



A Grid-Computing Infrastructure for Monte Carlo Applications

Michael Mascagni (mascagni@fsu.edu)

*Department of Computer Science and
School of Computational Science*

Florida State University, Tallahassee, FL USA

and Yaohang Li (yaohang@ncat.edu)

*Department of Computer Science
North Carolina A&T University*

Greensboro, NC USA





Outline



- Monte Carlo Methods
 - Convergence Acceleration
- Grid-based Monte Carlo Applications
 - *N-out-of-M* Scheduling Strategy
 - Lightweight Checkpointing
 - Partial Result Validation
 - Intermediate Value Checking
- Grid-computing Infrastructure for Monte Carlo Applications
- An Example Application
 - Grid-based Molecular Dynamics/Brownian Dynamics Simulation
- Conclusions



Monte Carlo Methods



- Monte Carlo Methods
 - Stochastic solutions to a variety of problems
 - Performing statistical sampling experiments
 - Based on random sample estimation
 - Slow convergence rate, approximately $O(N^{-1/2})$
 - Acceleration of Monte Carlo methods
 - Variance Reduction
 - Quasi-Monte Carlo
 - Parallel Computing
 - Applications are ubiquitous



Variance Reduction Techniques

To evaluate $\theta = \int_0^1 f(x) dx$

- Stratified Sampling

$$\theta' = \sum_{j=1}^k \sum_{i=1}^{n_j} (\alpha_j - \alpha_{j-1}) \frac{1}{n_j} f(\alpha_{j-1} + (\alpha_j - \alpha_{j-1}) \xi_{ij})$$

- Control Variates

$$\theta = \int_0^1 \phi(x) dx + \int_0^1 [f(x) - \phi(x)] dx$$

- Importance Sampling

$$\theta = \int_0^1 \frac{f(x)}{g(x)} g(x) dx = \int_0^1 \frac{f(x)}{g(x)} dG(x)$$

$$G(x) = \int_0^x g(y) dy, \text{ and } G(1) = 1$$

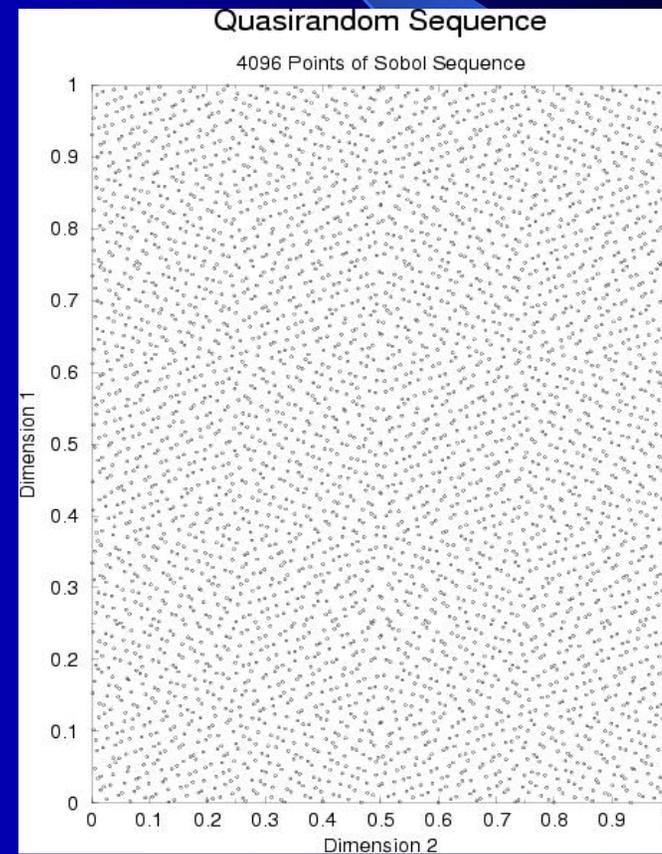
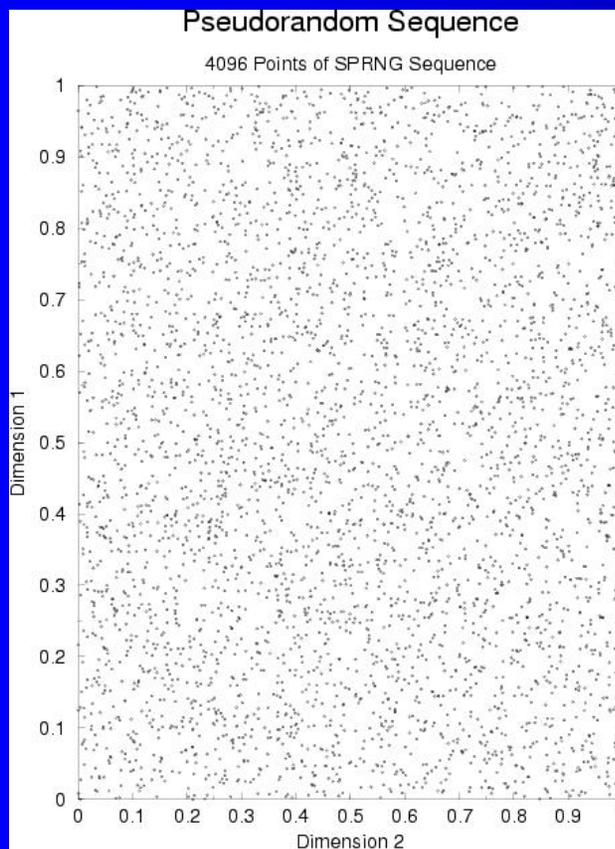
- Antithetic Variates

$$\theta' = \frac{1}{2} f(\xi) + \frac{1}{2} f(1 - \xi)$$



Quasi-Monte Carlo (I)

- Quasirandom Numbers vs. Pseudorandom Numbers
 - Uniformity (low discrepancy)
 - “A picture is worth a thousand words”

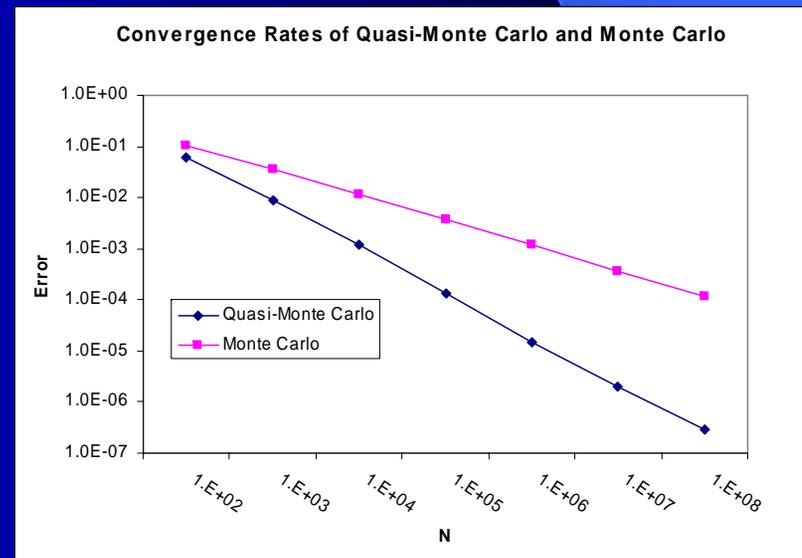
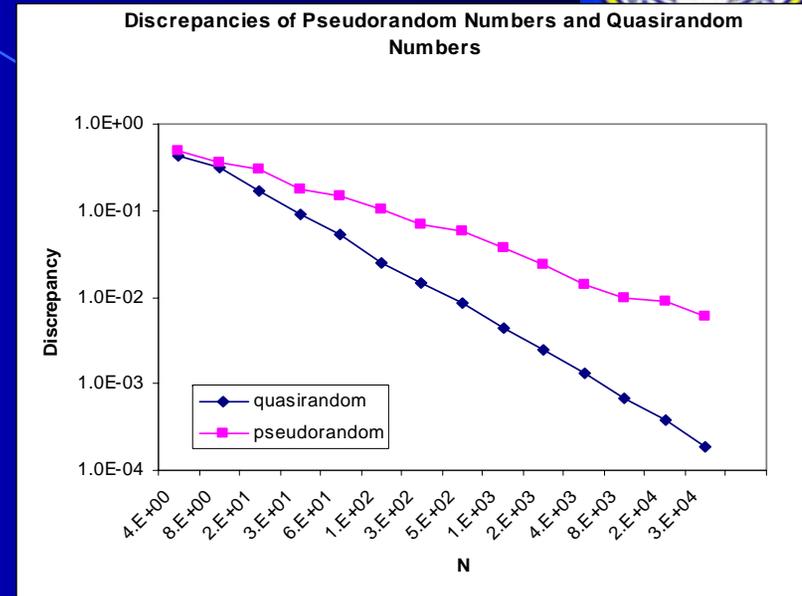




Quasi-Monte Carlo (II)



- Quasi-Monte Carlo Method
 - Convergence rate close to $O(N^{-1})$
 - the Koksma-Hlawka Inequality
 - Limitations
 - Integration and the K-H inequality
 - Smoothness of the integrand
 - Convergence rate is related to dimension, s
- Quasirandom Number Sequences
 - Van der Corput
 - Halton
 - Faure
 - Sobol'
 - Niederreiter





Parallel Monte Carlo Applications



- Parallelism in Monte Carlo Applications
 - Computationally intensive but naturally parallel
 - Appropriate for dynamic *bag-of-work* paradigm
 - Requirements
 - Independence of underlying random number streams
 - SPRNG (Scalable Parallel Random Number Generation) library
 - Large-scale computational resources
 - Fits the distributed computing paradigm



Large-Scale Monte Carlo Applications in a Puzzle



Development of Monte Carlo Methods

Parallel Random Number Generators

Large-Scale Computational Resources

Applications in Science and Engineering

Our Goal: To Develop A High Performance and Trustworthy Large-scale Monte Carlo Computing Infrastructure



Grid-based Monte Carlo Applications



- Grid Computing
 - Large-scale resources cooperation and sharing
- Issues in Grid Computing
 - From application point-of-view
 - Performance
 - Trustworthiness (especially from volunteers)
- Our Approach
 - Address issues from the application level
 - Analyze characteristics of Monte Carlo applications
 - Statistical nature
 - Cryptographic aspects of underlying random number generator
 - Develop strategies and tools



Techniques for Grid-based Monte Carlo Applications

- Improve Performance
 - *N-out-of-M* Strategy
 - Lightweight Checkpointing
- Improve Trustworthiness
 - Partial Result Validation Scheme
 - Intermediate Value Checking



N-out-of-M Strategy



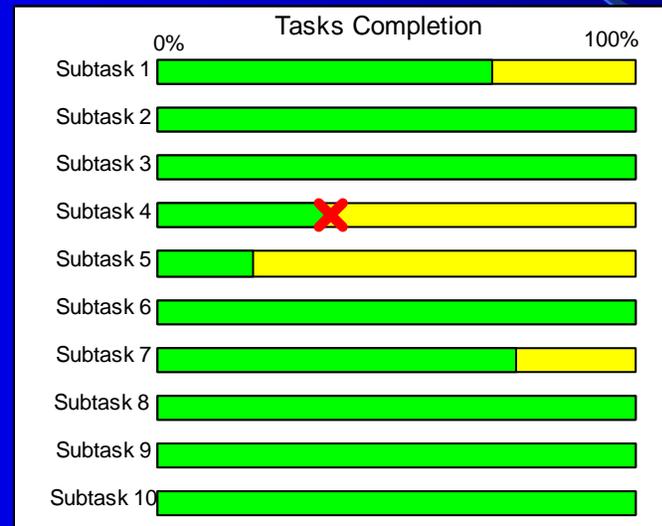
- Reality of Grid Computing
 - Service providers' performance varies
 - Resources are widely distributed
- Problems
 - A slow node might become the bottleneck of the whole computation
 - A halted subtask may prevent the whole task from completing
- Monte Carlo Applications
 - Care: How many random samples are estimated?
 - *Don't Care*: Which random sample set is used in the estimate?
- *N-out-of-M* Strategy for Subtask Schedule
 - Subtasks
 - Same description of the problem
 - Different independent random streams
 - Enlarge number of subtasks
 - N subtasks $\rightarrow M$ subtasks, $M > N$
 - Gather N subtasks
 - N partial results ready \rightarrow Whole computation is done



Example of *N-out-of-M* Strategy



- Example



Example of *6-out-of-10* Scheduling

- Benefits of *N-out-of-M* Strategy for Job Scheduling
 - None of the subtasks is a “key” subtask
 - Can tolerate at most $M - N$ delayed or halted subtasks
 - Computation is still reproducible

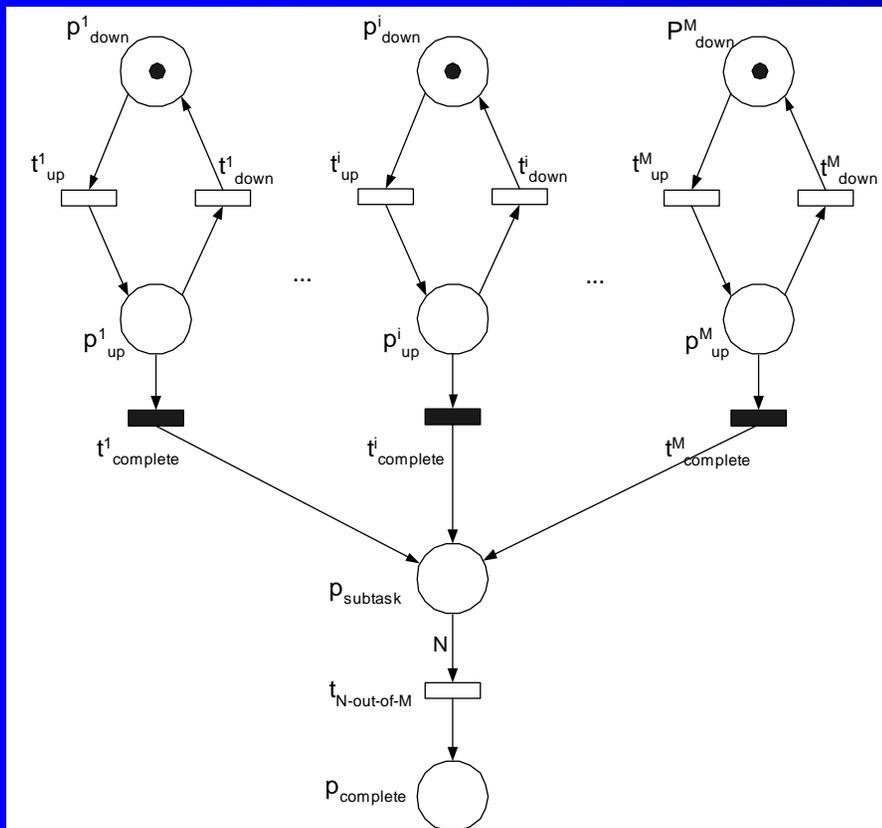


Modeling the *N-out-of-M* Strategy



- Petri Net Model of the *N-out-of-M* Strategy

- Binomial Probability Model of the *N-out-of-M* Strategy



$$P_{Exactly-N-out-of-M}(t) = \binom{M}{N} p^N(t) \times (1 - p(t))^{M-N}$$

$$P_{N-out-of-M}(t) = \sum_{i=N}^M \binom{M}{i} p^i(t) \times (1 - p(t))^{M-i}$$

- Model of *N-out-of-N*

$$P_{N-out-of-N}(t) = p^N(t)$$

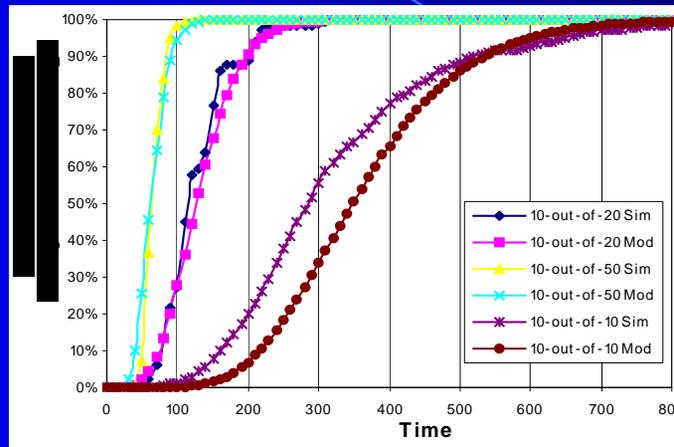
$p(t)$ is the probability of a subtask completing before time t



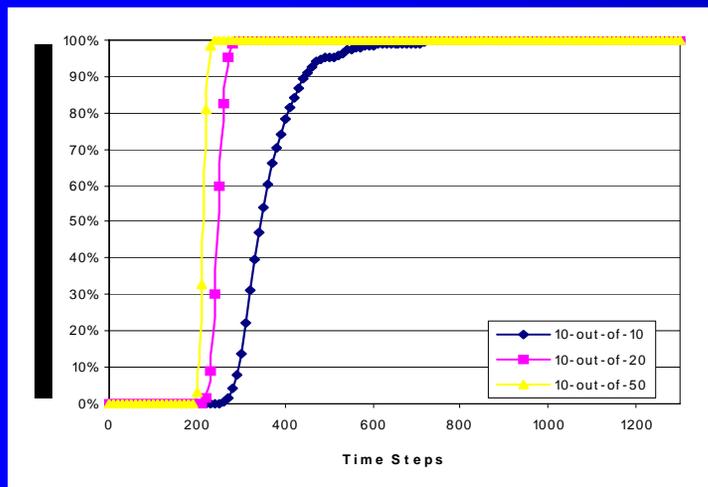
Simulation of *N-out-of-M* Strategy



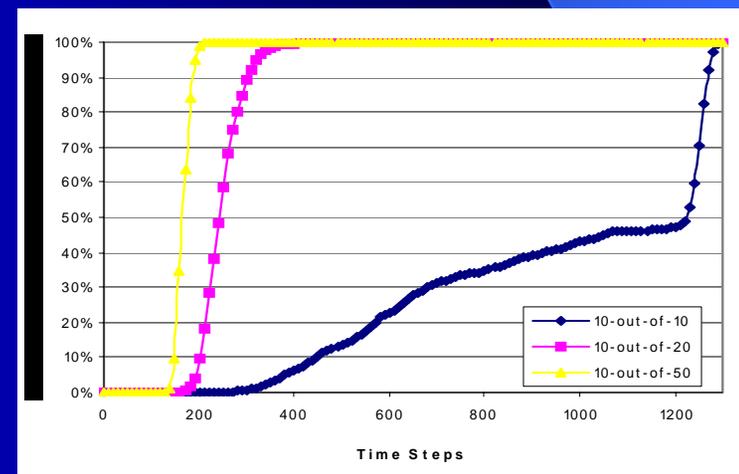
Model Validation by Simulation



Simulations of the *N-out-of-M* strategy on a grid with node service rates normally distributed



Mean=0.005, Variance=0.001

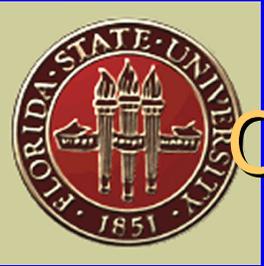


Mean=0.005, Variance=0.003



Lightweight Checkpointing

- Process-level Checkpointing
 - A snapshot of a process' current state
 - Costly
 - Platform-dependent
- Application-level Checkpointing for Monte Carlo Applications
 - Amenable to application-based checkpointing
 - A small amount of information
 - Status and parameters of RNG
 - Iteration number
 - Other intermediate results

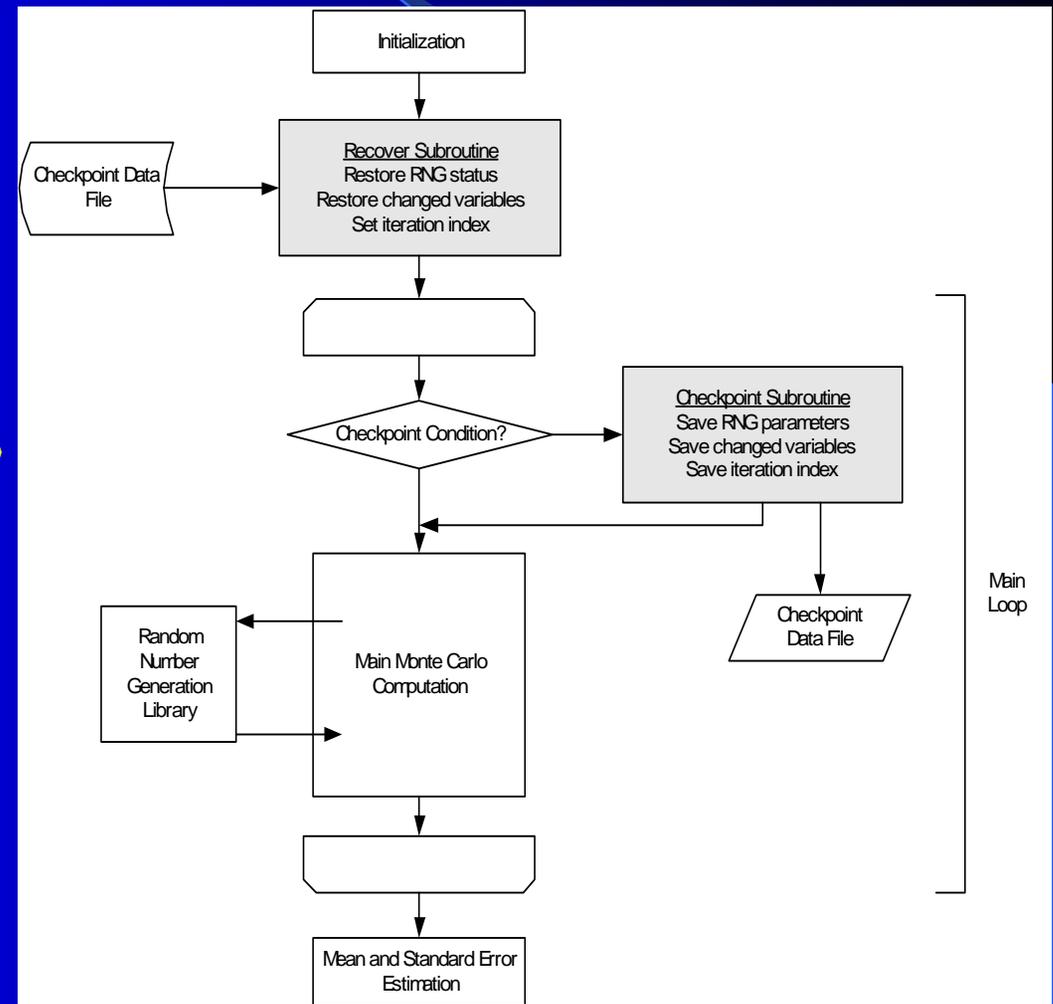
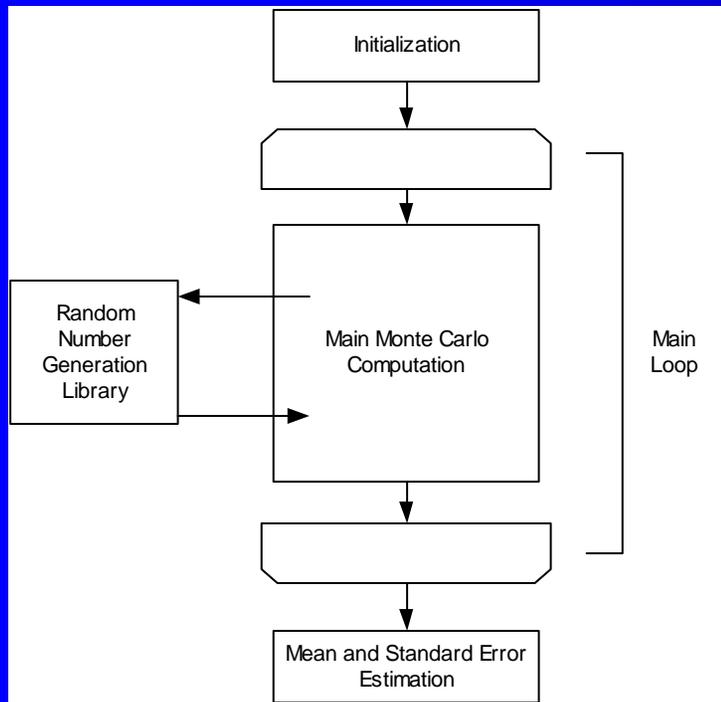


Implementation of Lightweight Checkpointing for Monte Carlo Applications



Monte Carlo Application with Checkpointing Facilities

Typical Monte Carlo Programming Logic





Distributed Monte Carlo Partial Result Validation



- Existing Problem in Grid Computing
 - “Problematic” Node
 - Not trustworthy/reliable
 - Not faithfully executing the code
 - Faking computation
- Our Approach
 - Statistical nature of the partial results from Monte Carlo applications
 - Examine partial results according to the expected distribution
- Distributed Monte Carlo Subtasks
 - Same program
 - Different independent random streams



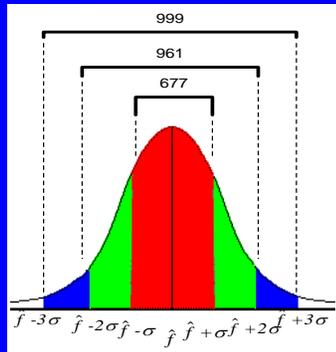
Partial Result Validation Procedure

- Central Limit Theorem
 - The mean of partial results

$$\hat{f} = \frac{1}{n} \sum_{i=1}^n f_i$$

- Standard Error

$$s = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (f_i - \hat{f})^2}$$



- 68% confidence in 1 standard deviation
- 95% confidence in 2 standard deviation
- 99% confidence in 3 standard deviation

$f_1, \dots, f_p, \dots, f_n$ are the n partial results

- n Subtasks

- For every partial result f_i
- Compute Normal Confidence Interval

- With confidence $\alpha\% = 1 - 1/cn$

$$[\hat{f} - Z_{\alpha/2}\delta, \hat{f} + Z_{\alpha/2}\delta], \text{ where } \int_0^{Z_{\alpha/2}} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx = \frac{\alpha}{2}$$

- c : 1 in c runs of size n is expected to be outlier
- In confidence interval \rightarrow trustworthy
- Outside confidence interval \rightarrow suspect
 - Rerun the subtask
 - Discard



Example of Monte Carlo Partial Result Validation



- Evaluate Integral

$$\int_0^1 \dots \int_0^1 \frac{4 x_1 x_3^2 e^{2 x_1 x_3}}{(1 + x_2 + x_4)^2} e^{x_5 + \dots + x_{20}} x_{21} x_{22} \dots x_{25} dx_1 \dots dx_{25}$$

– Exact solution in 8-digit

- 103.81372

– With 1,000 subtasks

- Hypothetical Partial Results

Subtask #	Partial Results
1	103.8999347
2	104.0002782
3	103.7795764
4	103.6894540
...	
561	89782.048998
...	
997	103.9235347
998	103.8727823
999	103.8557640
1000	103.7891408

Partial result of #561 will be expected to happen once in 10⁹ experiments

Suspected

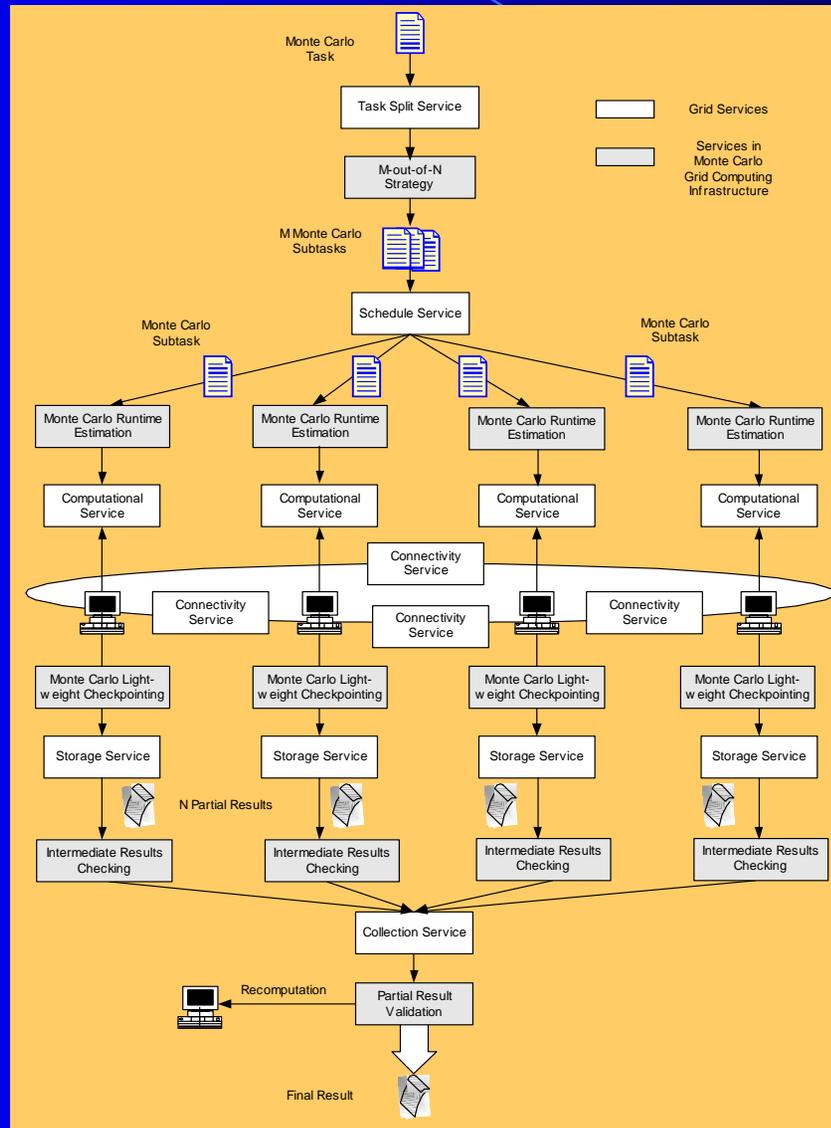
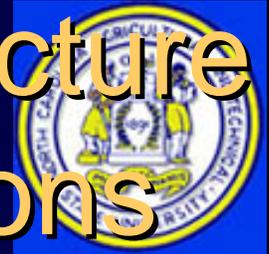


Intermediate Value Checking

- Goal
 - Detect if a subtask is faithfully executed
 - Detect bogus results
- Intermediate Value Candidates
 - To the node
 - Unknown until reach a specific point of the program
 - To the program owner
 - Pre-known or easy generated
- Pseudorandom Numbers
 - Consumed in Monte Carlo applications
 - Deterministic
 - Unknown to a Computational Service Provider until the number is generated
 - Cheap to calculate during validation depending on the generator
- Our Approach
 - Store certain pseudorandom numbers during the execution of a subtask and compare with those computed on server



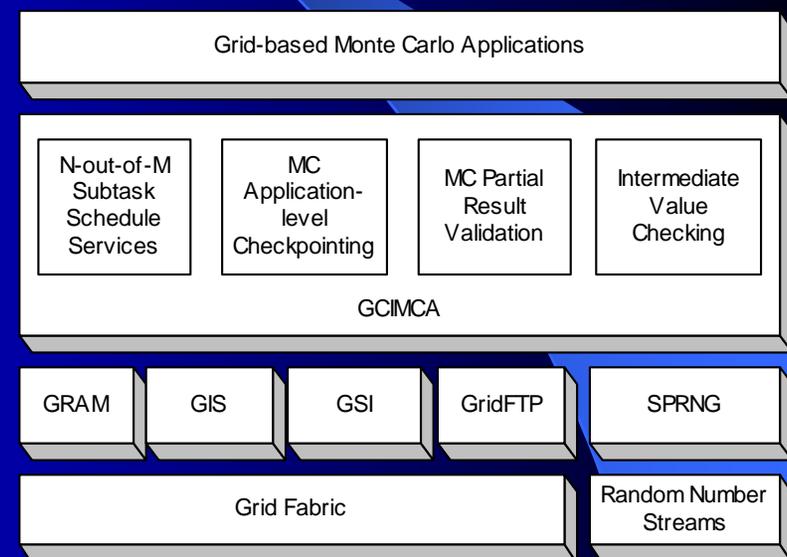
A Grid-Computing Infrastructure for Monte Carlo Applications





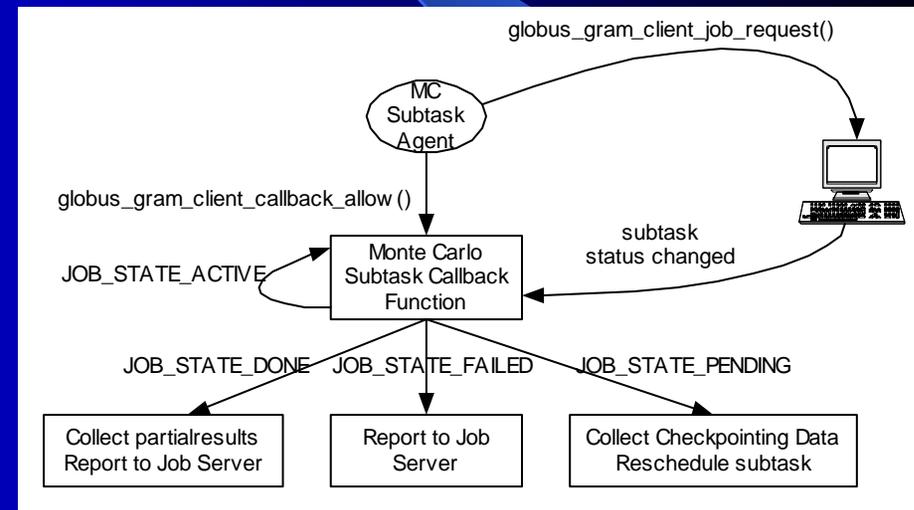
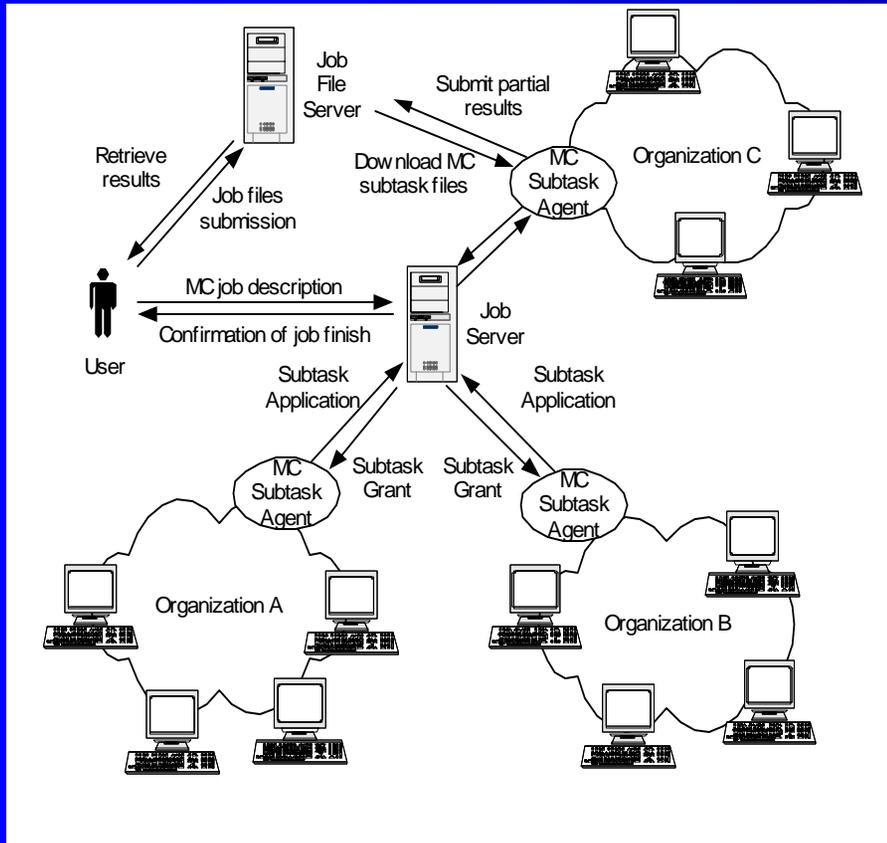
GCIMCA: A Globus and SPRNG Implementation of A Grid-Computing Infrastructure for Monte Carlo Applications

- Globus
 - Facilities to Create and Utilize a Computational Grid
- SPRNG (Scalable Parallel Random Number Generators) Library
 - Large Scale Parallel Random Number Generation





Working Paradigm for GCIMCA



Remote Execution of a Monte Carlo Subtask

Working Paradigm



A Monte Carlo Application: MD/BD Simulation of Ligand-Receptor Interaction in Structured Protein System



- Hybrid Molecular Dynamics (MD)/Brownian Dynamics (BD)

- MD

- Analysis of the force autocorrelation function
- Determine the friction tensor

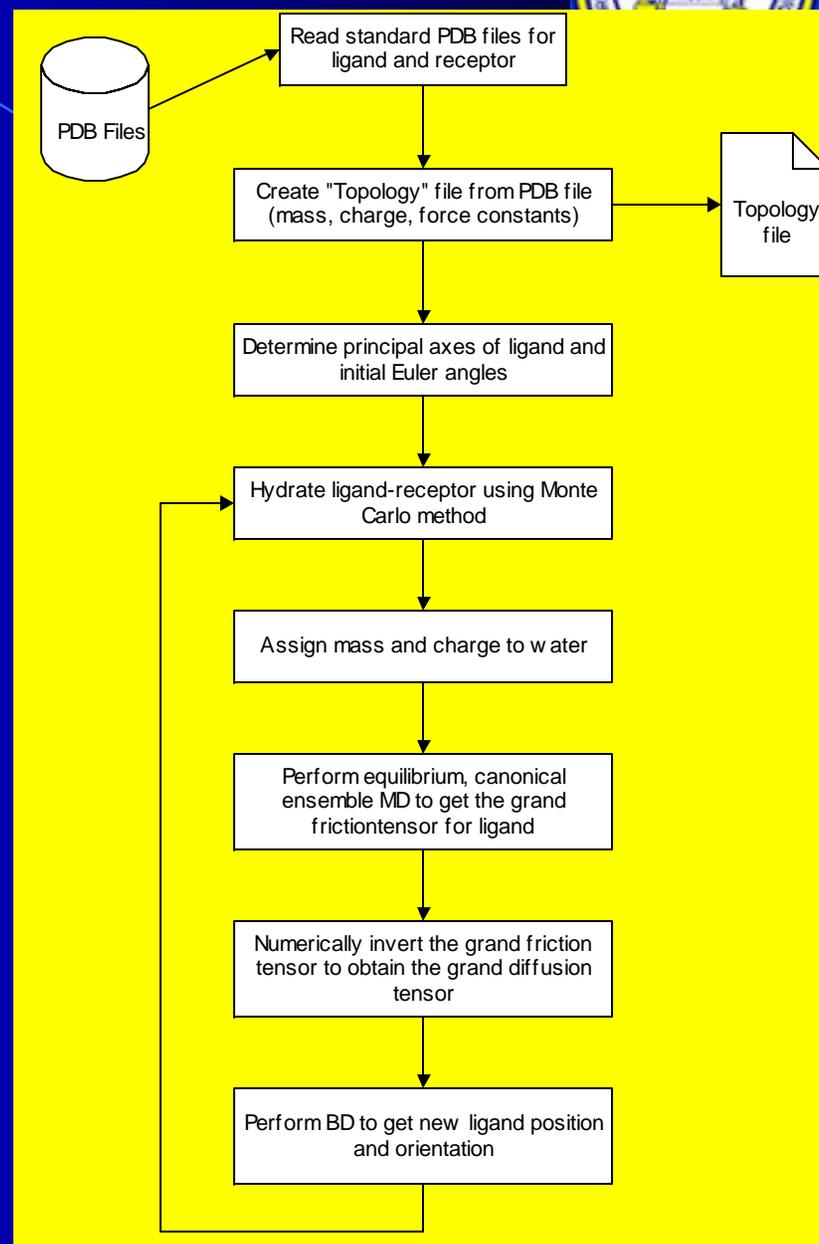
- BD

- The Fokker-Planck equation is solved for discrete times
 - Assuming the friction tensor remains constant at this time step

- Computation Analysis

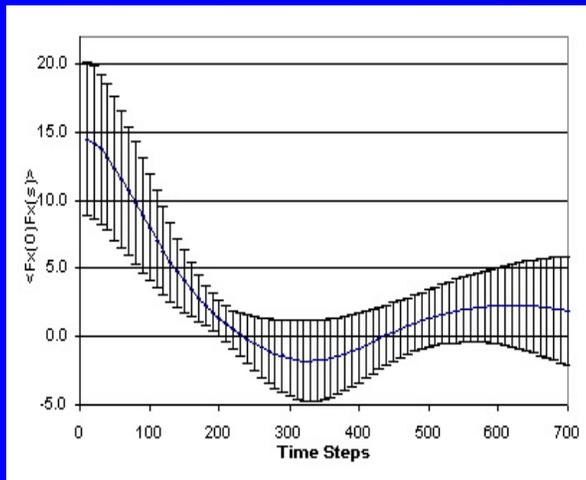
- Autocorrelation Calculation

- Computationally costly
- Amenable to multiprocessor systems

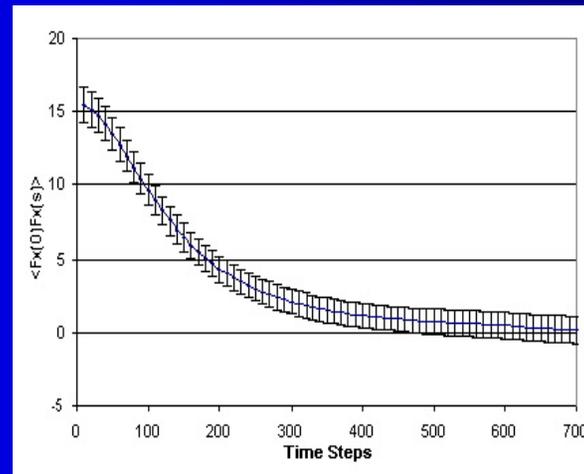




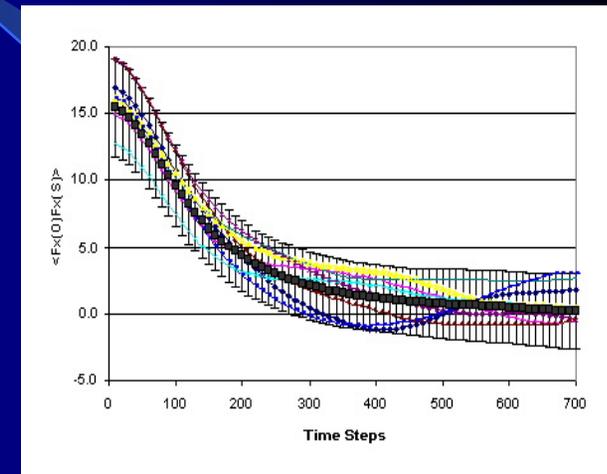
Preliminary Results of Grid-based MD/BD Simulation



40 ensembles
on a parallel system



400 ensembles
on a computational
grid



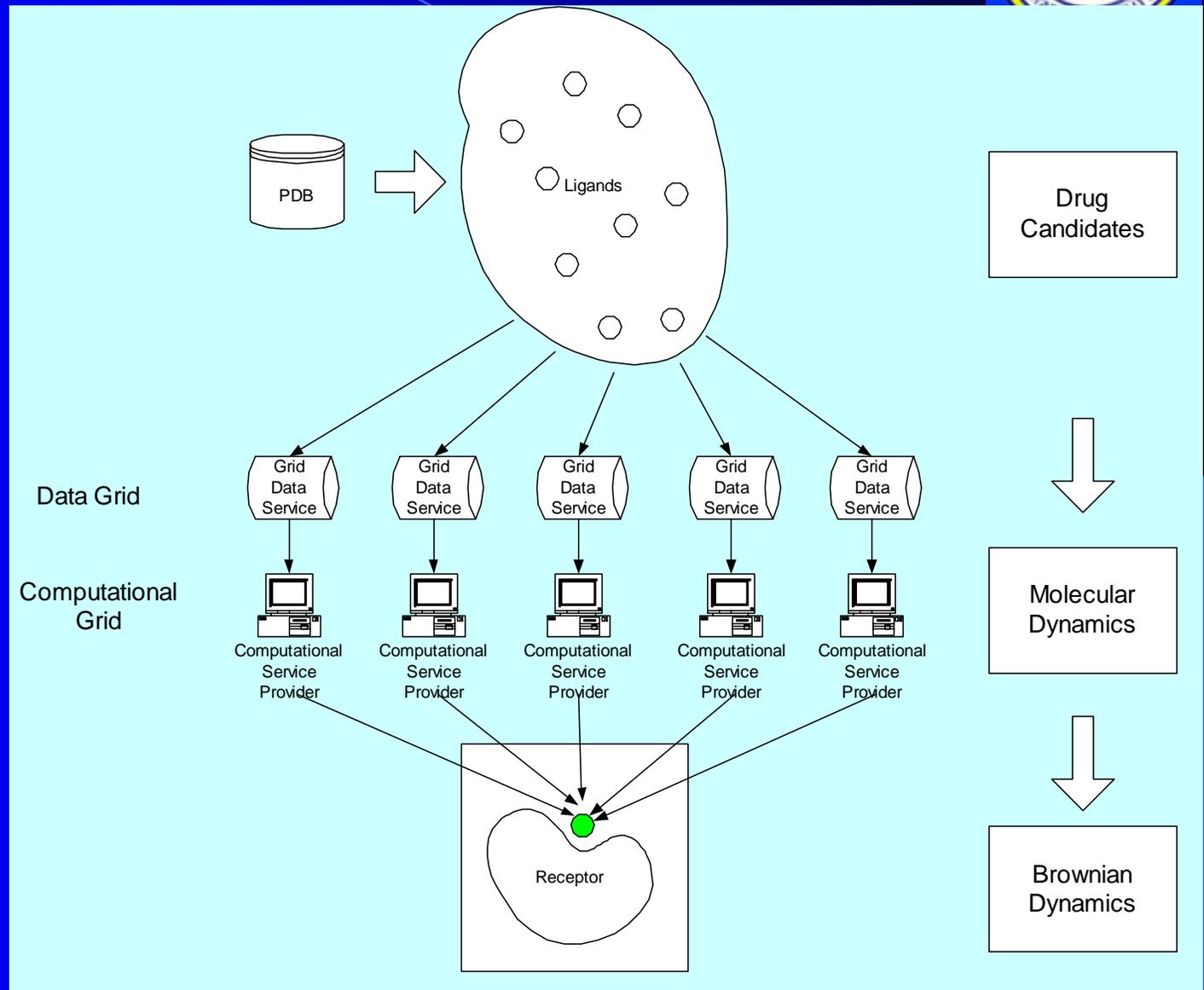
Validation of each
ensembles



Grid-based "Plug-Drug" System



- Data Grid
- Computational Grid
- Hybrid MD/BD





Conclusions

- Conclusions
 - Accelerate convergence of Monte Carlo applications
 - Improve performance and trustworthiness of grid computing from the application level
 - Statistical nature of Monte Carlo methods
 - Characteristics of pseudorandom number generators
 - Develop grid-computing infrastructure software
 - High-performance and reliable large-scale Monte Carlo
 - Grid-based MD/BD simulation



Future Work



- Future Developments
 - GCIMCA based on OGSA
 - Grid-based Monte Carlo System
 - Remote checkpointing facilities using GSoap
 - Grid-based quasi-Monte Carlo applications
 - Biologically inspired lightweight scheduling
 - Applications to Markov Chain Monte Carlo



Relevant Bibliography

1. Y. Li and M. Mascagni (2003), "Analysis of Large-scale Grid-based Monte Carlo Applications," *International Journal of High Performance Computing Applications (IJHPCA)*, **17(4)**: 369-382.
2. Y. Li and M. Mascagni (2004), "Grid-based Quasi-Monte Carlo Applications," accepted for publication in *Monte Carlo Methods and Applications*, 18 pages.
3. Y. Li and M. Mascagni (2004), "E-Science Workflow on the Grid," *Proceedings of the International Association for the Development of the Information Society (IADIS) International Conference: e-Society 2004*, P. Isaías, P. Komers, M. McPherson (eds.), pp. 1041-1046.
4. M. Mascagni and Y. Li (2004), "Computational Infrastructure for Parallel, Distributed, and Grid-based Monte Carlo Computations," *Lecture Notes in Computer Sciences*, **2907**: 39-52.
5. Y. Li and M. Mascagni (2005), "A Bio-inspired Job Scheduling Algorithm for Monte Carlo Applications on a Computational Grid," submitted for consideration in the proceedings of the *17th IMACS World Congress*, 7 pages.