SPEEDUP: A New High Performance Path Integral Monte Carlo Code

A. Bogojević, A. Balaž, and A. Belić
Scientific Computing Laboratory, Institute of Physics
P.O. Box 57, 11001 Belgrade, Serbia and Montenegro
E-mail: {alex, antun, abelic}@phy.bg.ac.yu

Abstract – We present some computational aspects of a new analytical method that systematically improves convergence of path integrals of a generic theory. The method leads to the significant speedup of Monte Carlo algorithms for the calculation of path integrals. We discuss the implementation of the developed code on a cluster as well as the prospects for its Grid-enabling.

I. INTRODUCTION

In this paper we present computational aspects of a series of investigations that have recently been undertaken at the Scientific Computing Laboratory (SCL) [1] at the Institute of Physics in Belgrade. The research deals with the optimization of Monte Carlo algorithms and codes for the calculation of path integrals of a generic quantum theory. As is well known, the path integral formalism [2, 3] represents the most concise and flexible formulation of quantum theories at different levels (quantum mechanics, relativistic and non-relativistic quantum field theory, string theory), as well as of many other complex systems (condensed matter physics, materials science, chemistry, economy). In all of these diverse fields the quantities of interest can be expressed as path integrals, i.e. as infinite limits of multiple integral expressions of the following form

$$A_N = \left(\frac{1}{2\pi\varepsilon_N} \right)^{\frac{N}{2}} \int dq_1 \cdots dq_{N-1} e^{-S_N} \qquad (1)$$

The above expression represents the evolution of the considered system described by variables q from time 0 to T. This evolution is determined by a quantity S which, following standard physics nomenclature is called the action. Time of evolution is subdivided into N equal steps of length $\varepsilon_N = T/N$, with fixed boundary conditions. The boundary conditions depend on the specific object one is calculating, however, the action is most often of the form

$$S = \int_0^T dt \left(\dot{q}^2/2 + V(q) \right), \qquad (2)$$

i.e. it is solely determined by a potential V. Of course, the interpretation of the quantity $A = \lim_{N \to \infty} A_N$ is quite different in quantum field theory, in the description of polymers, or the calculation of stock options.

The very definition of the functional integral makes it necessary to make the transition from the continuum to the discretized theory. Discretization is far from unique. In fact, as we shall see, the details of the discretization procedure are extremely important both for analytical and numerical calculations of path integrals. The naively discretized action is simply

$$S_N = \sum_{n=0}^{N-1} \left(\frac{\delta_n^2}{2\varepsilon_N} + \varepsilon_N V(\bar{q}_n) \right), \qquad (3)$$

where $\delta_n = q_{n+1} - q_n$, and $\bar{q}_n = (q_{n+1} + q_n)/2$. To this we can add arbitrary additional terms like

$$\sum_{n=0}^{N-1} \varepsilon_N \delta_n^2 g(\bar{q}_n), \qquad (4)$$

where g is regular when $\varepsilon_N \to 0$. This does not change the continuum theory ($N \to \infty$ limit) since it goes over into $\varepsilon_N^2 \int_0^T dt\, \dot{q}^2 g(q)$, i.e. it vanishes as ε_N^2. Although such additional terms do not change the final result they do affect the speed of convergence to that result. This paper reports on some computational aspects of a systematic analysis of what constitutes the best solution in the class of all equivalent discretized actions.

There exist several typical groups of complex problems requiring high performance computing resources focused on speed rather than data analysis. The classical examples are *ab initio* simulations of many body systems. Path integrals represent the preferred formalism for treating such systems, or for systems that can be mapped mathematically onto these. Therefore, numerical treatment of path integrals is relevant to the solution of many canonical computing problems. Returning to physics, the most complex quantum field theory calculations (e.g. quantum chromodynamics, QCD) have for decades represented the *de facto* performance benchmarks for the fastest supercomputers, as well as one of the principle reasons for building them. There exist two general approaches to the numerical treatment of path integrals: molecular dynamics and Monte Carlo methods. The second approach [4-7] is applicable in a much wider setting and is the one discussed in this paper.

All the numerical calculations discussed were done on SCL's Linux cluster *Paradox*. The cluster consists of 64+2 Xeon 2.8 GHz processors (32 double processor nodes and server with two processor) with a total of 33 GB RAM, with 1.5 TB of disk space in the nodes and a RAID-1 array of SCSI disks in the server. Communication between nodes is based on a Gigagbit over copper technology established via a 3Com superstackable layer III 4924 switch, while the

connection to the academic network is over a dedicated duplex 1 Gbps optical link. The operating system is Red Hat 9.0/Scientific Linux 3.0.4. *Paradox* has an aggregate speed of Rmax=0.21 Tflops on the Linpack benchmark. The machine was assembled in April 2004 and upgraded to the current configuration in March 2005.

Although one of the fastest machines in the region *Paradox* certainly can't compare in brute force with the world's top supercomputers. The current top five machines are:

1. *BlueGene* (integrated by IBM for the U.S. Department of Energy, 32768 PowerPC processors, aggregate speed of 70.72 Tflops)
2. *Columbia* (integrated by SGI for NASA/Ames, 10160 Altix processors, aggregate speed of 51.87 Tflops)
3. *EarthSimulator* (integrated by NEC for the Earth Simulator Center, Japan, 5120 processors, aggregate speed of 35.86 Tflops)
4. *MareNostrum* (Barcelona Supercomputing Center, Spain, 3564 PowerPC processors, aggregate speed of 20.53 Tflops)
5. *Thunder* (Lawrence Livermore National Laboratory, 4096 Itanium2 processors, aggregate speed of 19.94 Tflops)

Therefore, the current configuration of *Paradox* is 337 times slower than *BlueGene* the current world leader, and about 100 times slower than the fourth and fifth machines on the list.

However, the *Paradox* cluster is dedicated to a small number of researchers affiliated with SCL and is so directly competitive with the above machines when one considers available computing power per user. For example, the path integral simulations reported here utilized more than 50% of the machine's resources, while for all of the above supercomputers resources are shared by hundreds of users at a given time. *Paradox* was designed so as to comply with SCL's principle research vision of scientific computing as a heuristic tool rather than brute force number cruncher. In this respect the research concerning path integrals has allowed us to fulfill this goal. Through numerical experiments made possible by our cluster we first uncovered crucial properties of generic path integrals that brought about a significant speedup of numerical algorithms. We were then able to use that intuition to analytically derive these properties.

The following sections deal with the implementation of our code on an MPI cluster as well as with various aspects related to the porting of that into a Grid environment.

II. PATH INTEGRAL MONTE CARLO ON CLUSTERS

Monte Carlo represents a powerful numerical tool for solving mathematical problems using (pseudo)random numbers. Various aspects of the Monte Carlo method are applicable for solving a wide range of different problems. However, its greatest advantage is in the calculation of multiple integrals – the greater the number of integrals the bigger the advantage over other numerical methods. This is easily shown. In standard quadrature formulas for integrals relative errors δ are proportional to ε^n, where ε is the coarseness of discretization and n depends on the specific quadrature formula.

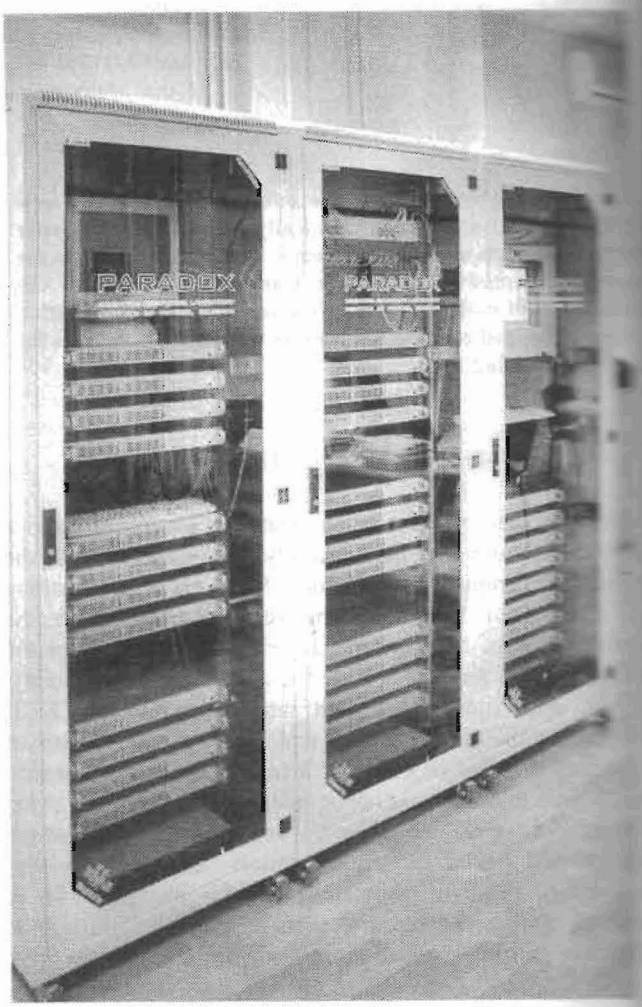

Fig. 1 The *Paradox* cluster at the Scientific Computing Laboratory in Belgrade.

The number of points in which we sample the integrand is inversely proportional to ε, so that the CPU time for calculating d-dimensional integrals satisfies $T_{CPU} \propto \varepsilon^{-d}$. As a result quadrature formulas satisfy

$$\delta \propto T_{CPU}^{-n/d},$$

and we can see, for large d, the ratio n/d becomes very small irrespective of the value of n. As a result, the decrease of relative error requires very large times of calculation. On the other hand, Monte Carlo calculations show the superior behavior

$$\delta \propto T_{CPU}^{-1/2}.$$

Notwithstanding the fact that Monte Carlo is ideally suited to path integral calculations, the problems we are often faced with are exceptionally demanding in computer resources. For example, in QCD the question whether quarks are confined in mesons and nuclei can not be treated numerically as it is far outside the scope of current

supercomputing power even if one considers the coarsest discretizations.

For this reason we have embarked on a set of analytical investigations coupled to numerical experiments geared at better understanding the configurations that dominantly contribute to path integrals. The central idea was the study of the relation between different discretizations of the same theory. Following this guiding idea we arrived at a path integral generalization of the famous Euler summation formula for ordinary integrals. This formula made it possible to decrease the error of the N-fold discretization of a generic path integral from N^{-1} to N^{-2}. The derivation was analytical, however a crucial aspect of the proof of N^{-2} dependence escaped us, i.e. we only saw it numerically. Building on this we set up a series of numerical investigations which ultimately gave us the hint how to [8, 9]:

a) Analytically prove the new N^{-2} dependence, and
b) Generalize the behavior (through an asymptotic expansion) to N^{-p} for arbitrary level p. So far we have done explicit calculations up to $p = 9$, i.e. this has led to an eight order of magnitude improvement in the convergence of the algorithm for a general path integral.

As we have seen in Fig. 2, by using larger levels p we drastically improved convergence to the continuum limit. An important consequence of this is that we can obtain the same precision using much smaller values of N, i.e. much coarser discretizations. This is at the root of the speedup that we find. However, there is a price to be paid. The algebraic complexity of the discretized actions that need to be used at higher p levels grows exponentially, putting an upper bound to p levels that are feasible to use. From Fig. 3 we see that $p = 9$ is still far from that upper bound – the gain of eight orders of magnitude in the speed of convergence far outweighs what is roughly a ten fold increase in computation time due to complexity.

The displayed numerical data shows explicitly what we have derived analytically – that the p level results differ from the continuum amplitudes as polynomials starting with N^{-p}. Because of this, the deviations from the continuum limit $\left|A_N^{(p)} - A\right|$ become exceedingly small for larger values of p making it necessary to use ever larger values of N_{MC} so that the Monte Carlo statistical error does not mask these extremely small deviations. For $p = 6$ we see that although we used an extremely large number of Monte Carlo samples $\left(N_{MC} = 3.68 \cdot 10^{11}\right)$ the statistical errors become of the same order as the deviations already at $N \geq 8$. For $p = 9$ this is the case even for $N = 2$, i.e. we already get the continuum limit within a Monte Carlo error of around 10^{-8}. The simulation that led to this graph took about a week on the *Paradox* cluster, a consequence of the fact that we needed to investigate exceedingly small relative errors (eight decimal precision). Without the speedup in the algorithm these kinds of calculations would be impossible on any machine. Standard calculations, however, require only several decimal places accuracy, and the required computation time for a generic $1d$ theory at level $p = 9$ is on the order of several minutes on *Paradox*. The new algorithm makes it even practical to do such path integrals on a single PC.

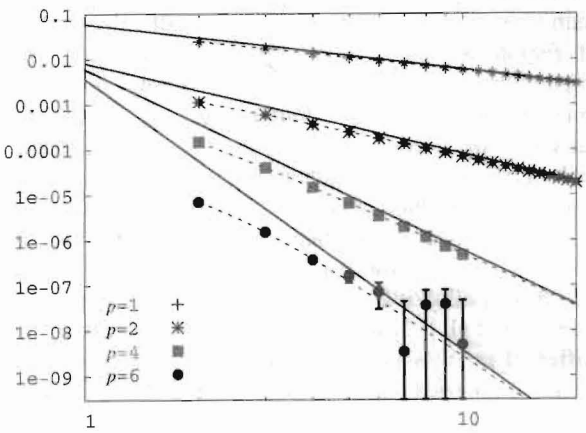

Fig. 2 The deviations from the continuum limit $\left|A_N^{(p)} - A\right|$ as a function of N for p = 1,2,4 and 6 (top to bottom). This particular plot is for the case of an anharmonic oscillator with quartic coupling in a non-perturbative regime. The number of Monte Carlo samples used was $N_{MC} = 9.2\ 10^9$ for p = 1,2, $N_{MC} = 9.2\ 10^{10}$ for p = 4, and $N_{MC} = 3.68\ 10^{11}$ for p = 6. Dashed lines correspond to appropriate $1/N$ polynomial fits to the data. The solid lines give the leading $1/N$ behavior. The level p curve has a $1/N^p$ leading behavior.

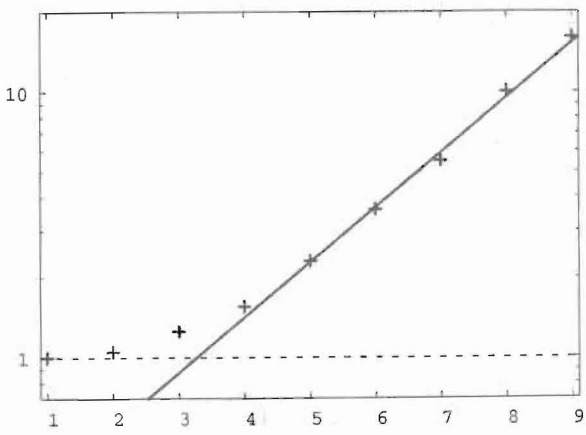

Fig. 3 Relative increase in computation time that comes about from the increased complexity of expression for higher p level effective actions.

This work is currently in the process of being extended to higher dimensional theories with more complex field content. Indications are that the speedup that will be achieved will make tractable calculations of even the most demanding theories.

At the end of this section we briefly comment on the two Monte Carlo algorithms developed for simulations in this investigation. In the first algorithm configurations were generated by a Gaussian distribution function obtained using a semi-classical expansion. The computing time of

this algorithm scales as $N^2 \cdot N_{MC}$ since it is necessary to diagonalize the quadratic form in the exponential of the distribution function. In the second algorithm we implemented the bisection method [6], which scales as $N \cdot N_{MC}$. Therefore, the bisection algorithm is the method of choice for large values of N. On the other hand, as we have illustrated above, our improved method allows us to obtain very precise results using even small values of N. In that region we have found the two algorithms to be comparable both in precision and running time. Both algorithms were used as a double check of all results. In both cases we needed to use random number generators which give a large number of uncorrelated random numbers in a fashion suitable for parallel programming. Our primary random number generator was the Scalable Parallel Random Number Generator library [10, 11] (SPRNG). Following the good practice suggested by Ferrenberg et al. [12] we have checked all our results using a different random number generator. Checks were made with the Numerical Recipes' RAN3 generator [13] with a different seed for each MPI process. Agreement was in all cases well within a one σ interval implying that there were no hidden systematic errors present in either the algorithms or the random number generators.

III. FUTURE PROSPECTS: PATH INTEGRAL MONTE CARLO ON GRIDS

As we have mentioned in the previous section, the new analytical input that has been obtained has led to an immense speedup in Monte Carlo algorithms for path integrals. For the first time it may be possible to do even the most complicated simulations (e.g. confinement in QCD) on existing machines. However, irrespective of the above speedup, the resources that will be needed will certainly exceed the capacities of machines like *Paradox*. On the other hand, the new Grid paradigm of computing on demand makes it possible to solve these long standing problems using shared high performance resources at the level of Europe. It is possible that even the resources at the level of a region like SEE would suffice for this task.

Having this in mind SCL has joined the SEE-GRID project and dedicated the following resources to the SEE-GRID VO: three servers (UI + CE, SE, RB) as well as nine WNs with 17 processors with LCG-2.3.1 middleware. These resources are put at the disposal of SEE-GRID VO (55% of resources [14]) as well as to Serbia's National VO *AEGIS* (Academic and Educational Grid Initiative of Serbia) (45% of resources).

Our key near term goal is to build a Grid-enabled version of our Path Integral Monte Carlo code, and to measure the relative performance of the same seventeen processors working as a local cluster and as a part of the Grid. The trivial parallelization structure of Monte Carlo algorithms makes this a fairly straight forward task. On the other hand our experience so far leads us to believe that there will be only modest overheads to pay in going over to the Grid. As a result, we believe that demanding Grid-enabled path integral calculations will soon be possible within the SEE-GRID framework.

The investigations presented in this paper were done at the Scientific Computing Laboratory at the Institute of Physics in Belgrade. The research was supported in part by the Ministry of Science and Environmental Protection of the Republic of Serbia through projects 1486 and [illegible]. This publication is based on work performed in the framework of the FP6 project SEE-GRID, which is funded by the European Community. The SEE-GRID consortium consists of eleven contractors: ten representatives of incubators of National Grid Initiatives (NGIs) from SEE European countries and CERN. The consortium contractors that represent NGIs are: GRNET (Greece), SZTAKI (Hungary), ICI (Romania), CLPP (Bulgaria), TUBITAK (Turkey), ASA (Albania), BIHARNET (Bosnia-Herzegovina), UKIM (FYRoM), UOB (Serbia-Montenegro), RBI (Croatia).

REFERENCES

[1] URL http://scl.phy.bg.ac.yu/.

[2] R. P. Feynman and A. R. Hibbs, *Quantum Mechanics and Path Integrals* (McGraw-Hill, New York, 1965).

[3] R. P. Feynman, *Statistical Mechanics* (W. A. Benjamin, New York, 1972).

[4] J. A. Barker and D. Henderson, Rev. Mod. Phys. 48, 587 (1976).

[5] M. H. Kalos and P. A. Whitlock, *Monte Carlo Methods*, vol. 1: Basics (John Wiley and Sons, New York, 1986).

[6] D. M. Ceperley, Rev. Mod. Phys. **67**, 279 (1995).

[7] E. L. Pollock and D. M. Ceperley, Phys. Rev. B 30, 2555 (1984).

[8] A. Bogojević, A. Balaž, and A. Belić, Phys. Rev. Lett. **94**, 180403 (2005).

[9] URL http://scl.phy.bg.ac.yu/speedup .

[10] A. Srinivasan, M. Mascagni and D. Ceperley, Parallel Computing, **29**, 69 (2003).

[11] URL http://sprng.cs.fsu.edu/.

[12] A. M. Ferrenberg, D. P. Landau and Y. J. Wong, Phys. Rev. Lett. **69**, 3382 (1992).

[13] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, (Cambridge University Press, 1992).

[14] SEE-GRID project (http://www.see-grid.org/) deliverable D4.1— Operational, Organizational and Policy Schemes