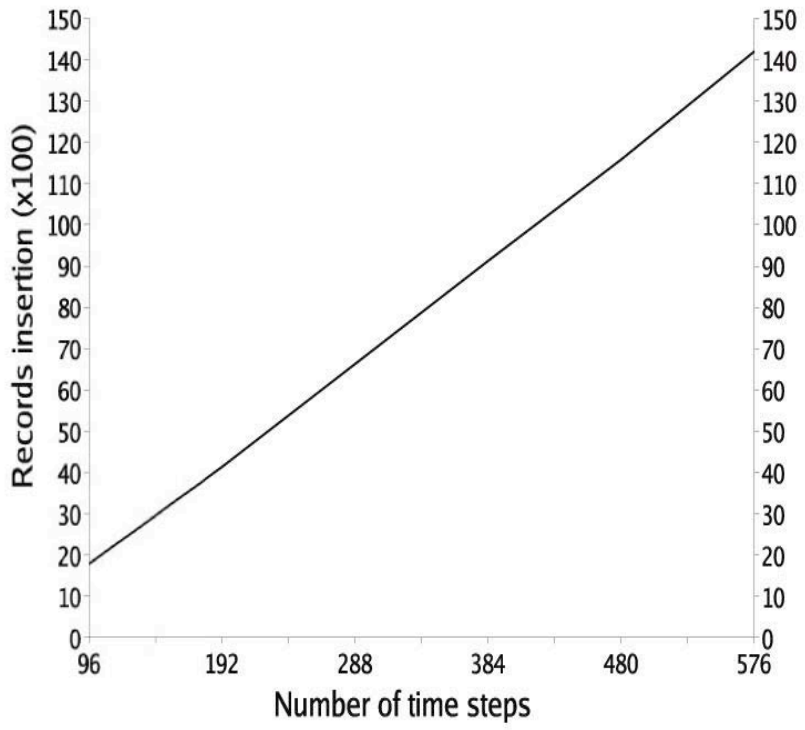
- Object allocation 50% longer than in C++. (Sosnoski, 1999)
- Excessive object creation:
  - Increases the memory footprint
  - Forces more CPU cycles to be used for garbage collection
  - Increases the possibility of a memory leak
- Object reuse
  - Isolate the object
  - Reinitialize the object
  - Object pool management

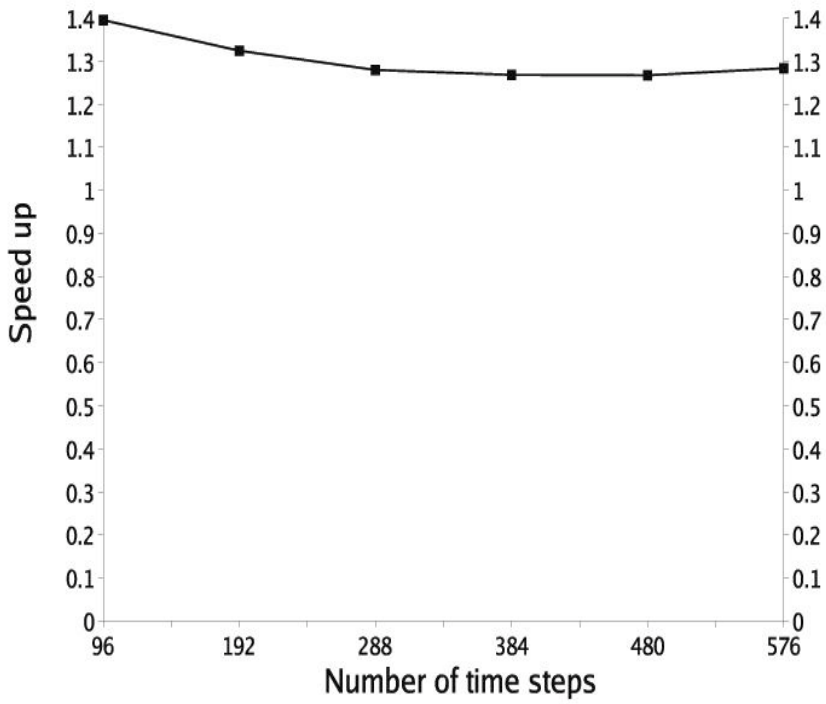
JDBC

- Benchmark:
  - Case 1: Statement
  - Case 2: PreparedStatement
  - Case 3: Statement with transaction management
  - Case 4: PreparedStatement with transaction management
  - Case 5: Batch updates

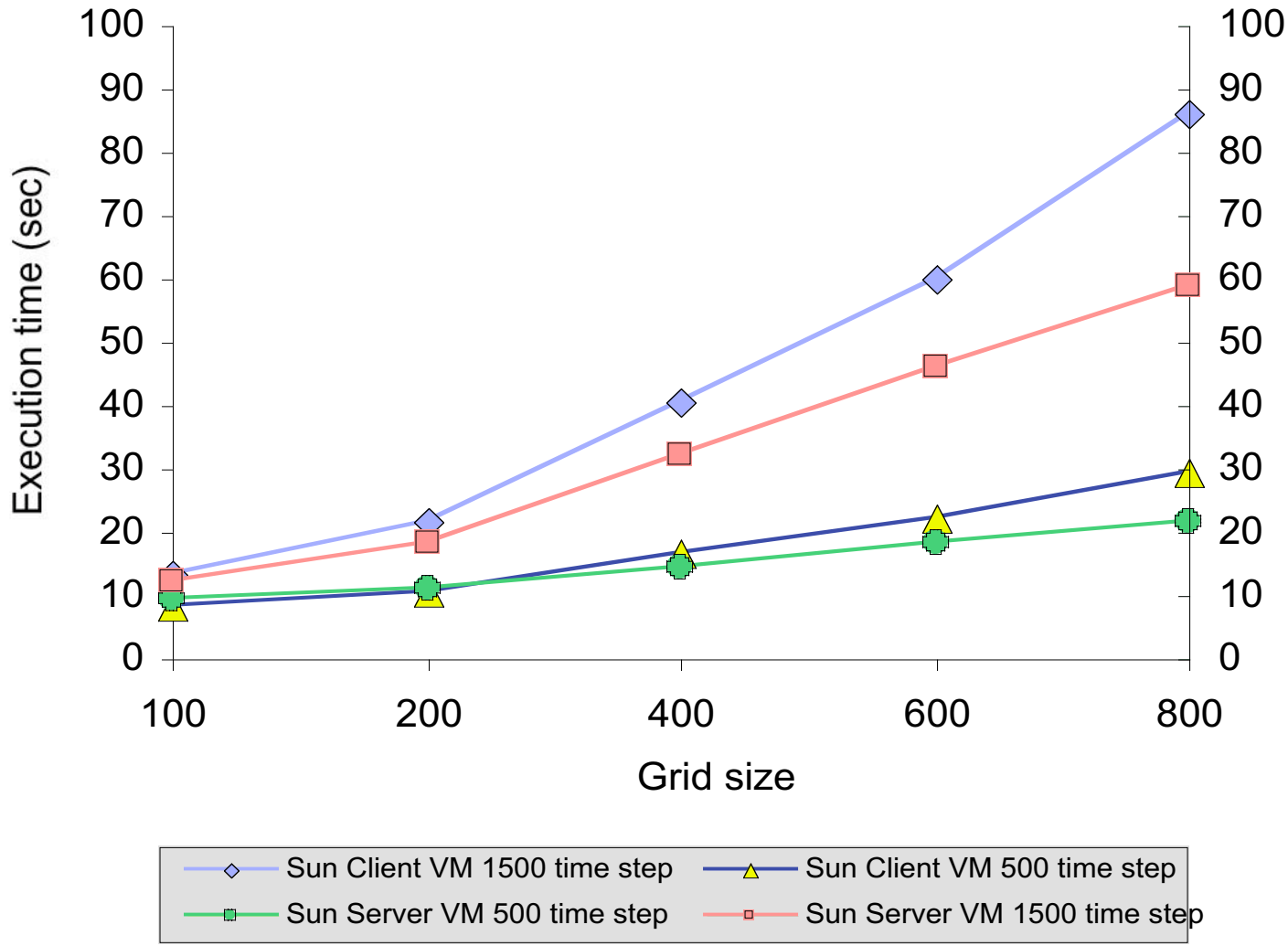


Parallel data output with Java threads

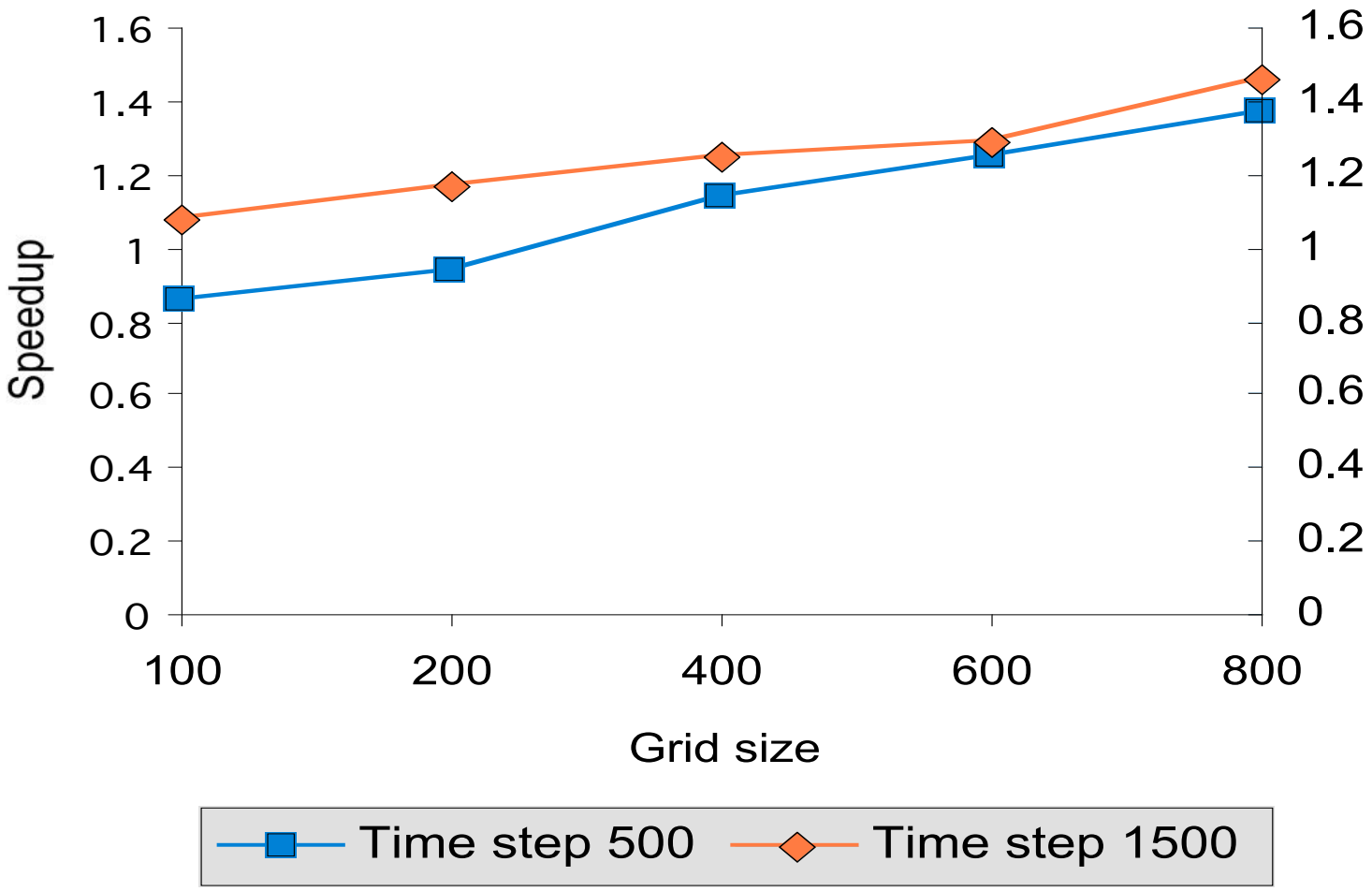




Choice of JVM



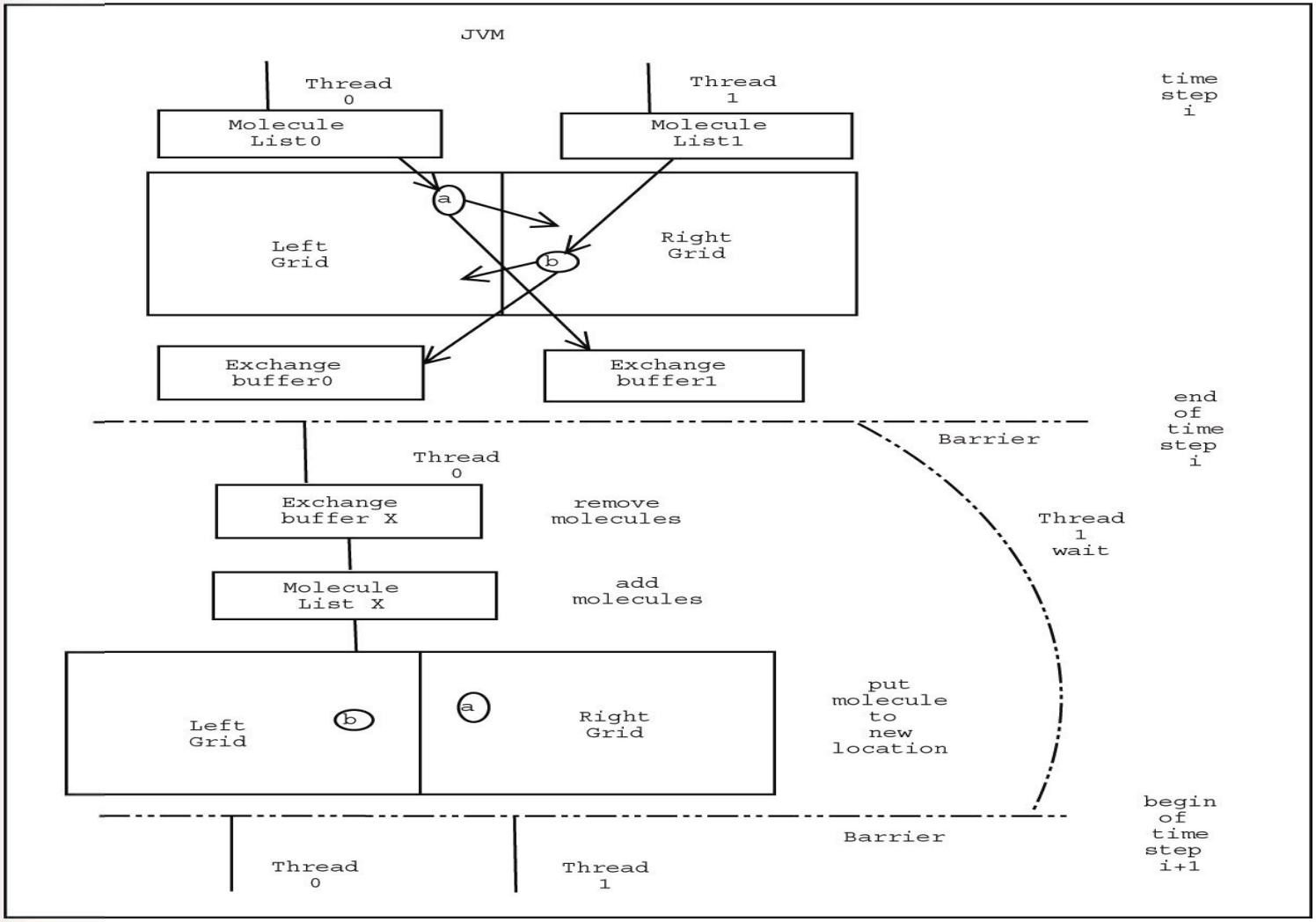
Choice of JVM (cont.)



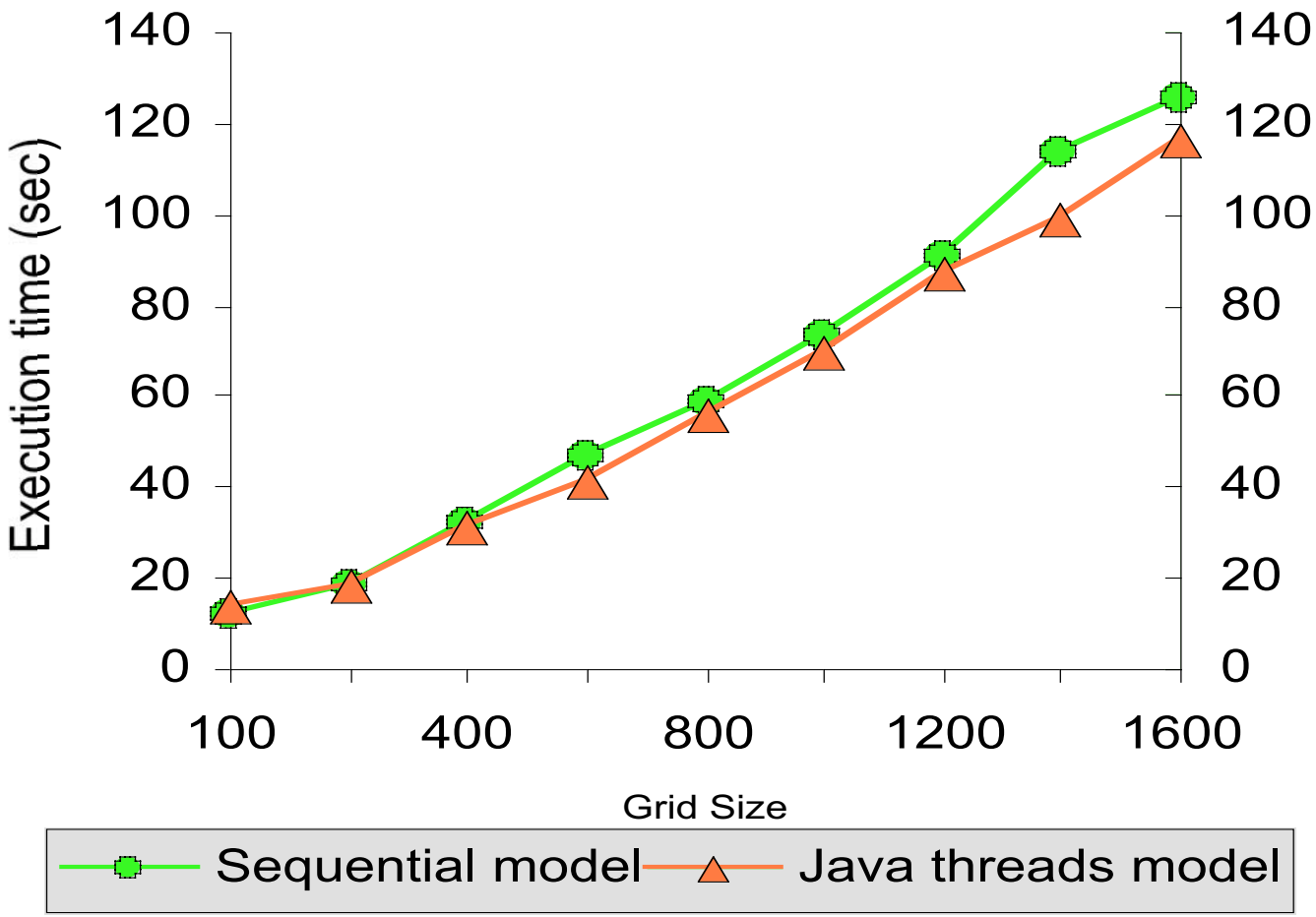
Scalability

- Two approaches:
  - Java built-in threads
    - Single JVM, shared memory
  - Java MPI (MPJ)
    - Multiple JVMs, distributed memory
- Equally separate the grid to 2 or 4 subset grids
- Synchronize all the threads or processes at each time step

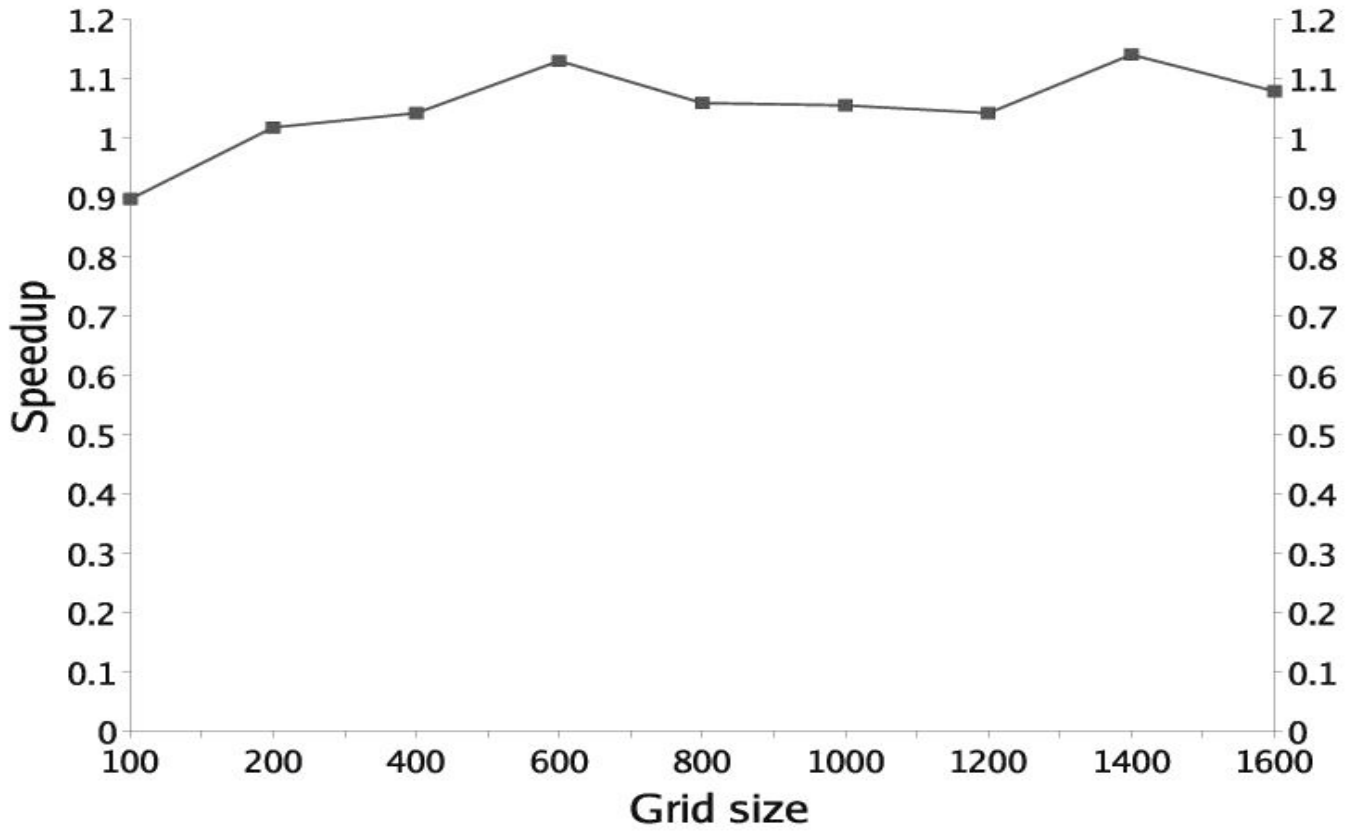
Java thread model



Java thread model (cont.)



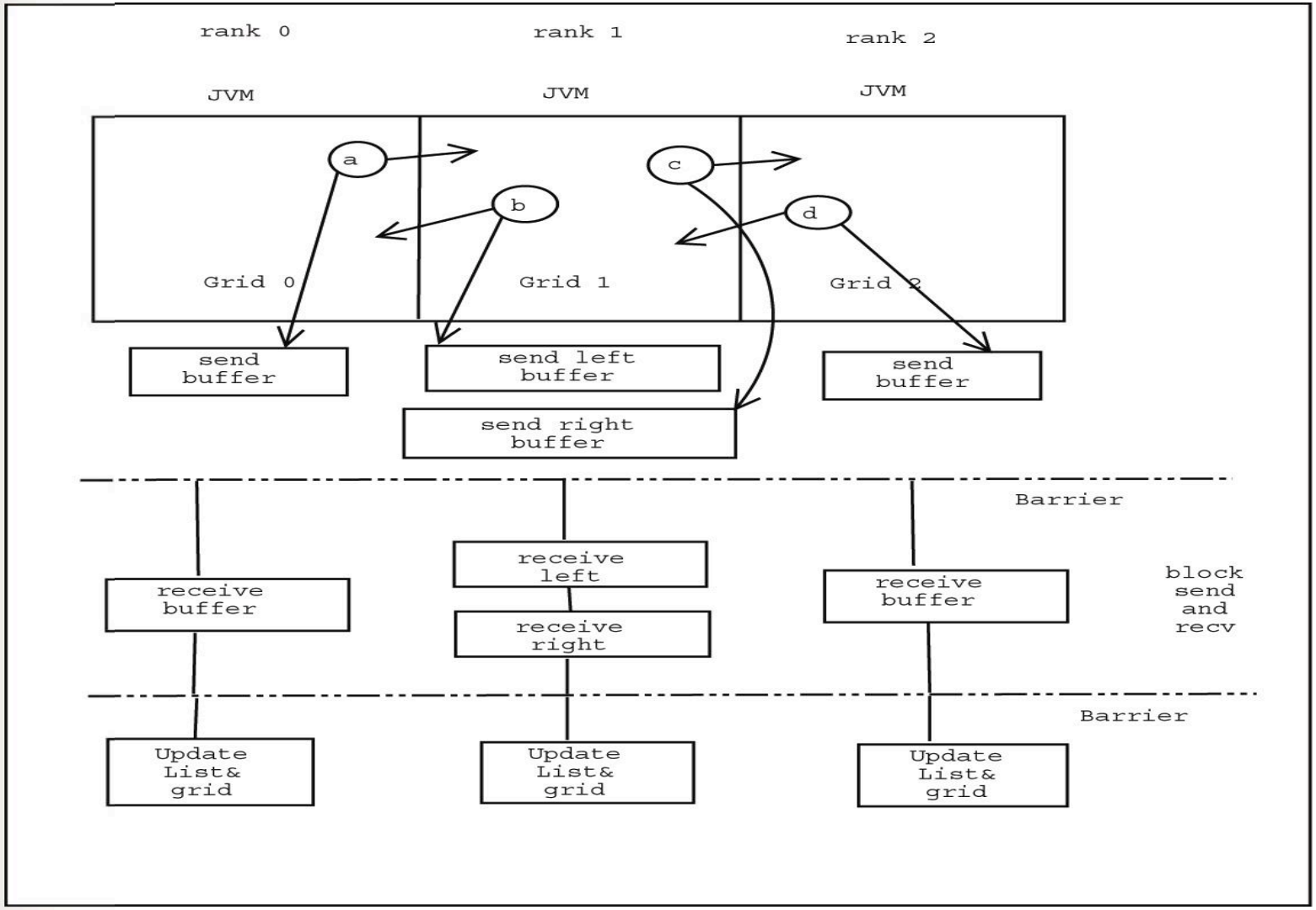
Java thread model (cont.)



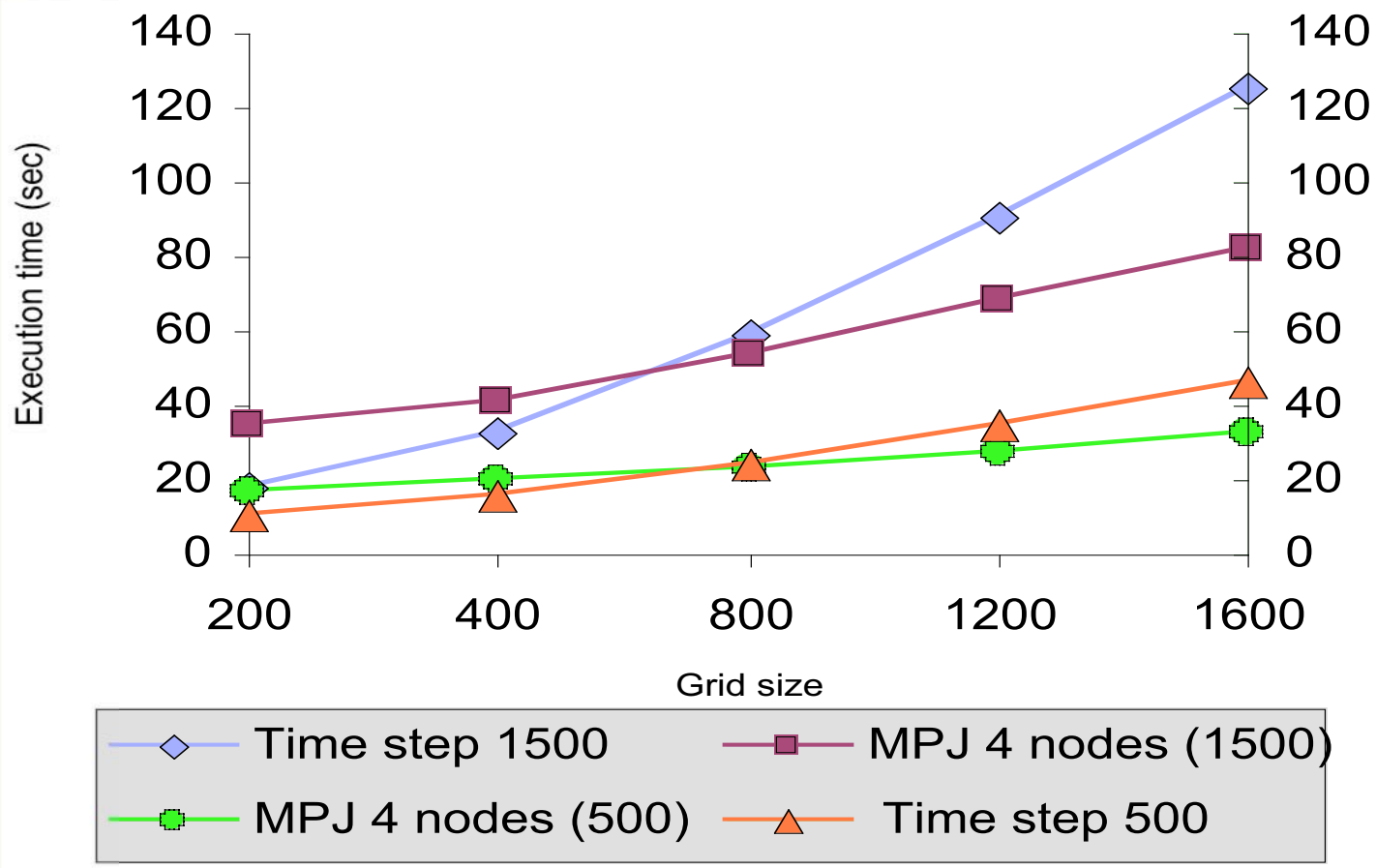
Message passing in Java (MPJ)

- MPJ specification
  - B. Carpenter et al: MPJ: MPI-like message passing for Java. Concurrency: Practice and Experience. 2000
- MPJ implementation
  - MPI wrapper
    - mpiJava (M. Baker et al: mpiJava: An object-oriented Java Interface to MPI. 1999)
  - Pure Java implementation
    - JMPI (S. Morin et al: JMPI: implementing the message passing standard in Java. 2002)
    - Jmpi (K. Dincer: Ubiquitous message passing interface implementation in Java: jmpi. 1999)
    - MPIJ (G. Judd: Dogma: Distributed object group management architecture. 1998)

Distributed memory model (MPJ)

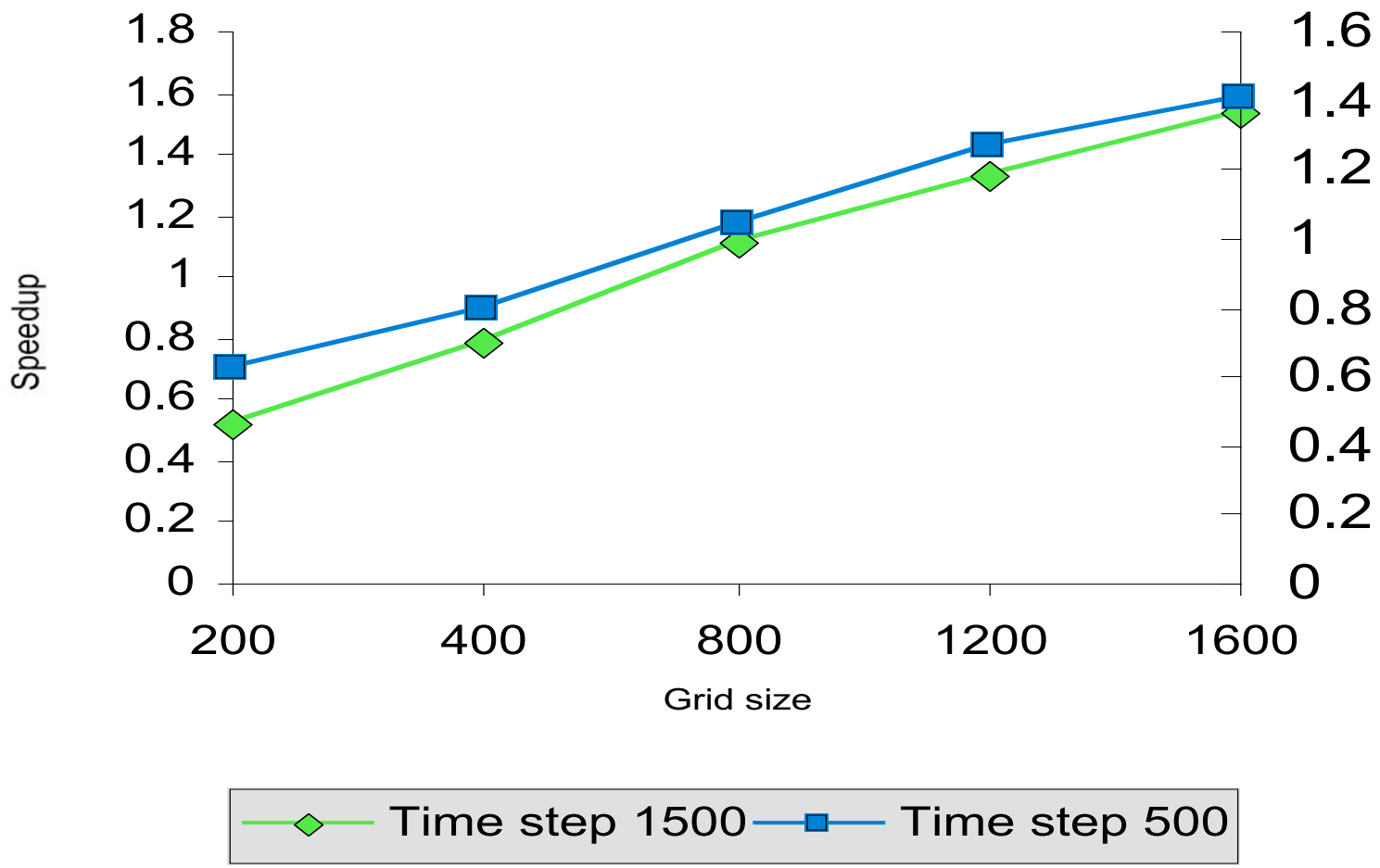


Distributed memory model

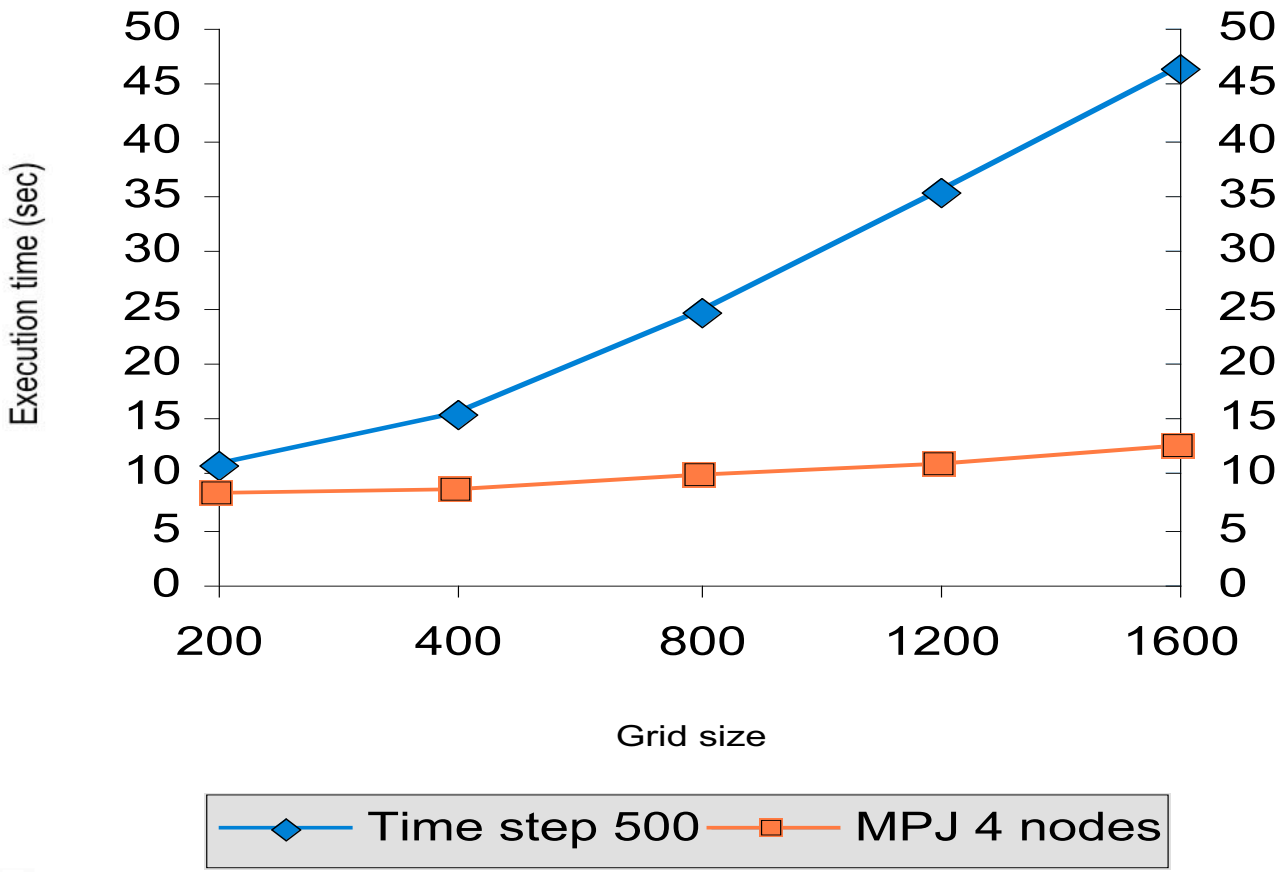


- LAM MPI, mpiJava. 4 machine in a cluster

Distributed memory model (cont.)

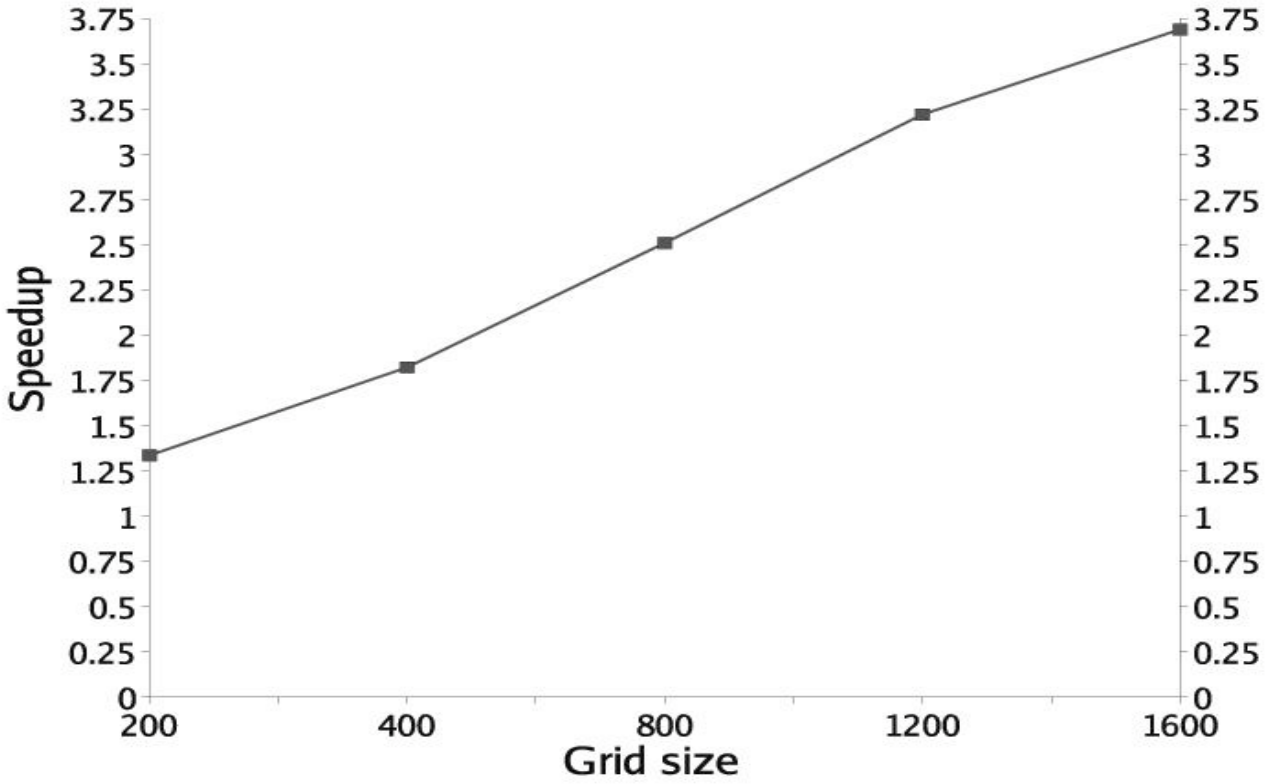


Distributed memory model (cont.)



- No communication between processes

Distributed memory model (cont.)



Other issue

- High performance compiler
  - GCJ
    - Depends on the applications
  - IBM High-Performance compiler for RS6000 architecture
- Code clean up

Summary - potential 25x

Approaches	Speedup	Comments
Choice of data structure	2.8	Evaluate overall performance
Object reuse	-	Performance gain is small
JDBC	3	Use different JDBC technologies
Parallel data output	1.3	Overlap the computation and I/O
Choice of JVM	1.4	IBM JVM is valuable to evaluate
Java threads model	1.1	Evaluate different OS
MPJ model with communication	1.5	Reduce the communication overhead

Outline

- **Introduction**
- **Modeling**
- Core simulation engine
- **Intelligent Web-based interface**
- **The NOM collaboratory**
- **Java performance analysis**
- **Conclusion**
- Future work

Conclusion

- Agent-based stochastic model for simulating the NOM evolution with discrete temporal and spatial properties
- A Web-based interface
- The NOM collaboratory
- Java performance analysis for large scale scientific applications

Outline

- Introduction
- Modeling
- Core simulation engine
- Intelligent Web-based interface
- The NOM collaboratory
- Java performance analysis
- Conclusion
- **Future work**

Future work

- Model testing
  - Testing on the sorption
  - More features need to add into the core simulation engine
  - Discrete event vs. Time step
- Advanced algorithm for search similar simulations
- Delicate way to save the JVM state and restart the simulation
- Collaboratory:
  - More communication tools
  - More simulation models for NOM study
  - NOML extension

Contributions

- New approach for NOM modeling
  - Agent-based modeling
- E-Science on the Web
- Intelligent interface components
- Built the NOM Collaboratory
- Performance analysis for scientific applications

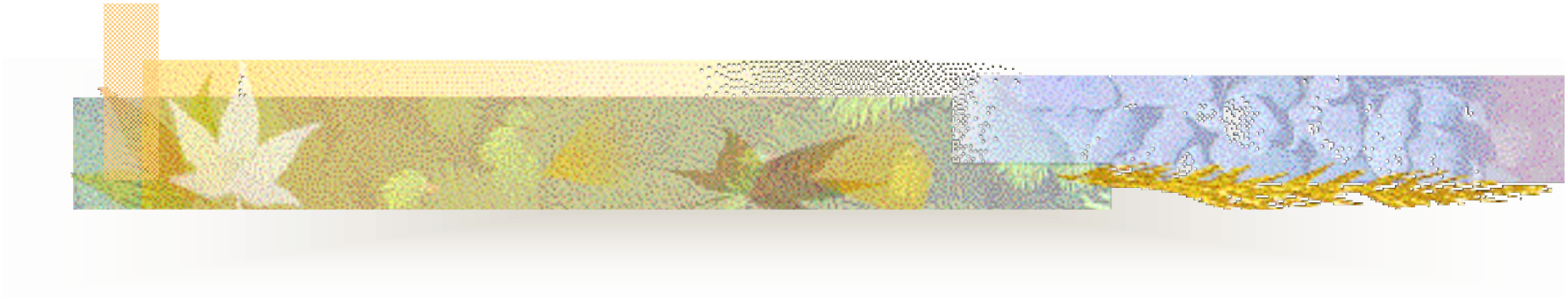
Publications to date

- **Proceedings**
  - Xiang, X., and Madey, G., "Exploring Performance Improvement for Java-based Scientific Simulations that use the Swarm Toolkit", 7th Swarm Researchers Conference (Swarm2003), Notre Dame, IN April 2003.
  - Huang, Y., Madey, G., Xiang, X., and Chanowich, E., "Web-based Molecular Modeling Using Java/Swarm, J2EE and RDBMS Technologies", 7th Swarm Researchers Conference (Swarm2003), Notre Dame, IN, April 2003.
- **Abstracts**
  - Madey, G., Huang, Y., Xiang, X., "Complex System Simulation: Interactions of NOM Molecules, Mineral Surfaces, and Microorganisms in Soils", USGS Workshop on Modeling Complex Systems, Reno, NV, November 2002.
  - Madey, G., Huang, Y., Xiang, X., and Chanowich, E., "Agent-Based Simulation of Biocomplexity: NOM, Mineral Surfaces, and Micro-Organisms", ASLO 2003 Aquatic Sciences Meeting, Salt Lake City, UT, February 2003
  - Cabaniss, S., Madey, G., Maurice, P., Leff, L., Huang, Y., and Xiang,X., "Stochastic synthesis model for the evolution of natural organic matter", 225th American Chemical Society National Meeting, New Orleans, LA, March 2003.
- **Posters**
  - Huang, Y., Xiang, X, Chanowich, E., Madey, G., "A Web-Based Stochastic Simulation of Natural Organic Matter", Annual Environmental Education and Research (NDEER) Symposium, Notre Dame, IN, November 2003.
  - Xiang, X., and Madey, G., "Exploring Performance Improvement for Java-based Scientific Simulations that use the Swarm Toolkit", Indiana Biocomplexity Symposium, Notre Dame, IN, April 2003.

Publication planning

- NOM simulation
  - World Conference on Natural Resource modeling
  - Simulation Practice and Theory, International Journal of the Federation of European Simulation Societies – EUROSIM
  - SIAM Journal on Scientific Computing
- Performance analysis of Java for Scientific Applications
  - Winter Simulation Conference
  - Joint ACM Java Grande – ISCOPE Conference
  - High performance computing and networking (HPCN)
  - IBM Systems Journal – Java performance
- Scientific Collaboratory
  - ACM Conference on Computer Supported Cooperative work (CSCW)
  - Information Resources Management Association, IRMA international conference
  - ACM Collaborative Virtual Environments
  - International Conference on Human Computer Interaction
  - Communications of the ACM

Acknowledgement

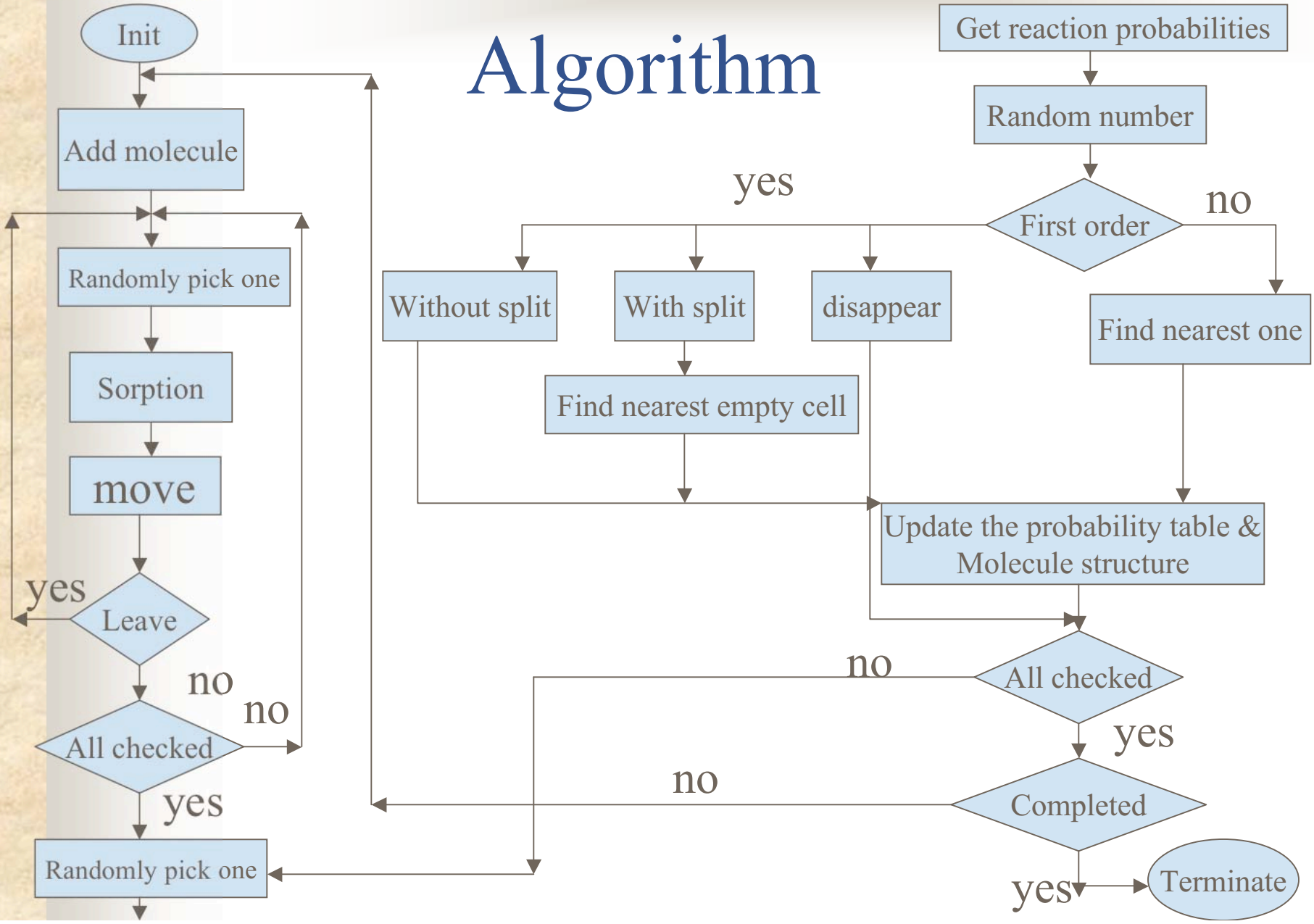


Thank you

Questions?

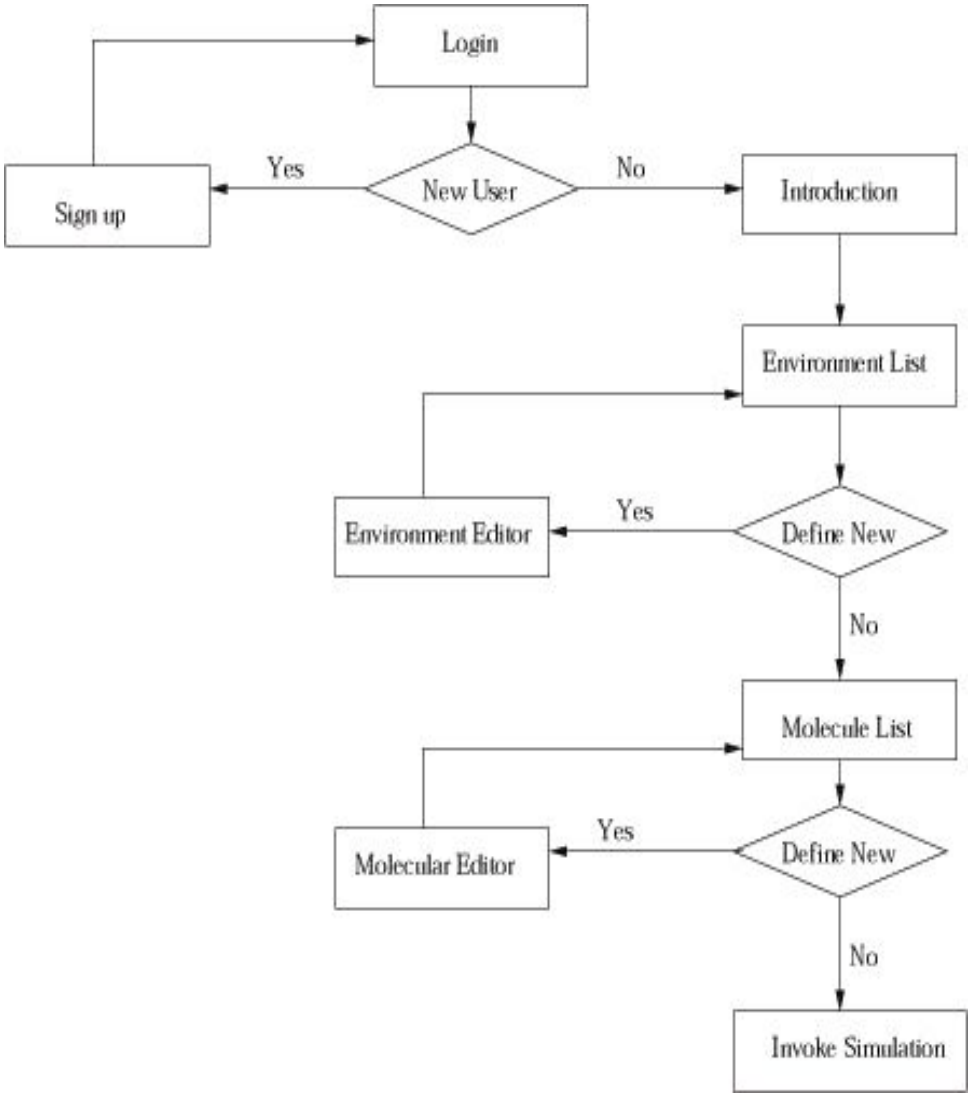
- A Java-based Direct Monte Carlo Simulation of a NanoScale Pulse Detonation Engine (2002)
- ESG (Environment Scenario Generator) http://msea.afccc.af.mil/html/projects.html

Algorithm


Init
Add molecule
Randomly pick one
Sorption
move
Leave
yes
no
All checked
no
yes
Randomly pick one
Get reaction probabilities
Random number
First order
yes
no
Without split
With split
disappear
Find nearest one
Find nearest empty cell
Update the probability table &
Molecule structure
All checked
no
yes
Completed
no
yes
Terminate

Web-based interface

- Input the simulation parameters
- Invoke the simulation
- Stop the simulation
- View the real-time simulation results



Scalability