Computational Geometry Aspects of Monte Carlo Approaches to PDE Problems in Biology, Chemistry, and Materials

Prof. Michael Mascagni

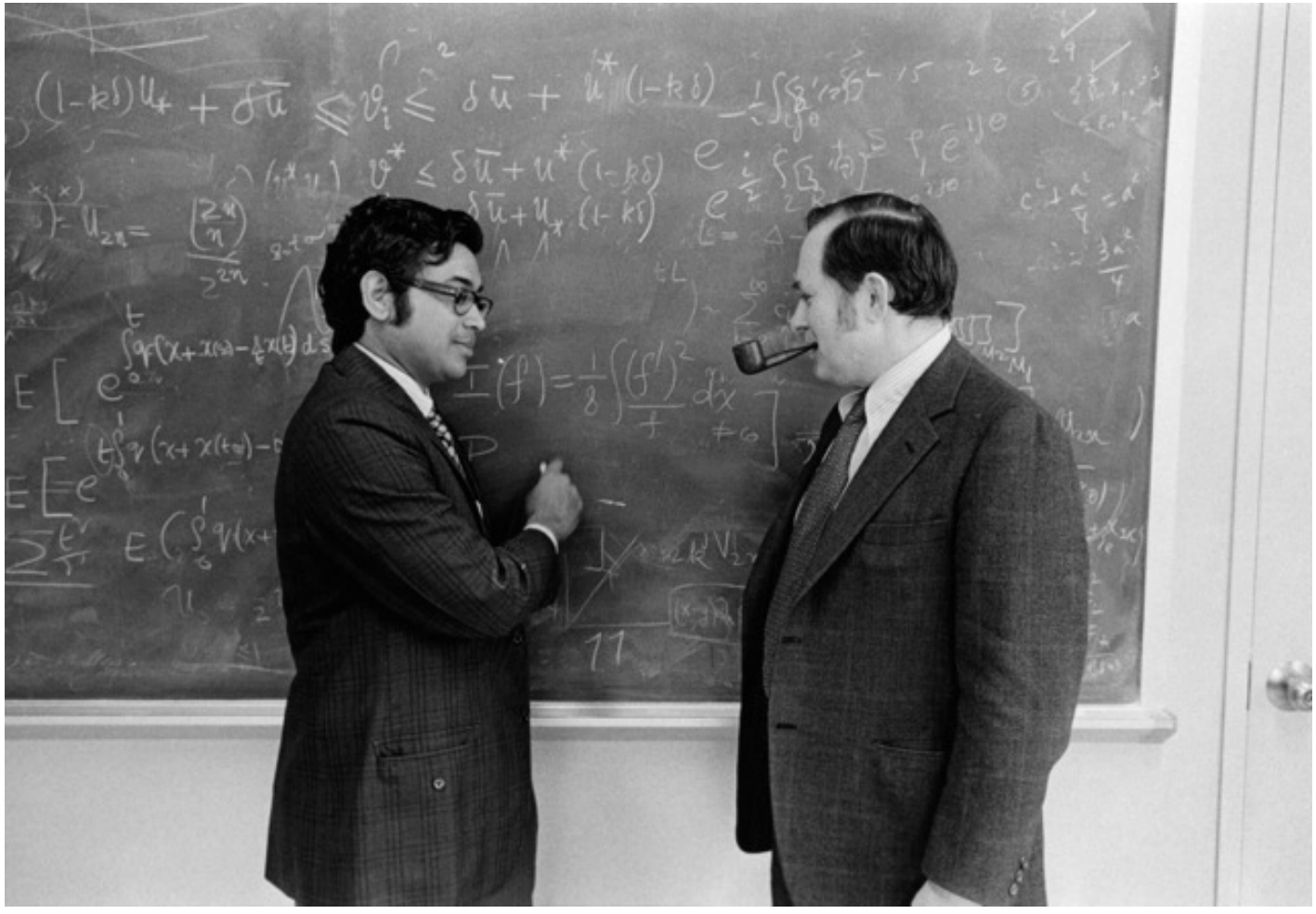
Department of Computer Science
Department of Mathematics
Department of Scientific Computing
Graduate Program in Molecular Biophysics
Florida State University, Tallahassee, FL 32306 **USA**
AND
Applied and Computational Mathematics Division, Information Technology Laboratory
National Institute of Standards and Technology, Gaithersburg, MD 20899-8910 **USA**
E-mail: mascagni@fsu.edu or mascagni@nist.gov
URL: http://www.cs.fsu.edu/∼mascagni

Research supported by ARO, DOE, NASA, NATO, NIST, National Science Fund (Bulgaria) & NSF, with equipment donated by Intel and Nvidia

Outline of the Talk

- Monte Carlo Methods for PDEs
  - A Little History on Monte Carlo Methods for PDEs
- Some Examples Using This for Computing Elliptic Problems
  - The Walk on Spheres Method
  - Biochemical Problems
- Generalization of the Monte Carlo Approach
  - Computational Geometry
  - Important Computational Paradigm: Computing Capacitance
  - Capacitance as a Computational Paradigm
  - Parallel Scalability
  - Scalability Plots
- Conclusions

Dedicated to My Probability Professors at Courant: Raghu Varadhan and Monroe Donsker



In Memoriam: Nikolai Simonov, 1956–2019

Figure: Died in an avalanche in the Altai Mountains, May 6, 2019

Early History of MCMs for PDEs

1. Courant, Friedrichs, and Lewy: Their pivotal 1928 paper has probabilistic interpretations and MC algorithms for linear elliptic and parabolic problems
2. Fermi/Ulam/von Neumann: Atomic bomb calculations were done using Monte Carlo methods for neutron transport, their success inspired much post-War work especially in nuclear reactor design
3. Kac and Donsker: Used large deviation calculations to estimate eigenvalues of a linear Schrödinger equation
4. Forsythe and Leibler: Derived a MCM for solving special linear systems related to discrete elliptic PDE problems

Integration: The Classic Monte Carlo Application

1. Consider computing $I = \int_0^1 f(x)\,dx$
2. Conventional quadrature methods:

$$I \approx \sum_{i=1}^{N} w_i f(x_i)$$

   - Standard quadrature is of this form with deterministic error bounds
   - If we hold work, $f(x_i)$, constant as dimension increases we see the MC advantage vs. the curse of dimensionality
3. Monte Carlo method has two parts to estimate a numerical quantity of interest, I
   - The random process/variable: $x_i \sim U[0,1]$ i.i.d.
   - The estimator or score: $f(x_i)$
   - One averages and uses a confidence interval for an error bound

$$\bar{I} = \frac{1}{N}\sum_{i=1}^{N} f(x_i), \quad var(I) = \frac{1}{N-1}\sum_{i=1}^{N}(f(x_i)-\bar{I})^2 = \frac{1}{N-1}\left[\sum_{i=1}^{N} f(x_i)^2 - N\bar{I}^2\right],$$

$$var(\bar{I}) = \frac{var(I)}{N}, \quad I \in \bar{I} \pm k \times \sqrt{var(\bar{I})}$$

Other Early Monte Carlo Applications

- Numerical linear algebra based on sums: $S = \sum_{i=1}^{M} a_i$
  1. Define $p_i \geq 0$ as the probability of choosing index i, with $\sum_{i=1}^{M} p_i = 1$, and $p_i > 0$ whenever $a_i \neq 0$
  2. Then a_i/p_i with index i chosen with $\{p_i\}$ is an unbiased estimate of S, as $E[a_i/p_i] = \sum_{i=1}^{M} \left(\frac{a_i}{p_i}\right) p_i = S$
- Can be used to solve linear systems of the form $x = Hx + b$
- Consider the linear system: $x = Hx + b$, if $||H|| = \mathbb{H} < 1$, then the following iterative method converges:

$$x^{n+1} := Hx^n + b, \quad x^0 = 0,$$

and in particular we have $x^k = \sum_{i=0}^{k-1} H^i b$, and similarly the Neumann series converges:

$$N = \sum_{i=0}^{\infty} H^i = (I - H)^{-1}, \quad ||N|| = \sum_{i=0}^{\infty} ||H^i|| \leq \sum_{i=0}^{\infty} \mathbb{H}^i = \frac{1}{1 - \mathbb{H}}$$

- Formally, the solution is $x = (I - H)^{-1} b$

More Modern Monte Carlo Applications

- Methods for partial differential and integral equations based on random walks/Markov chains (no need to find a discrete approximation to the PDE/IE)
  1. Integral equation methods are similar in construction to the linear system methods
  2. PDEs can be solved by using the Feynman-Kac formula
  3. Some Monte Carlo methods can now beat deterministic solvers (electrostatics)
- Efficient methods that exploit fast probabilistic application of a linear operator
- Modern sampling methods linear algebra (SVD) based loosely on the Johnson-Lindestrauss projection method
- Generation of random fields
- Stochastic DEs and PDEs
- Financial computing
- Uncertainty quantification (UQ)

The First Passage (FP) Probability is the Green's Function

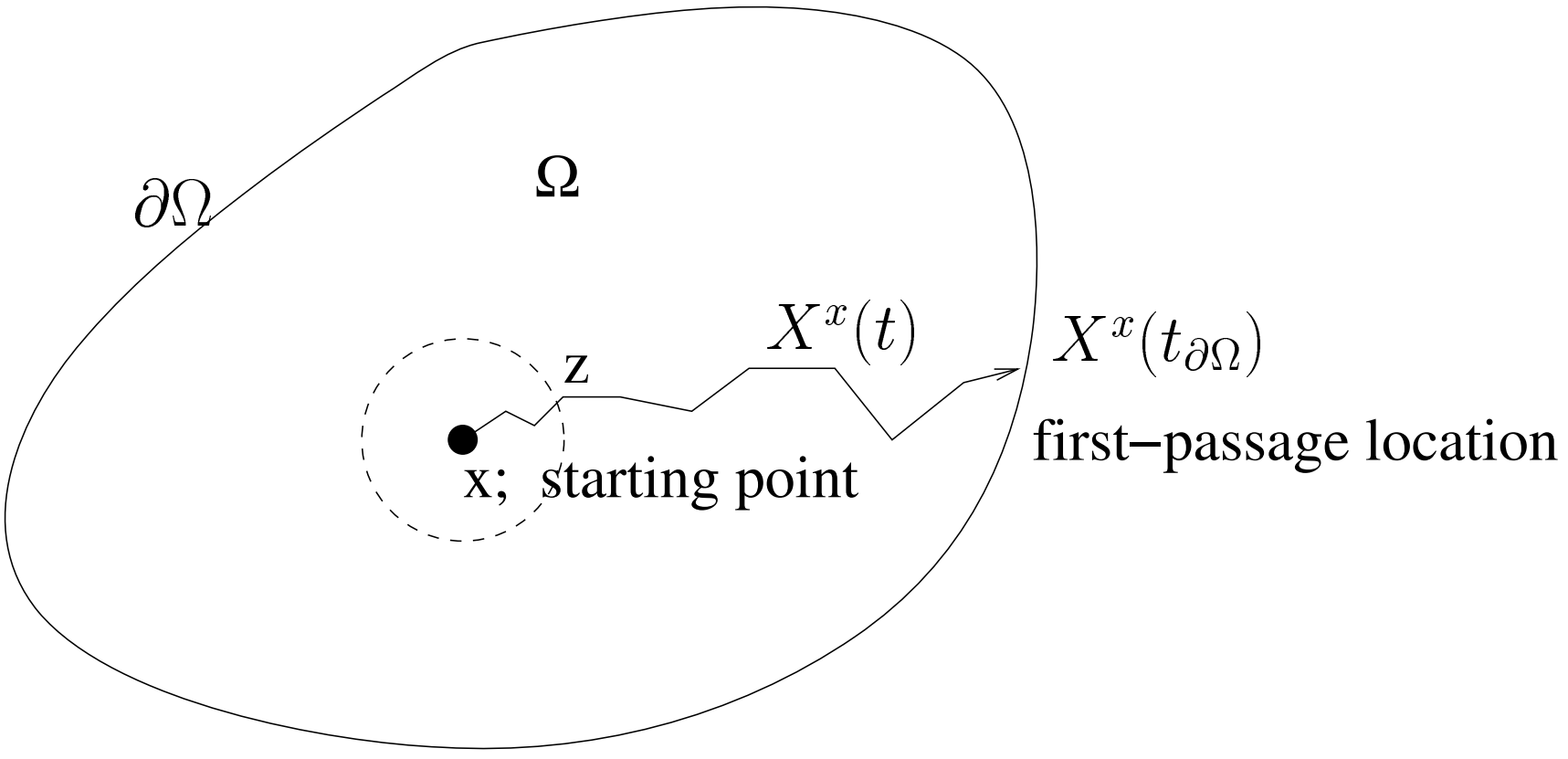
Back to our canonical elliptic boundary value problem:

$$\frac{1}{2}\Delta u(x) = 0, \quad x \in \Omega$$
$$u(x) = f(x), \quad x \in \partial\Omega$$

- Distribution of z is uniform on the sphere
- Mean of the values of $u(z)$ over the sphere is $u(x)$
- $u(x)$ has mean-value property and harmonic
- Also, $u(x)$ satisfies the boundary condition

$$u(x) = \mathbb{E}_x[f(X^{\mathbf{x}}(t_{\partial\Omega}))] \tag{1}$$

The First Passage (FP) Probability is the Green's Function



The First Passage (FP) Probability is the Green's Function

Reinterpreting as an average of the boundary values

$$u(x) = \int_{\partial\Omega} p(x,y)\, f(y)\, dy \tag{2}$$

Another representation in terms of an integral over the boundary

$$u(x) = \int_{\partial\Omega} \frac{\partial g(x,y)}{\partial \mathbf{n}}\, f(y)\, dy \tag{3}$$

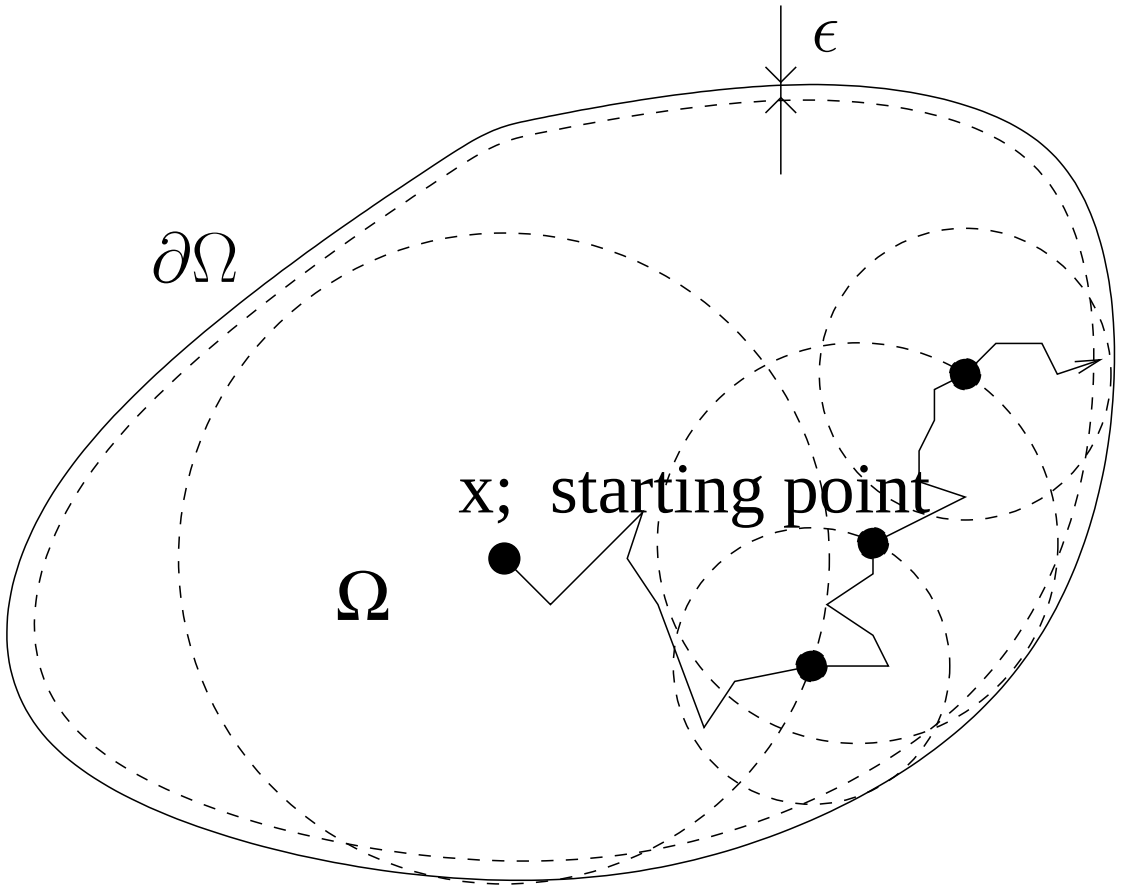
$g(x,y)$ – Green's function of the Dirichlet problem in Ω

$$\Longrightarrow p(x,y) = \frac{\partial g(x,y)}{\partial \mathbf{n}} \tag{4}$$

'Walk on Spheres' (WOS) and 'Green's Function First Passage' (GFFP) Algorithms

- Green's function is known
  $\Longrightarrow$ direct simulation of exit points and computation of the solution through averaging boundary values
- Green's function is unknown
  $\Longrightarrow$ simulation of exit points from standard subdomains of Ω, e.g. spheres
  $\Longrightarrow$ Markov chain of 'Walk on Spheres' (or GFFP algorithm)
  $x_0 = x, x_1, \ldots, x_N$
  $x_i \to \partial\Omega$ and hits ε-shell is $N = O(|\ln(\varepsilon)|)$ steps
  x_N simulates exit point from Ω with $O(\varepsilon)$ accuracy

'Walk on Spheres' (WOS) and 'Green's Function First Passage' (GFFP) Algorithms



Architectural Considerations as We Move Towards the Exascale

- Some trends in HPC architectures
  1. Memory per processor/core has inflected and is now decreasing
  2. Long-term trend is that memory bandwidth is the limiting factor for performance and cost
  3. High clock rates and high bandwidth communication lead to high energy consumption and hot boxes that need cooling
- These Monte Carlo algorithms avoid all three of issues due to their innate performance
  1. Minimal memory usage has always been a benefit of Monte Carlo methods
  2. Independent sampling means that the communication to computation ratio is extremely small and tunable
- Monte Carlo is a very simple computational paradigm to explore fundamental aspects of parallelism, algorithmic resilience, fault-tolerance

Continuum Biochemical Electrostatics

Motivation

- Experimental Data: Folding, stability & binding behavior of biomolecules can be modulated by changes in salt concentration
- Physical Model: Implicit solvent-based Poisson-Boltzmann model can provide accurate predictions of salt dependent behavior of biomolecules
- Mathematical Model: Elliptic boundary-value problems

Specific Problems

- Electrostatic free energy for linear case: only finite number of electrostatic potential point values
- Dependence of energy on geometry: needs accurate treatment
- Singularities in solution: have to be taken into account analytically
- Behavior at infinity: must be exactly enforced
- Functional dependence on salt concentration: needs accurate estimate

Mathematical Model: Molecular Geometry

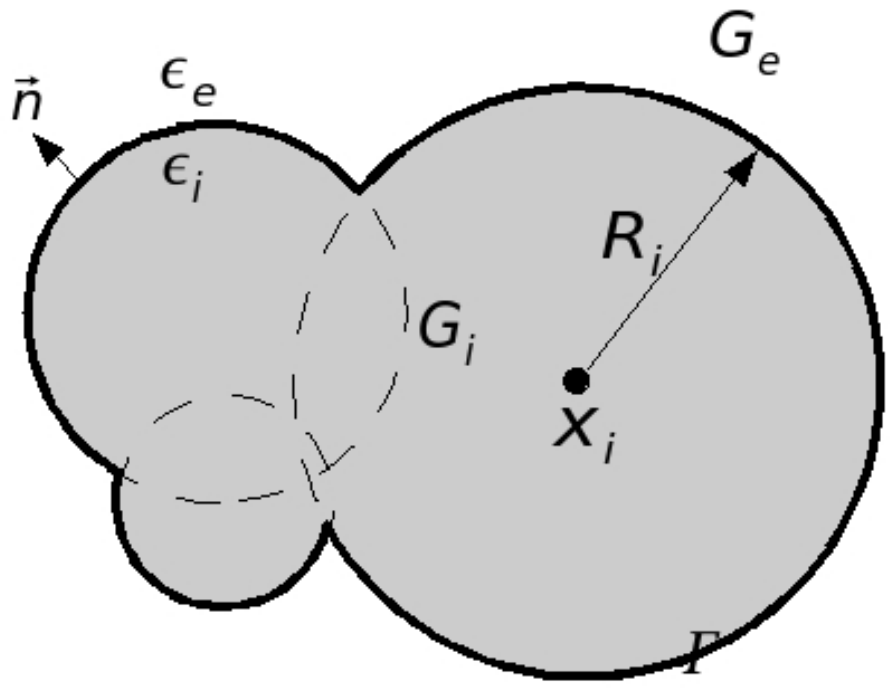


Figure: Biomolecule with dielectric ϵ_i and region region G_i is in solution with dielectric ϵ_e and region G_e. On the boundary of the biomolecule, electrostatic potential and normal component of dielectric displacement continue

Mathematical Model: Partial Differential Equations

- Poisson equation for the electrostatic potential, Φ_i, and point charges, Q_m, inside a molecule (in CGS units):

$$\epsilon_i \Delta \Phi_i(x) + 4\pi \sum_{m=1}^{M} Q_m \delta(x - x^{(m)}) = 0 \ , \ x \in G_i$$

- For 1-1 salt (such as *NaCl*) Poisson-Boltzmann equation (PBE):

$$\Delta \Phi_e(x) - \kappa^2 \sinh(\Phi_e(x)) = 0 \ , \ x \in G_e \ ,$$

but we only consider the linearized PBE:

$$\Delta \Phi_e(x) - \kappa^2 \Phi_e(x) = 0 \ , \ x \in G_e$$

- For one-surface model: continuity condition on the dielectric boundary

$$\Phi_i = \Phi_e \ , \ \epsilon_i \frac{\partial \Phi_i}{\partial n(y)} = \epsilon_e \frac{\partial \Phi_e}{\partial n(y)} \ , \ y \in \Gamma$$

Electrostatic Potential and Energy

- Point values of the potential: $\Phi(x) = \Phi_{rf}(x) + \Phi^c(x)$
  Here, singular part of Φ:

$$\Phi^c(x) = \sum_{m=1}^{M} \frac{Q_m}{|x - x^{(m)}|}$$

- Reaction field electrostatic free energy of a molecule is linear combination of point values of the regular part of the electrostatic potential:

$$W_{rf} = \frac{1}{2} \sum_{m=1}^{M} \Phi_{rf}(x^{(m)}) Q_m \ ,$$

- Electrostatic solvation free energy = difference between the energy for a molecule in solvent with a given salt concentration and the energy for the same molecule in vacuum:

$$\Delta G^{elec}_{solv} = W_{rf}(\epsilon_i, \epsilon_e, \kappa) - W_{rf}(\epsilon_i, 1, 0)$$

The Feynman-Kac Formula

- If we set $f(x) = 0$ and have $g(x) \neq 0$, the solution is

$$u(y) = \mathbb{E}\left[\int_0^{\tau_{\partial\Omega}} g(\beta_y(s))\, ds\right]$$

- By linear superposition, the solution to Poisson equation is given probabilistically as

$$u(y) = \mathbb{E}\left[\int_0^{\tau_{\partial\Omega}} g(\beta_y(s))\, ds + f(\beta_y(\tau_{\partial\Omega}))\right]$$

- The linearized Poisson-Boltzmann equation is given by

$$\Delta u(x) - \kappa^2 u(x) = 0, \;\; x \in \Omega, \;\; u(x) = f(x), \;\; x \in \partial\Omega, \;\; u \to 0 \text{ as } |x| \to \infty$$

and has Wiener integral representation:

$$u(y) = \mathbb{E}\left[f(\beta_y(\tau_{\partial\Omega}))e^{-\int_0^{\tau_{\partial\Omega}} \kappa^2\, ds}\right]$$

'Walk-on-Spheres' Algorithm

- Walk-on-spheres (WOS) algorithm for general domains with a regular boundary
- Define a Markov chain $\{x_i,\ i = 1, 2, \ldots\}$
- Set $x_0 = x^{(m)}$ for some m, $x_i = x_{i-1} + d_i\omega_i\ ,\quad i = 1, 2, \ldots$, where
  1. $d_i = d(x_{i-1})$ is distance from x_{i-1} to Γ
  2. $\{\omega_i\}$ is sequence of independent unit isotropic vectors
  3. x_i is the exit point from the ball, $B(x_{i-1}, d(x_{i-1}))$, for a Brownian motion starting at x_{i-1}
- Outside the molecule, on every step, walk-on-spheres terminates with probability $1 - q(\kappa, d_i)$, where $q(\kappa, d_i) = \dfrac{\kappa d_i}{\sinh(\kappa d_i)}$ to deal with LPBE

'Walk-on-Spheres' and 'Walk-in-Subdomains'

- For general domains, an efficient way to simulate exit points is a combination of
  1. Inside the molecule: 'walk-in-subdomains'
  2. Outside the molecule 'walk-on-spheres'
- The whole domain, G_i, is represented as a union of intersecting subdomains:

$$G_i = \bigcup_{m=1}^{M} G^m$$

- 'Walk-in-Subdomains': Simulate exit point separately in every G^m

  1. $x_0 = x, x_1, \ldots, x_N$ – Markov chain, every x_{i+1} is an exit point from the corresponding subdomain for Brownian motion starting at x_i
  2. For spherical subdomains, $B(x_i^m, R_i^m)$, exit points are distributed in accordance with the Poisson kernel:

$$\frac{1}{4\pi R_i^m} \frac{|x_i - x_i^m|^2 - (R_i^m)^2}{|x_i - x_{i+1}|^3}$$

Monte Carlo Estimates

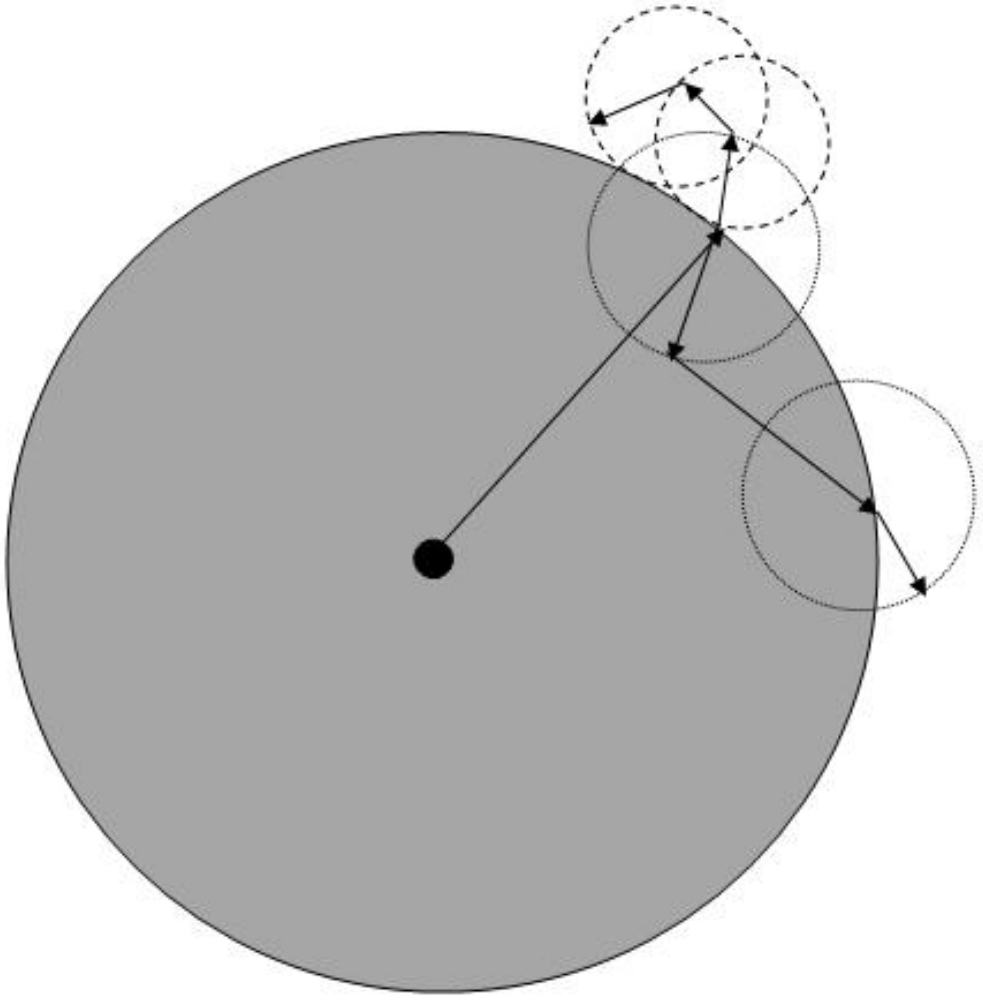
- The estimate for the reaction-field potential point value:

$$\begin{aligned} \xi[\Phi_{rf}](x^{(m)}) &= -\Phi^c(x_1^*) \\ &+ \sum_{j=2}^{N_{ins}} F_j(\kappa)\,(\Phi^c(x_j^{ins}) - \Phi^c(x_{j,ins}^*)) \end{aligned} \tag{5}$$

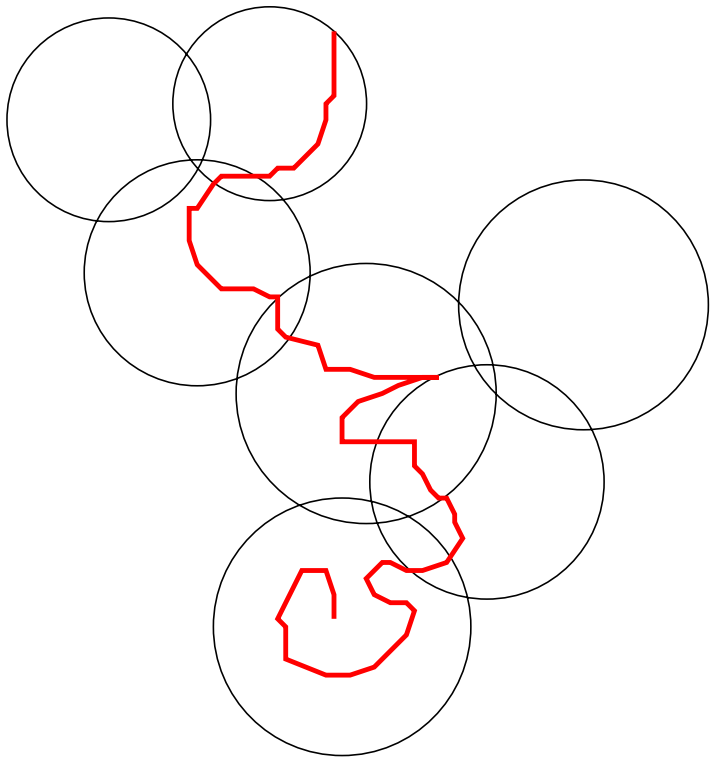
- Here $\{x_{j,ins}^*\}$ is a sequence of boundary points, after which the random walker moves inside the domain, G_i, to x_j^{ins}
- The estimate for the reaction-field energy:

$$\xi[W_{rf}] = \frac{1}{2}\sum_{m=1}^{M} Q_m\,\xi[\Phi_{rf}](x^{(m)}) \tag{6}$$

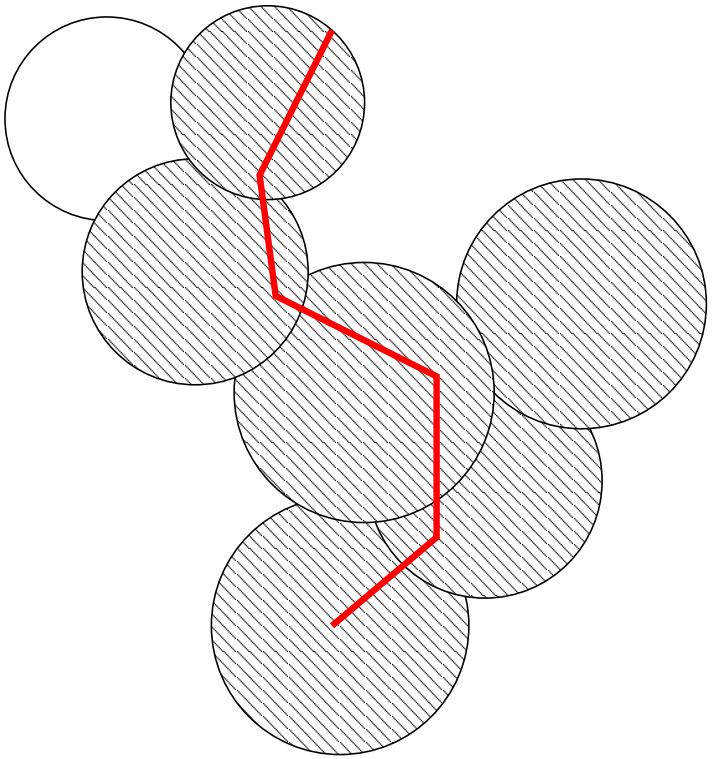
A Picture: The Algorithm for a Single Spherical Atom



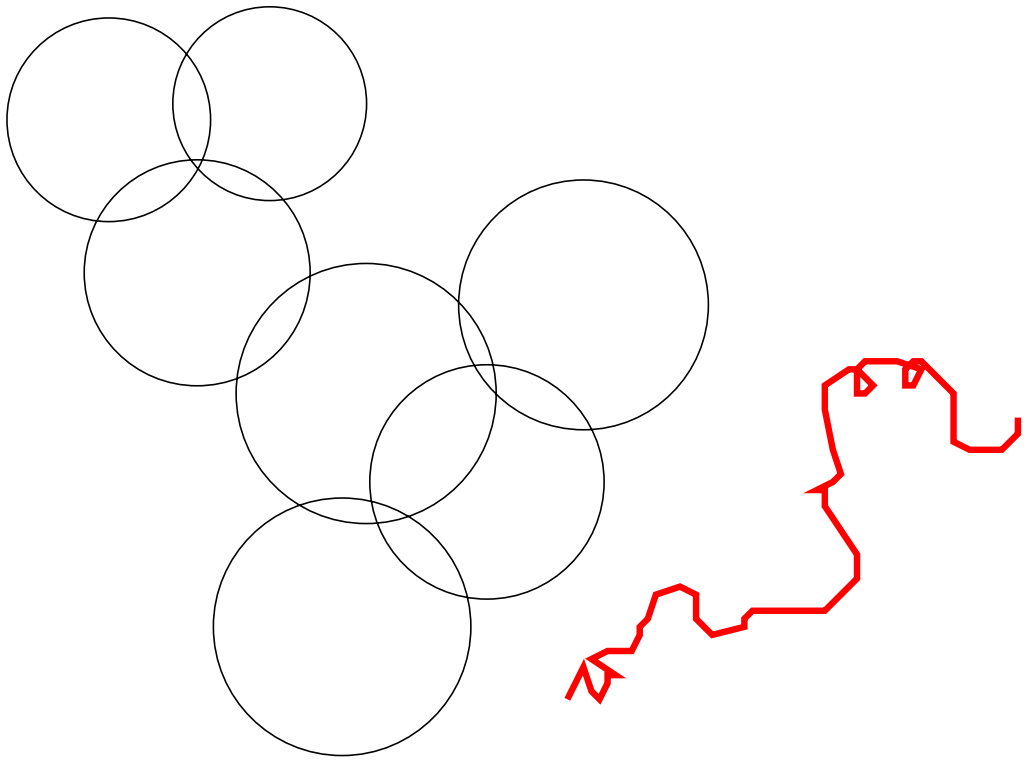
The Algorithm in Pictures: Walk Inside



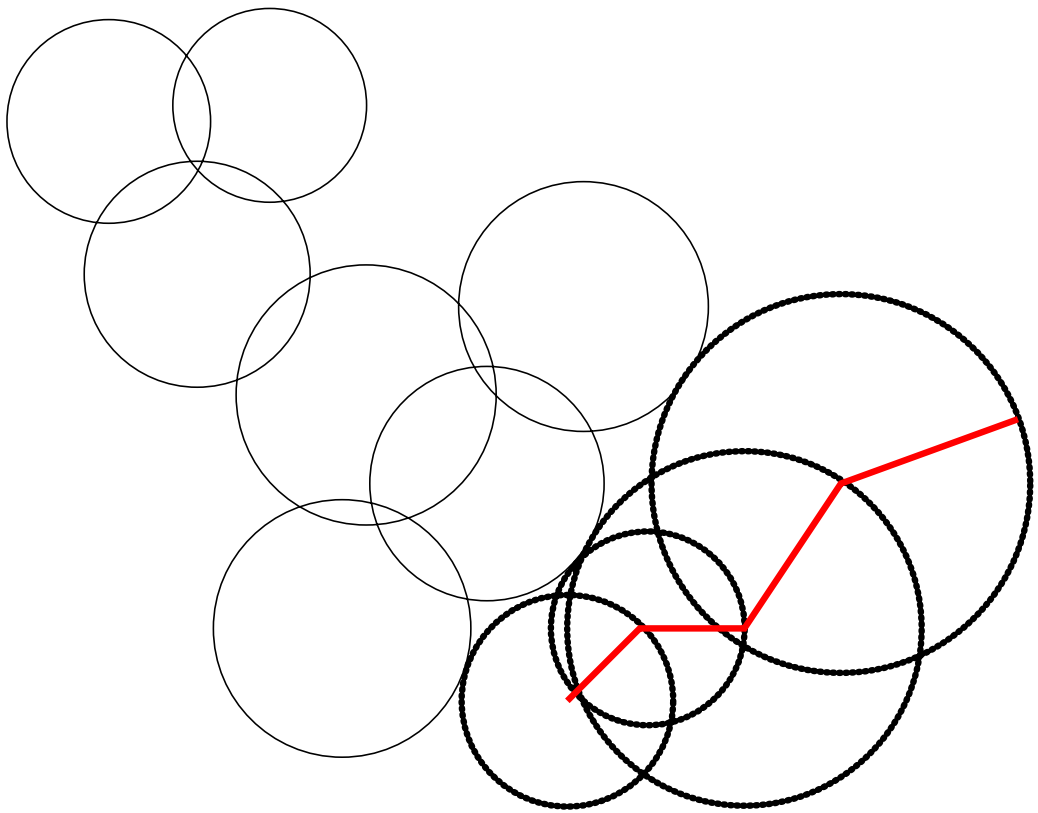
The Algorithm in Pictures: Walk Inside



The Algorithm in Pictures: Walk Outside



The Algorithm in Pictures: Walk Outside



Monte Carlo Algorithm's Computational Complexity

Cost of a single trajectory

- Number of steps is random walk is not dependent on M, the number of atoms
- The cost of finding the nearest sphere is $M \log_2(M)$ due to optimizations

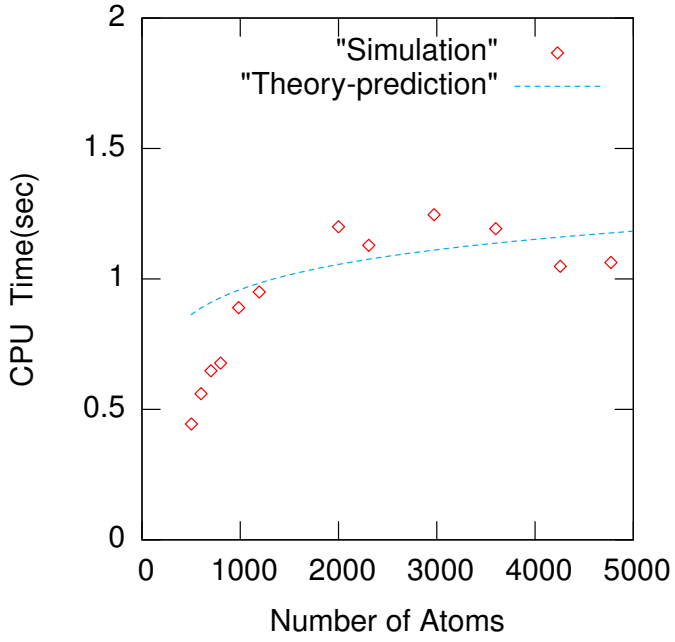
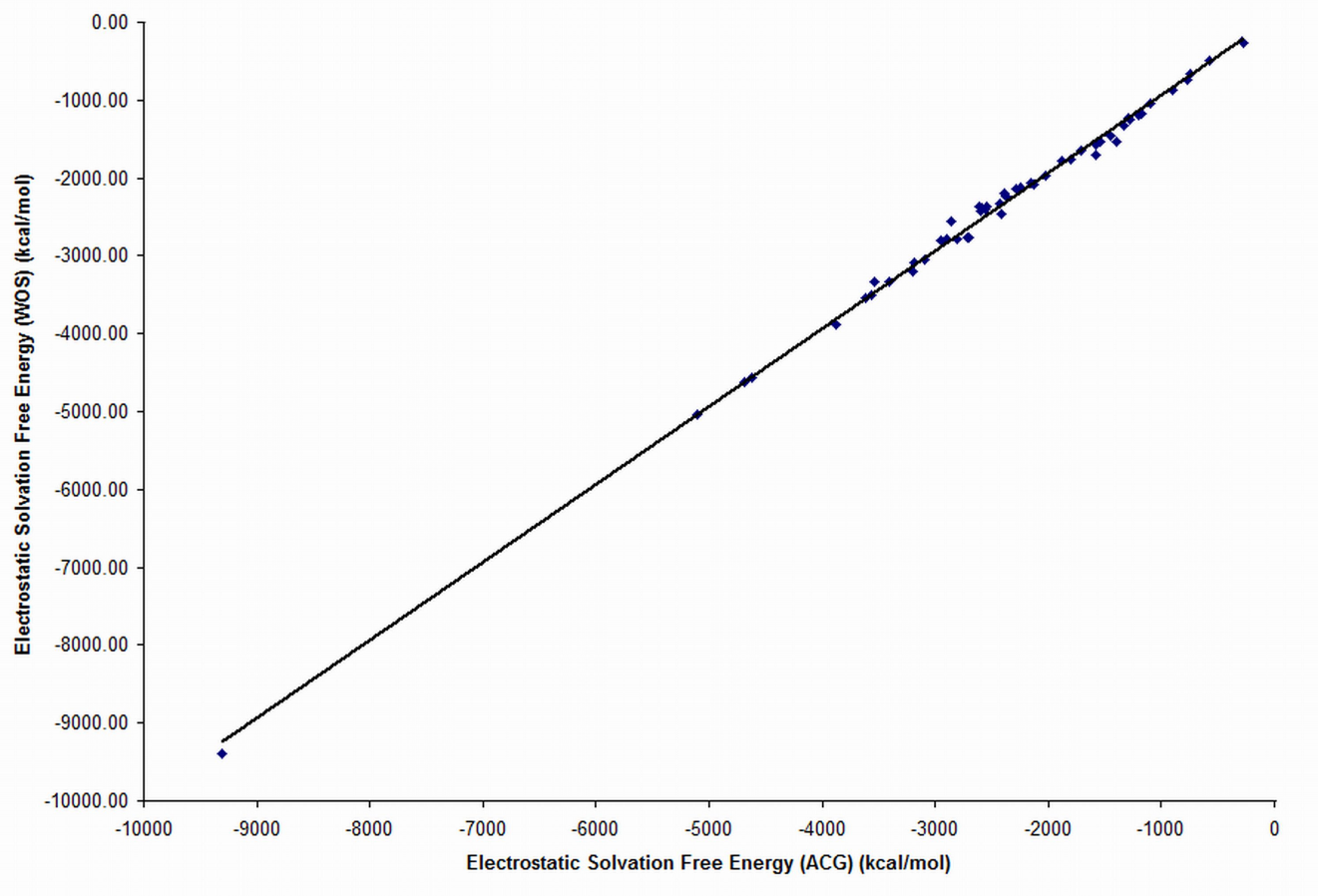


Figure: The CPU time per atom per trajectory is plotted as function of number of atoms. For small number of atoms the CPU time scales linearly and for large number of atoms it asymptotically scales logarithmically

Accuracy: Monte Carlo vs. Deterministic

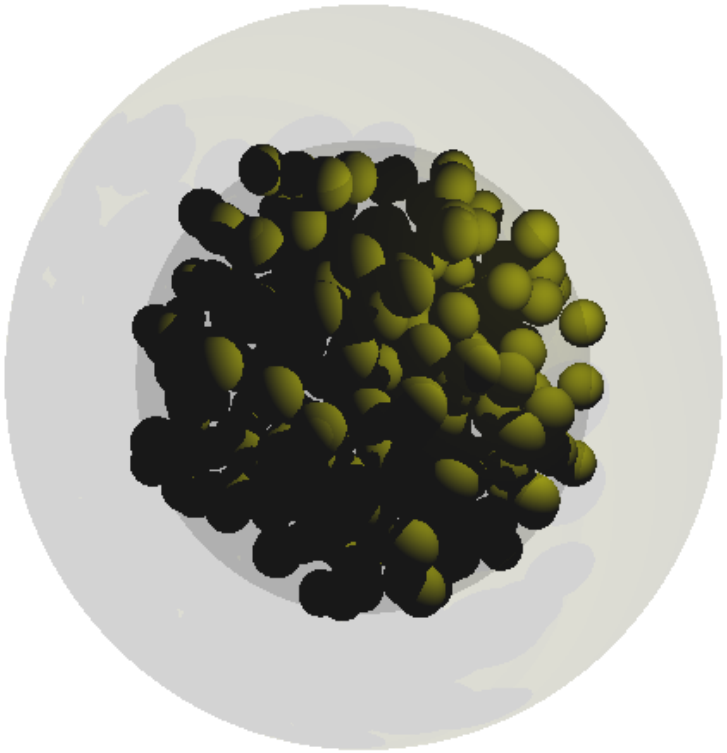


Geometry: Problem Descriptions

There are many geometric problems that arise in this algorithm:

- Efficiently determining if a point is on the surface of the molecule or inside of it (for interior walks)
- Efficiently determining the closest sphere to a given exterior point (for walks outside molecule)
- Efficiently determining if a query point is inside of the convex hull of the molecule
- Efficiently finding the largest possible sphere enclosing a query point for external walks

Porous Media: Complicated Interfaces



Computing Capacitance Probabilistically

- Hubbard-Douglas: can compute permeability of nonskew object via capacitance
- Recall that $C = \frac{Q}{u}$, if we hold the conductor, Ω, at unit potential $u = 1$, then $C =$ total charge the conductor's surface, $\partial\Omega$
- The PDE system for the potential is

$$\Delta u = 0, \quad x \notin \Omega; \qquad u = 1, \quad x \in \partial\Omega; \qquad u \to 0 \ \text{as} \ |x| \ \to \infty$$

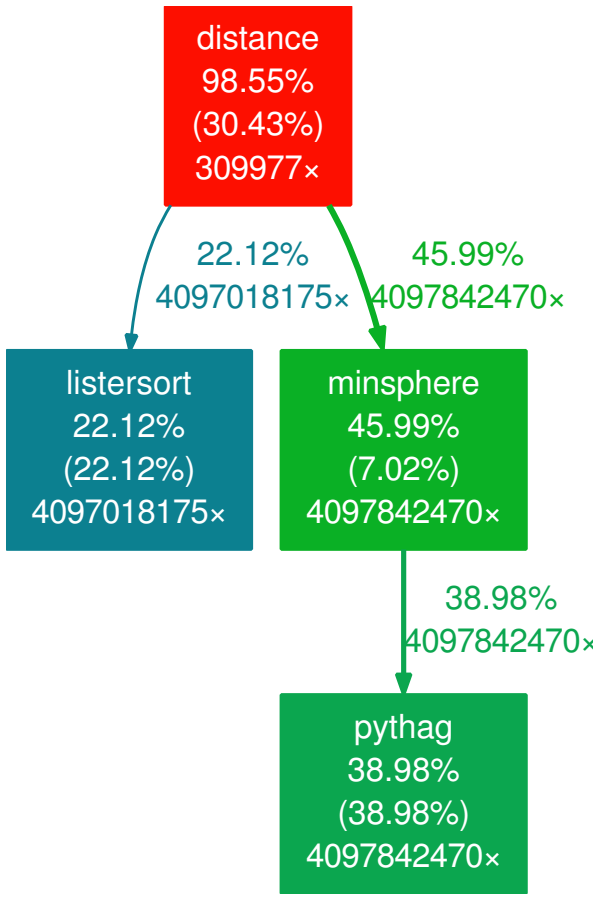
- Recall $u(\mathbf{x}) = \mathbb{E}_x[f(X^{\mathbf{x}}(t_{\partial\Omega}))] = 1$, if the walker hits $\partial\Omega$, or 0, if the walker $\to \infty =$ probability of walker starting at $\mathbf{x}$ hitting Ω before escaping to infinity
- Charge density is first passage measure on the exterior of Ω
- Construct a sphere, $S(R)$, such that $\Omega \subset S(R)$
- Capacitance is $C = R \times \mathbb{P}(t_{\partial\Omega} < \infty) = R \times \int_{\partial S(R)} u(\mathbf{x}) ds$, where the starting point of the walk is chosen uniformly on the $\partial S(R)$
- Note, this definition is independent of R

Capacitance as a Computational Paradigm

- We have found that the running time of these codes hinged on the efficiency of geometrical computations
- The probabilistic capacitance computation is key in
  1. Permeability computation via Hubbard-Douglas
  2. Many other electrostatics computations in areas of chemistry, computer science, physics, and materials science, including our own PBE computations
  3. In the ZENO code written for materials property computations at NIST
- We took the NIST ZENO code and analyzed it to determine areas for improvement

The NIST ZENO Code

- ZENO uses Monte Carlo sampling to compute a variety of materials properties
- ZENO is a FORTRAN 77 code that had a very suggestive CPU profile



What is the Computational Geometric Problem?

- ZENO is named for the paradox, and getting close the boundary for producing approximate first-passage locations is done with the ZENO algorithm, which is WOS by another name
- The usual geometry in these problems is additive
- As mentioned above the primitive computation geometrically is given a query point, find the closest point to the boundary, and that is used as the radius to construct the WOS radius
- The usual ZENO problem is the so-called exterior problem, which is what is done in permeability: computing the probability of first-passage from ∞ using relative capacitance of a bounding sphere

Different Approaches to the Computational Geometry

- ZENO used a linear pass thought the additive list of subcomponents of the geometry, far from optimal
- This is a problem that computational geometers have studied, mostly in very high dimensions
- Here is a comparison of some low-dimensional techniques that use a hierarchical k-D tree decomposition of the additive components

Type	**Time (s)**	**Speedup**
Linear List	3,993	
ANN (*k*-D Tree)	82.19	48.58x
NanoFLANN (*k*-D)	41.97	95.28x
CGAL (*k*-D Tree)	63.20	63.18x

Table: Zeno C++, 1M paths, Intel® Xeon® CPU E5-2630 v2 @ 2.60 GHz

Parallel Scalability using MPI

Nodes	Processes per node	Time (s)	Speedup
1	1	41.91	–
1	12	3.383	12.39x
12	1	3.699	11.33x
12	12	0.5068	82.70x

Table: Zeno C++, 1M paths, Intel® Xeon® CPU E5-2630 v2 @ 2.60 GHz, 12 physical & 24 logical cores

Nodes	Processes per node	Time (s)	Speedup
1	1	416.8	–
1	12	31.45	13.25x
12	1	35.16	11.86x
12	12	2.922	142.6x

Table: Zeno C++, 10M paths, Intel® Xeon® CPU E5-2630 v2 @ 2.60 GHz, 12 physical & 24 logical cores

Single Node Scalability

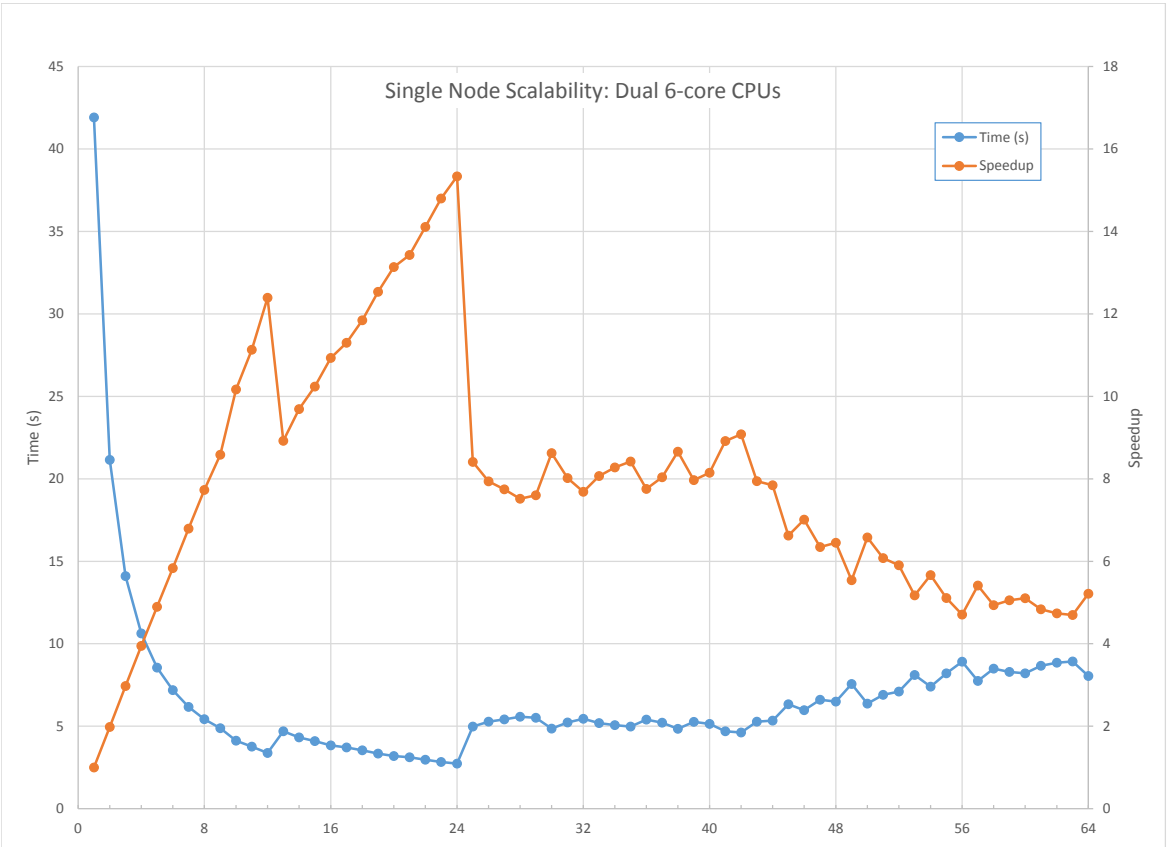


Figure: Single Node Scalability

Almost Linear Scalability

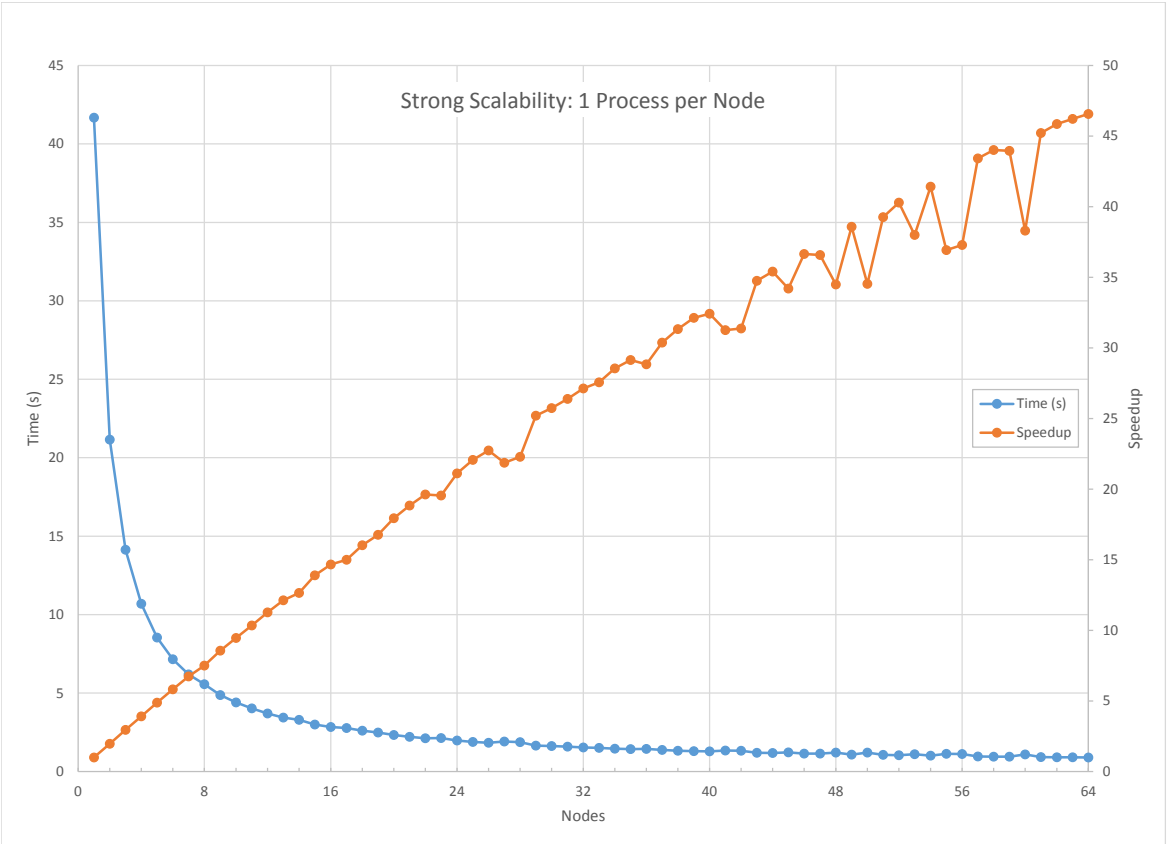


Figure: Scalability Many Nodes

Conclusions

- Monte Carlo can be an efficient method for the numerical solution of PDEs as evidenced by
  1. Financial computing
  2. Numerous problems in electrostatics
  3. Problems in computational materials
- The computation of capacitance is a good model problem for computational investigation
  1. Computational geometry is the bottleneck that can be overcome with proper choice of data-structure/algorithm
  2. Allows for almost perfect parallelization, across cores and multicore nodes
- C++ version of ZENO uses `SPRNG`

Future Work

- We expect to be able to improve all the codes we have developed
  1. New C++ version of ZENO
  2. Current C++ version of the WOS-PBE code
- We want to have a WOS-based implementation of the capacitance code using CUDA on NVidia GPUs
  1. Must use the analogous implementation that was developed for vectorizing neutron transport
  2. Will take advantage of the GPU version of `SPRNG`
- Plan to move to nonlinear PB solver using branching processes
- Two orders of magnitude improvement with almost perfect parallel scaling will permit the rapid solution of very large problems

Bibliography


[T. Mackoy, R. C. Harris, J. Johnson, M. Mascagni and M. O. Fenley (2011)] Numerical Optimization of a Walk-on-Spheres Solver for the Linear Poisson-Boltzmann Equation *Communications in Computational Physics*, in the press.


[A. Rasulov, G. Raimova and M. Mascagni (2010)] Monte Carlo Solution of Some Nonlinear Parabolic Initial-Value Problems *Mathematics and Computers in Simulation*, **80(6)**: 1118–1123.


[M. Fenley, M. Mascagni, J. McClain, A. Silalahi and N. Simonov (2010)] Using Correlated Monte Carlo Sampling for Efficiently Solving the Linearized Poisson-Boltzmann Equation Over a Broad Range of Salt Concentrations *Journal of Chemical Theory and Computation*, **6(1)**: 300–314.

Bibliography

- [N. Simonov and M. Mascagni and M. O. Fenley (2007)] Monte Carlo Based Linear Poisson-Boltzmann Approach Makes Accurate Salt-Dependent Solvation Energy Predictions Possible *Journal of Chemical Physics*, **187(18)**, article #185105, 6 pages.
- [M. Mascagni and N. A. Simonov (2004)] Monte Carlo Methods for Calculating Some Physical Properties of Large Molecules *SIAM Journal on Scientific Computing*, **26(1)**: 339–357.
- [N. A. Simonov and M. Mascagni (2004)] Random Walk Algorithms for Estimating Effective Properties of Digitized Porous Media *Monte Carlo Methods and Applications*, **10**: 599–608.
- [M. Mascagni and N. A. Simonov (2004)] The Random Walk on the Boundary Method for Calculating Capacitance *Journal of Computational Physics*, **195**: 465–473.

Bibliography

- [C.-O. Hwang and M. Mascagni (2003)] Analysis and Comparison of Green's Function First-Passage Algorithms with "Walk on Spheres" Algorithms *Mathematics and Computers in Simulation*, **63**: 605–613.
- [C.-O. Hwang and M. Mascagni (2001)] Efficient modified "walk on spheres" algortihm for the linearized Poisson-Boltzmann equation *Applied Physics Letters*, **78**: 787–789.
- [C.-O. Hwang, J. A. Given and M. Mascagni (2001)] The Simulation-Tabulation Method for Classical Diffusion Monte Carlo *Journal of Computational Physics*, **174**: 925–946.
- [C.-O. Hwang, J. A. Given and M. Mascagni (2000)] On the Rapid Calculation of Permeability for Porous Media Using Brownian Motion Paths *Physics of Fluids*, **12**: 1699–1709.

Copyright Notice