IL NUOVO CIMENTO Vol. 123 B, N. 6-7 Giugno-Luglio 2008
DOI 10.1393/ncb/i2008-10621-8

Monte Carlo simulations for statistical physics: Janus

F. Belletti[1], M. Cotallo[2][3], A. Cruz[2][3], L. A. Fernandez[6][3], A. Gordillo-Guerrero[7][3], M. Guidetti[1], A. Maiorano[1][3], F. Mantovani[1], E. Marinari[4], V. Martin-Mayor[6][3], A. Muñoz Sudupe[6][3], D. Navarro[2], G. Parisi[4], S. Perez-Gaviro[2][3], M. Rossi[5], J. J. Ruiz-Lorenzo[7][3], J. F. Saenz-Lorenzo[3], S. F. Schifano[1], D. Sciretti[2][3], A. Tarancon[2][3], R. Tripiccione[1], J. L. Velasco[2][3], D. Yllanes[6][3] and G. Zanier[5]
The JANUS Collaboration

([1]) *Università di Ferrara and INFN, Sezione di Ferrara - Ferrara, Italy*
([2]) *Universidad de Zaragoza - Zaragoza, Spain*
([3]) *BIFI - Zaragoza, Spain*
([4]) *Università di Roma "La Sapienza" - Rome, Italy*
([5]) *ETHlab - Pergine Valsugana, Italy*
([6]) *Universidad Complutense - Madrid, Spain*
([7]) *Universidad de Extremadura - Badajoz, Spain*

(ricevuto il 15 Settembre 2008; pubblicato online il 14 Novembre 2008)

Summary. — Janus is an FPGA-based computer optimized for the simulation of spin glasses or similar condensed matter systems. Computing requirements in this area are still not met by available commercial systems, so an application-driven machine, boosting performance by approximately a factor 100× is in this case the only viable option to simulate large systems for a time window comparable with experiments.

PACS 05.10.Ln – Monte Carlo methods.
PACS 75.40.Mg – Numerical simulation studies.

1. – Spin glass systems

Several problems in computational physics, requiring computing power that commodity HPC systems are not yet able to deliver, have triggered the development of application-driven computers. This is the case of Lattice Quantum Chromodynamics [1,2] and of large gravitating systems [3].

Spin glass simulation is another challenging problem for which computing power currently available in high-end processors is still not enough. Also in this case, application-driven machines have been developed, such as SUE [4]. Janus, described in this paper, is an evolution of SUE, optimized for spin glasses, but able also to tackle similar problems in

different fields. Consider for definiteness the Edwards-Anderson (EA) spin glass model, defined on a D-dimensional square lattice. Spin variables, defined at the lattice sites, take the value $\sigma = \pm 1$. The interaction energy is a sum of terms associated to all pairs of nearest-neighbor spins in the lattice. For each pair of spins i and j a random coupling $J_{ij} = \pm 1$ is defined, mimicking the structural disorder of the system. The randomness of the J_{ij} has dramatic consequences on the dynamics of the system, since there is no state of the spin variable at a given site that satisfies all energy constraints. This is at the origin of a highly "corrugated" energy landscape and, in turn, of a sluggish approach to equilibrium.

Monte Carlo simulations of these systems are extremely time consuming, but the relevant algorithms have a large and easily-identified parallelism: trivially exploitable is the fact that many replicas of the system (*i.e.* independent assignments of the set of J_{ij}) must be studied, while it is more difficult to update in parallel all mutually not interacting spins. The former approach is used by traditional computers, while the latter is what is done in the Janus system, described below.

2. – Monte Carlo simulations for spin glasses

Monte Carlo algorithms have been implemented on commodity processors using two approaches, *Synchronous Multi-Spin Coding* (SMSC) and *Asynchronous Multi-Spin Coding* (AMSC). In the SMSC approach each program thread simulates one lattice, updating as many spin as possible in parallel; different replicas are studied on a cluster. Typical computer architectures limit to just a few the number of sites that can be handled in parallel. The complementary approach (AMSC) simulates on a single CPU the same spin of many (*e.g.*, 64) replicas, globally boosting performance of the overall simulation. However, in the very relevant case in which we need to simulate large lattices for many Monte Carlo steps, what really matters is to speed up the simulation of just one replica of the system; this is poorly done in both approaches. Indeed, performance an on Intel Core 2 DUO 64-bit CPU 2.4 GHz is 7.0 ns/spin for the SMSC code. The AMSC code is faster, with an update time of 0.77 ns/spin. These values are almost independent of the lattice size, as long as all variables fit inside the cache. These numbers mean that following the evolution of a system of size $L = 80$ for 10^{11}–10^{12} Monte Carlo steps requires tens or even hundreds of years. This is why an application-driven machine has been considered.

Architectural guidelines for an application-driven computer basically boil down to a processor with a very large number ($\simeq$ 1000) of computing cores, each optimized for bit-wise operations, random number generation and comparison, and with a high-bandwidth interface with a local store; such a unit processes single replicas. Hundreds of these processors can then be farmed to make up an ideal spin glass computer, that uses all available sources of algorithmic parallelism.

3. – The Janus system

Janus is a massively parallel machine, matching the requirements outlined above, that makes it possible to simulate a lattice of size $L = 80$–128 for $\simeq 10^{12}$ Monte Carlo steps, corresponding to approximately 1 second, allowing for the first time to make contact with experimental results. Janus is made by FPGA-based reconfigurable processor cores called *Simulation Processors* (SP), connected to a cluster of standard PCs, through an FPGA-based I/O processor (IOP). The SPs can be configured to become a specific computing engine, perform their computation and offload results to the host PC. A set of 16 SPs,

TABLE I. – *Processing time for* 10^{11} *Monte Carlo steps of* 256 *replicas of a* $3D$ *lattice of* 80^3 *points. Runs on Janus are performed on* 256 *SPs, one for each replica. Runs on commodity processors are performed on the optimal number of CPUs to exploit all available parallelism, both for AMSC and SMSC code. The energy dissipation of the systems is also quoted.*

	Janus	AMSC	SMSC
processor	256 SP	2 CPUs	256 (64) CPUs
statistics	256	256	256
wall-clock time	24 days	310 years	24 years
energy	22 GJ	1.8 TJ	18 (4,5) TJ

called Janus-core, arranged at the vertexes of a 2D-grid, have direct low-latency high-bandwidth communication links with nearest neighbors. A large Janus system has many cores: the largest system that we have deployed has 16 Janus cores (that is, 256 SPs) and 8 PC-blades; different set-ups are obviously possible. More details on Janus can be found in [5]. Janus is programmed in a mixed language of C (or Perl) and VHDL. The C program runs on the PC-blade, while the VHDL program implements the computational kernel on the SP processors. The C program is linked together to C libraries implementing the Janus Operating System (JOS), the run-time support allowing the user to monitor the run, and move data to and from the host-blade to the memory buffers implemented into the SPs.

4. – Results and conclusions

The 256-SP Janus system has been assembled in February 2008. The current production code for the EA model [6] runs at 62.5 MHz and uses $\approx 90\%$ of available FPGA resources. Each SP updates 800 spins of a lattice of 80^3 sites at each clock cycle [7] so the spin update time is $\simeq 100\times$ better than achievable on commodity PCs. Table I compares performances on Janus with commodity processors running AMSC and SMSC codes. The overall performance of a 256-SP system is impressive in terms of simulation speed. Results are also remarkable using traditional performance figures: a conservative estimate for the large Janus system is $\simeq 75$ Tera-ops. Janus is also friendly to the environment: each SP consumes ≈ 40 W, that is ≈ 7.5 Giga-ops/watt. For comparison, the best entry of the February 2008 Green500 list (`www.green500.org`) is ≈ 357 Mflops/watt.

In conclusion, Janus is a successful example of the exceptional performances that can be obtained by carefully matching a computer architecture to a specific problem in computational physics.

REFERENCES

[1] BOYLE P. *et al.*, *IBM J. Res. Develop.*, **49** (2005) 351.
[2] BELLETTI F. *et al.*, *Comput. Sci. Engin.*, **8** (2006) 18.
[3] MAKINO J. *et al.*, *A 1.349 Tflops Simulation of Black Holes in a Galactic Center on GRAPE-6*, in *ACM/IEEE Supercomputing Conference SC2000* (2000) 43.
[4] CRUZ A. *et al.*, *Comput. Phys. Commun.*, **133** (2001) 165.
[5] BELLETTI F. *et al.*, to be published in *Comput. Sci. Engin.*, arXiv:0710.3535v2.
[6] BELLETTI F. *et al.*, *Phys. Rev. Lett.*, **101** (2008) 157201.
[7] BELLETTI F. *et al.*, *Comput. Phys. Commun.*, **178** (2008) 208.