LA-UR-09-03785

Approved for public release;
distribution is unlimited.

Title:	Hybrid petacomputing meets cosmology: the Roadrunner Universe project
Author(s):	S. Habib, A. Pope, Z. Lukic, D. Daniel, P. Fasel, N. Desai, K. Heitmann, C. Hsu, L. Ankeny, G. Mark, S. Bhattacharya, J. Ahrens
Intended for:	Journal of Physics: Conference Series - Scientific Discovery Through Advanced Computing, Volume 180, 2009

Journal of Physics: Conference Series **180** (2009) 012019 doi:10.1088/1742-6596/180/1/012019

Hybrid petacomputing meets cosmology: the Roadrunner Universe project

Salman Habib, Adrian Pope, Zarija Lukić, David Daniel, Patricia Fasel, Nehal Desai, Katrin Heitmann, Chung-Hsing Hsu, Lee Ankeny, Graham Mark, Suman Bhattacharya and James Ahrens

Los Alamos National Laboratory, Los Alamos, New Mexico, NM 87545

E-mail: habib@lanl.gov

Abstract. The target of the Roadrunner Universe project at Los Alamos National Laboratory is a set of very large cosmological N-body simulation runs on the hybrid supercomputer Roadrunner, the world's first petaflop platform. Roadrunner's architecture presents opportunities and difficulties characteristic of next-generation supercomputing. We describe a new code designed to optimize performance and scalability by explicitly matching the underlying algorithms to the machine architecture, and by using the physics of the problem as an essential aid in this process. While applications will differ in specific exploits, we believe that such a design process will become increasingly important in the future. The Roadrunner Universe project code, MC^3 (Mesh-based Cosmology Code on the Cell), uses grid and direct particle methods to balance the capabilities of Roadrunner's conventional (Opteron) and accelerator (Cell BE) layers. Mirrored particle caches and spectral techniques are used to overcome communication bandwidth limitations and possible difficulties with complicated particle-grid interaction templates.

1. Introduction

1.1. Motivation for petascale cosmological simulations

Cosmology has made tremendous progress over the last two decades, driven by the development of precision tools for studying the cosmic microwave background (CMB) and the large scale structure of the Universe, as well as its expansion history. These precision probes have been associated with large-volume cosmological surveys: all-sky CMB satellite missions, very large sky area and volume galaxy surveys, as well as observations of the sky in the time domain. Results from modern surveys have cemented a cross-validated cosmological "Standard Model" presenting a comprehensive picture of the evolution history of the Universe and its fundamental make-up: 23% in dark matter which only interacts gravitationally (and a large fraction of which is in localized clumps called halos), 72% in a smooth "dark energy" component described by a cosmological constant, adiabatic Gaussian random initial density fluctuations, and flat spatial geometry [1]. Although this is a great triumph, it has exposed some of the biggest puzzles in physical science: What is dark matter made of? What drives the accelerated expansion of the Universe? Does general relativity need to be modified? What is the origin of primordial fluctuations? Is the Universe exactly spatially flat?

In order to help answer these questions, the observational state of the art continues its rapid progress: surveys now coming on line and those planned for the next decade represent

an improvement in capability by roughly two orders of magnitude. This translates into a determination of cosmological parameters at the 1% level, however, such an advance will only be possible if the underlying theory can be controlled to the sub-percent level. This is a severely demanding task [2].

With one or two exceptions, all cosmological probes are based on (statistical) studies of fluctuations, whether temperature in the case of the CMB, or of the mass distribution in the case of weak gravitational lensing. The fundamental task for theory is to produce accurate predictions for the CMB, for the mass density and velocity field (along with associated statistical measures such as the fluctuation power spectrum), and for tracers of mass and velocity, such as galaxies and galaxy clusters. Cosmological information on large spatial scales is (relatively) easier to interpret because on these scales the physics is essentially linear, however, observable quantities can have a large variance due to finite sampling limitations. At smaller, more nonlinear scales, the statistical limitations can be essentially removed, but modeling becomes significantly more complicated; keeping the associated systematic errors in check is a difficult challenge. There is currently no alternative to precision simulation as the theoretical tool of choice for dealing with the nonlinear regime of structure formation.

Numerical simulations are also essential in providing the underlying skeleton of cosmological structure from which object catalogs (for galaxies, clusters, etc.) can be built. The resulting mock catalogs have many uses: to design and test observational strategies, to understand possible systematic errors therein, and to confront theoretical predictions with observations.

The large-scale distribution of matter – on scales greater than several Mpc (megaparsec; 1 parsec=3.26 light-years) – is determined primarily by gravity. On smaller scales, complex baryonic physics requires modeling of (magneto) hydrodynamic, thermal, chemical, and radiative processes. Because dark matter outweighs baryons by a factor of five, gravity-only N-body simulations provide the bedrock on which all other techniques rest. Therefore, the first set of simulation desiderata are determined by what is required of N-body simulations. These are: (i) ability to resolve dark matter halos and subhalos hosting galaxies, (ii) sufficient volume to control sample variance, and (iii) enough information to model galaxy bias. For next-generation surveys, this translates into simulations with multi-Gpc box-sizes and particle counts in the 10^{11-12} range, all with $\sim$kpc force resolution. These requirements are $1-2$ orders of magnitude beyond the current state of the art.

Aside from sheer size, the time to completion for a single run is a significant factor. Current "hero" simulations take months to complete. Such long times are quite unacceptable if simulations are to play an important role in the inverse analysis of observational data – as they must for next-generation surveys. Brute force Markov Chain Monte Carlo as applied to parameter estimation requires $O(10^{4-5})$ simulation runs. Although sophisticated statistical techniques can be used to lower this requirement by $2-3$ orders of magnitude [3], one still desires an individual run to take a few days at most, i.e., an order of magnitude faster than present practice. (There are other reasons for requiring large numbers of simulations: exploration of different physical mechanisms, testing at scale, computations of covariance matrices., etc.) Therefore, in terms of overall throughput, simulations need to be improved by $2-3$ orders of magnitude. This is the petascale computing challenge for precision cosmology.

1.2. Equations and computational techniques

Structure formation in the Universe is driven by the gravititional instability. In the current theoretical picture, initial density perturbations collapse and merge in a hierarchical fashion to form dark matter halos within a global "cosmic web" structure. Baryonic matter collects in halos, eventually forming stars and galaxies. The collisionless evolution of matter subject only to gravity is given by the Vlasov-Poisson equation in an expanding Universe (with scale factor

$a(t)$ and in comoving coordinates, i.e., with the metric $ds^2 = dt^2 - a(t)^2 d\mathbf{x} \cdot d\mathbf{x}$):

$$\frac{\partial f}{\partial t} + \dot{\mathbf{x}} \cdot \frac{\partial f}{\partial \mathbf{x}} - \nabla\phi \cdot \frac{\partial f}{\partial \mathbf{p}} = 0, \qquad \mathbf{p} = a^2 \dot{\mathbf{x}} \tag{1}$$

$$\nabla^2 \phi = 4\pi G a^2 (\rho(\mathbf{x}, t) - \rho_b(t)), \qquad \rho(\mathbf{x}, t) = a^{-3} m \int d^3\mathbf{p} f(\mathbf{x}, \dot{\mathbf{x}}, t). \tag{2}$$

The phase-space distribution $f(\mathbf{x}, \dot{\mathbf{x}}, t)$ satisfies a flow (Vlasov) equation (1) with a self-consistent force determined by the Poisson equation (2) sourced by the matter inhomogeneity (ρ_b is the background density). To include hydrodynamics, the baryons are split off as a separate species subject to pressure-gradient forces as well as to gravity. We do not consider this extension here.

The problem with the Vlasov-Poisson equation is that it is six-dimensional and forms small structures even with smooth initial conditions, rendering it impossible to solve directly. The way out of this problem is to use N-body techniques. Here tracer particles represent the distribution function and their evolution is generated by self-consistent evaluation of the inter-particle forces and time-stepping of the particle equations of motion. The scale factor itself can be taken as the time variable, and in this case the nontrivial equation of motion is $d\mathbf{p}/da = \nabla\phi/(aH)$, where the Hubble parameter, $H = \dot{a}/a$. In the case of a spatially flat Friedmann-Robertson-Walker cosmology with a cosmological constant, $H = H_0 a^{-3/2}\sqrt{\Omega_0 + \Omega_\Lambda a^3}$ so that H is completely determined by its present value (H_0) and by the corresponding values of the matter and dark energy densities (as a fraction of the critical density), Ω_0 and Ω_Λ.

Inter-particle forces can be computed directly (using approximate methods to avoid the naive N_p^2 scaling) or by using grid-based Poisson solvers, the Particle-Mesh (PM) method [4]. Here the particles are deposited on a regular three-dimensional grid, followed by a Poisson solve implemented via an FFT, the resulting grid force then being interpolated back onto the particles. This method reduces the direct $O(N_p^2)$ computation to $O(N_g \log N_g) + O(N_p)$ where N_g is the number of grid points. Unfortunately, the force resolution is now controlled by the grid size, and leads to a runaway memory requirement. Gpc-scale cosmology runs require a force resolution of order 10 kpc, a spatial dynamic range requirement of $> 10^5$. This imposes in turn a memory requirement of 10-100 PBytes. One solution to this problem is the use of superposition: the PM algorithm contributes the medium/long range force and a direct or tree-based solver handles the near-field regime. (Other approaches include AMR methods and global tree-based algorithms.) MC^3 splits the inter-particle force problem into two parts, a medium resolution solver based on FFTs augmented by a direct particle-particle short-range solver. The biggest FFT provides up to four orders of magnitude of dynamic range, the remaining factor of $10 - 100$ coming from the short-range force evaluations.

2. The Roadrunner architecture

Since the Roadrunner architecture is unusual, we present a short review of it here (Fig. 1). At the top level, Roadrunner is a set of "Connected Units" (CUs), each CU having 180 compute nodes and 12 I/O nodes. Each compute node has two dual-core Opterons clocked at 1.8 GHz and four IBM PowerXcell 8i processors (Cell BEs) coupled to the Opterons via 8×PCIE. Each Cell BE has a peak performance of 100 GFlops (double precision) and twice that in single precision. A Cell BE consists of eight Synergistic Processor Units (SPUs) and one Power Processing Unit (PPU), an in-line CPU with 512KB of L2 cache. The SPUs are connected via the Element Interconnect Bus with a bandwidth of ~ 200 GB/s per Cell BE. The SPUs do not access system memory directly and there is no L1 cache; each SPU has 256K of SRAM-based locally addressable memory for both code and data (not coherent with main memory). Control of memory access is via a DMA-based interface and data transfers can be initiated either by the SPU or the PPU. The central idea is to obtain performance through the use of SIMD parallelism, low memory latency and high memory bandwidth between SPUs.

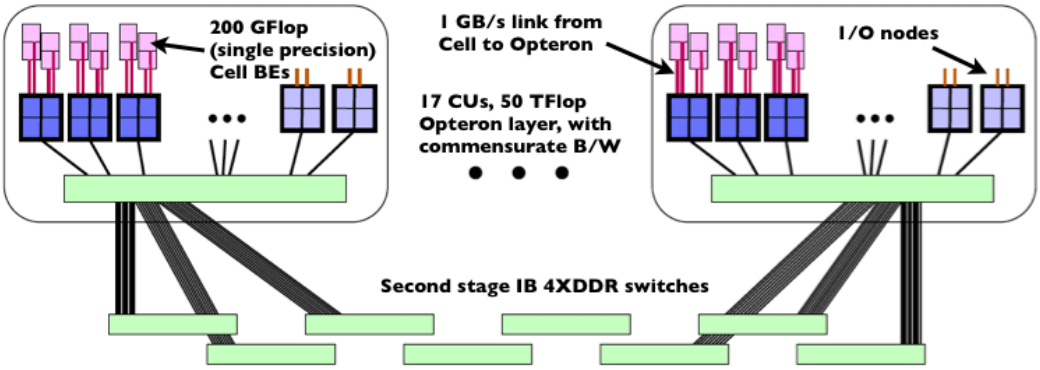

Figure 1. Roadrunner architecture schematic, see text for details.

Local RAM is 16 GB each for the Opterons and for the Cell BEs. The Cell BEs are paired in blades, each blade having 8 GB of NUMA shared memory. There is no local disk storage. The inter-node communication is a fat-tree 4×DDR Infiniband network. For our application, each node is best thought of as two "doublets" each, with one dual-core Opteron and one Cell blade. The reason for this is that the Cell blade pairs share NUMA memory with a ∼5 GB/s bandwidth. Going across blades is much slower – by more than an order of magnitude. The communication bottleneck between the Cell BEs and the Opterons is due to the PCIE bandwidth being ∼ 1 GB/s. Thus, to fully utilize the power of the Cell BEs, they must be given very compute-intensive tasks with only limited (two-way) communication. Roughly speaking, Roadrunner is a factor of 50-100 out of balance in terms of matching bandwidth to raw flops. Consequently, the code design for MC^3 is based on two guiding principles: (i) Simplify and reduce communication as much as possible, compensating with increased computation and memory overhead, and (ii) Match the code and algorithm design to the general machine architecture.

3. MC^3: Cell-accelerated P^3M

The Roadrunner Universe project code, MC^3, is a hybrid design: At the top level of code organization, the medium/long range force is handled by an FFT-based method that operates at the Opteron layer. At this layer, only grid information is stored and manipulated (except when carrying out analysis steps). Particles live at the Cell layer. There is a rough memory balance between the grid and particle data (in most cases, $N_p = N_g$) matching well to the memory organization on the machine. The particle-grid deposition and grid-particle interpolation steps are performed on the Cells with only grid information passing between the two layers. This compensates for the limited bandwidth available between the Cell BEs and the Opterons. The local force calculations stay at the Cell level. Because implementing complicated data structures at the Cell level is difficult, and conditionals can severely degrade performance, our choice for the local force solve is a direct particle-particle interaction, i.e., a P^3M code [4] with hardware acceleration.

At the second level, our approach has two key aspects. (i) Since particle communication across the Cell layer is slow relative to the computational performance of the Cell processors, we have essentially eliminated it as a factor by using *particle overloading*, a mirrored particle cache (details below). (ii) Conventional P^3M codes use a mixture of spatial filtering/differencing and spectral techniques; complicated communication templates can be problematic, particularly at the Cell level. To combat this, MC^3 uses *digital filtering and differencing* in the spectral domain, and only the simplest spatial deposition technique, Cloud-In-Cell (CIC), is used (Cell

layer only). The combination of particle overloading and spectral techniques can eliminate essentially all communication (or at least reduce it to a negligible overhead, if one wants to be more memory efficient), except that in the parallel FFT. The enhanced filter optimization possible with spectral techniques allows for a simple P^3M implementation, with good error control and performance.

In general, P^3M codes perform best when the particle distribution is not very clustered (Molecular Dynamics-like), so that the short-range force calculations do not become overwhelmingly time-consuming. However, extreme clustering is precisely the signature of cosmological structure formation so we must investigate whether Cell-acceleration is up to the task. Given the typical size of the simulation cube (∼Gpc) and the biggest FFTs that can be performed (10^4 cubed), we are led to the conclusion that a typical particle-particle interaction domain will be of size 1 Mpc or less. Given the total number of particles that we can run (about 512^3/doublet), and knowing the mass profile of halos from previous simulations, we find that the worst case scenario is roughly 10^5 particles in the interaction region, which translates into roughly 3×10^{11} floating point operations. Since these computations can be carried out in local coordinate single precision in each doublet node on the Cell BEs, the timing estimate works out to a few seconds per step. The typical number of timesteps is $O(10^4)$, which amounts to roughly 10 hours of run-time. Since code scalability is guaranteed by the particle overloading, these are very encouraging numbers: what currently takes months on other machines, can – potentially – be done on Roadrunner on the timescale of a day.

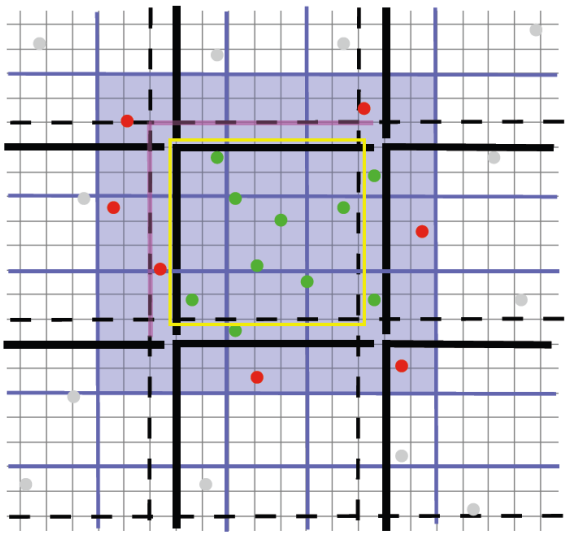

Figure 2. Particle overloading in 2-D: A node holds particles within its spatial domain ("active"/green within yellow boundary) and copies of particles within a given distance from its boundary ("passive"/red in outer blue square). More than one particle copy can be held by neighboring processors – overloading is a mirrored particle cache, not a conventional guard zone.

3.1. Particle overloading

An important aspect of large-volume cosmological simulations is that the density distribution is very highly clustered with an overall topology descriptively referred to as the "cosmic web". The clustering is such that the maximum distance moved by a particle is roughly 30 Mpc, much smaller than the overall scale of the simulation box (∼Gpc). With a 3-D domain decomposition, each doublet node will hold a roughly 100-500 Mpc box on a side, depending on the simulation

run size. The idea behind particle overloading is for a given node to also own particles belonging to a zone of size roughly 30 Mpc (or some similar range) extending out from the nominal spatial boundary for that node (so-called "passive" particles). Note that copies of these particles – essentially a replicated particle cache – will also be held by other processors, in only one of which will they be "active" in the sense of contributing to the density field, hence the term overloading (Figure 2).

The point of this cache is that for a large number of time-steps – potentially for the entire simulation if the cache is big enough – *no explicit particle communication* across nodes is required (from knowledge of the particle's local position, each processor knows whether its copy is active or passive, the particle state being switched as domain boundaries are crossed). The cached particle 'skin' also allows particle deposition and force interpolation to be done using information entirely local to a node, thus grid communication is also reduced.

The size of the overloading layer need not be large enough to avoid communication entirely. A more practical idea is to use a smaller size, say 5 Mpc, and refresh the particle cache as needed. This involves only 26-way near-neighbor communication and we have verified that the penalty is a small fraction of the time spent in a single global (long-range) time step.

This reduction in communication has two associated costs: (i) loss of memory efficiency because of the overloading, (ii) numerical error for the short-range force that will slowly leak in from the edge of the outer (passive particle) nodal boundary (the medium/long-range force remains exact). In our applications, the memory inefficiency can be tolerated to the point where the memory in passive and active particles is roughly equal, but this is not expected to be a limitation in the majority of the cases of interest. In any case, this problem can be completely side-stepped using cache refreshes as mentioned above. The second problem can be mitigated by increasing the boundary thickness. However, in this case also, a better method is to use the cache refresh, "recycling" the passive particles after some finite number of steps: each processor gets rid of all of its passive particles and then refreshes the passive zone with (high accuracy) active particles exchanged from nearest-neighbor nodes, incurring only a small and intermittent communication overhead.

3.2. Spectral filtering and differentiation

Conventional P^{3}M codes employ a combination of spatial and spectral techniques [4]. The spatial operations are the particle deposition, the force interpolation, and the finite-differences for computing field derivatives. The spectral operations include the influence (or pseudo-Green) function FFT solve, digital filtering, and spectral differentiation techniques. In general, spatial techniques are less flexible and more tedious to implement. Also, templates for higher-order spatial operators can be complicated and lead to messy and dense communication patterns, increasing the communication load across the Opteron and Cell BE layers, and being very unsuited to particle operations on the Cell BE.

Accurate P^{3}M codes are usually run with Triangle-Shaped-Cloud (TSC), a higher-order spatial deposition/filtering scheme and with higher-order spatial differencing templates. In grid units, the CIC deposition kernel filters at the level of two grid cells with a large amount of associated "anisotropy noise"; TSC filters at about three grid cells with much reduced noise. MC^3 uses only CIC deposition/interpolation with a Gauss-sinc spectral filter to mimic TSC-like behavior. In addition, a fourth-order Super-Lanczos spectral differentiator [5] is used, along with a sixth-order influence function. This approach has two key benefits: (i) no spatial differentiation is needed reducing work at the Cell level involving data motion and a potential source of communication is eliminated, and (ii) the filtering is flexible and tunable, allowing an excellent force match at only three grid cells, and reducing the anisotropy noise of the basic CIC scheme by better than an order of magnitude. The "Poisson-solve" in the RRU code is the composition of all the kernels above in one single Fourier transform. Note that each component

of the field gradient requires an independent FFT.

Figure 3 shows the excellent force-matching due to the spectral techniques; the multiple realizations in a test code use particle pairs at fixed distances but with random orientations of the separation vector in order to sample the anisotropy imposed by the grid-based calculation of the force. To obtain the short-range force, one subtracts the filtered grid force from the exact

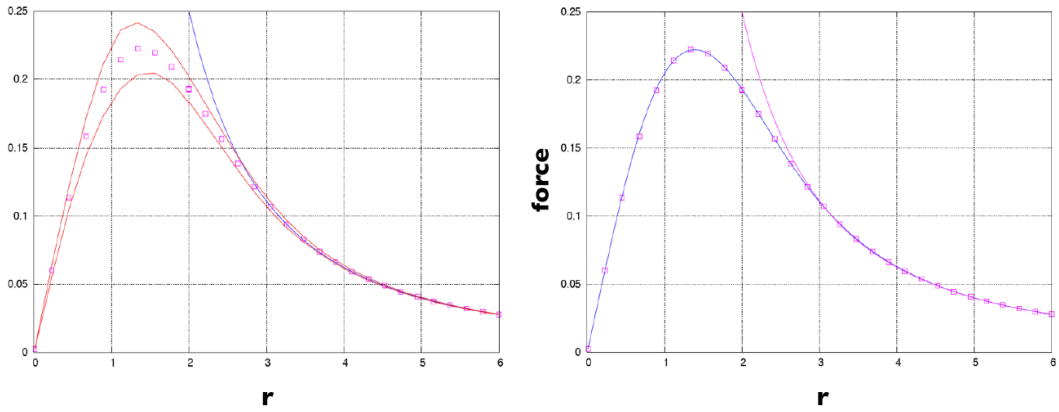

Figure 3. Force between two particles using digital filtering and spectral (Super-Lanczos) differentiation with CIC deposition matching the exact $1/r^2$ behavior at a separation of three grid cells, along with low anisotropy noise – the bracketing red lines showing the 1-σ deviation (left panel). The worst case noise for the total force is only at the ∼percent level. The right panel shows the accuracy of the fitting function used to determine the short range force.

Newtonian force. In our case, this is best done by fitting an expression which has the correct asymptotic behaviors at small and large separation distances. The detailed form of the fit is not particularly illuminating, but its high accuracy can be seen in Fig. 3.

3.3. Time-stepping

The time-stepping in MC^3 is based on the widely employed symplectic scheme. The basic idea behind symplectic integration is not to finite-difference the equations of motion, but to view evolution as a symplectic map on phase space, written in the exponentiated form familiar from Lie methods:

$$\zeta(t) = \exp(-t : H :)\zeta(0) \tag{3}$$

where ζ is a phase-space vector $(\mathbf{x}, \mathbf{v})$ for a particle, H is the effective single-particle Hamiltonian, and the operator, $: H := [H, \ ]_P$, denotes the action of taking the Poisson bracket with the Hamiltonian. Suppose that the Hamiltonian can be writen as the sum of two parts; then by using the Campbell-Baker-Hausdorff (CBH) theorem:

$$\exp(X)\exp(Y) = \exp(X + Y + \frac{1}{2}\{X, Y\} + \frac{1}{12}(\{X, \{X, Y\}\} + \{\{Y, X\}, Y\} + \cdots)) \tag{4}$$

(where the commutator, $\{X, Y\} = XY - YX$), we can build an integrator for the time evolution due to that Hamiltonian. Repeated application of the CBH formula can be used to show that

$$\exp(-t(: H_1 : + : H_2 :)) = \exp(-t : H_1 : /2)\exp(-t : H_2 :)\exp(-t : H_1 : /2) + O(t^3), \tag{5}$$

a second order symplectic integrator. In the basic PM application, the Hamiltonian H_1 is the free particle (kinetic) piece while H_2 is the one-particle effective potential; corresponding respectively

to the 'stream' and 'kick' maps $M_1 = \exp(-t : H_1 :)$ and $M_2 = \exp(-t : H_2 :)$. In the stream map, the particle position is drifted using its known velocity, which remains unchanged; in the kick map, the velocity is updated using the force evaluation, while the position remains unchanged. This way of writing one symmetric 'split-operator' step is termed SKS (stream-kick-stream). A KSK step is another way to have a second-order symplectic integrator.

In the presence of both short and long-range forces, we split the Hamiltonian into two parts, $H = H_{sr} + H_{lr}$ where H_{sr} contains the kinetic energy term and the direct particle-particle interaction potential (with an associated map M_{sr}), whereas H_{lr} is just the long range potential, corresponding to the map M_{lr}. Since the long range force varies relatively slowly, we construct a single time-step map by subcycling M_{sr}:

$$M_{full}(t) = M_{lr}(t/2)(M_{sr}(t/n_c))^{n_c} M_{lr}(t/2), \tag{6}$$

the total map M_{sr} being a usual second-order symplectic integrator. This corresponds to a KSK step, where the S is not an exact stream step as in the PM case, but has enough M_{sr} steps composed together to obtain the required accuracy [6].

For typical problems the number, n_c, of short time steps for each long time step will range between $10 - 100$. In gravity-only mode the total time in the code is spent roughly equally between the long and short time steps. Once hydrodynamics, cooling, and feedback terms are added, the short-range part of the code dominates and the effective efficiency increases (although so does the workload).

4. Notes on implementation

MC^3 is written in C/C++/MPI and is highly optimized for performance and efficient memory use. For enhanced scalability, the parallelization is based on a 3-D domain decomposition of the periodic simulation cube. The FFT implementation is based on a 2-D domain decomposition with repeated application of optimized serial 1-D FFTs (we use FFTW [7]). This methodology is known to scale to thousands of processors. The initializer for MC^3 has been implemented and is based on spectral methods for obtaining realizations of a Gaussian random field. A sufficiently large FFT ($N_p = N_g$) can be performed directly at the Opteron level even for the largest problems of interest, with $N_g = 10^4$.

Although Roadrunner is one machine, it has three programming levels. At the top level are the Opterons connected via Infiniband – general purpose CPUs with MPI as the communication library. The Cell level itself consists of programming the PPU and the SPUs. Communication across the top level and the Cells is handled by the Data Communication and Synchronization (DaCS) library which provides mechanisms for managing, synchronizing, and sharing data. The PPU is a stripped down PowerPC without an instruction pool or branch prediction but clocked at 3.2 GHz. The PPU functions as a bridge from the Opterons to the eight SPU (in-order) vector co-processors and can also carry out independent computations. Altivec registers are avalable to the PPU, and we have found that compiler optimizations can have major impacts on performance resulting in speedups by a factor of 3. With a lightweight programming style, the PPU can yield good performance. Thus, it is worthwhile to look for a role for the PPU beyond that simply of work partitioning, launching tasks for SPUs, and retrieving results from them.

Particle-grid interactions can be complicated to handle on the SPUs because of the way arrays are stored (column/row order, Fortran/C) since spatial locality of particle information does not map to memory locality for the grid. Data reordering by the PPU leads to too high an overhead, and we did not try data movement at the SPU level due to coherence issues. Because of its access to all available memory, the PPU is our choice to carry out the particle deposition and force interpolation steps (CIC and inverse CIC). Compiler optimization results in an Opteron level of performance for this task.

In MC^3, data transfers consist of a DaCS launch and SPU DMA – we do not use the PPU (Cf. Section 2). To maximize performance in the absence of a cache, we use asynchronous memory access to overlap data movement and computation ('cache by hand'). We found the availability of high-performance libraries for the SPU a necessary aid in obtaining good performance (many useful intrinsics exist, e.g., vector instructions, memory control). MC^3 employs the SPUs for the "hot" part of the computation – the direct particle-particle force calculation and the associated particle position and velocity updates. The number of pairwise force computations in the P^{3}M algorithm is controlled by the size of the matching zone where the short-range force computation is handed over to the long-range, grid-based, PM force calculation. This local range control is affected in MC^3 by a chaining mesh with a (radial) bucket size of three grid cells. Our results show that the short-range computation can be run on the Cells more than fifty times faster than on the Opterons.

Because some of the techniques used in MC^3 are new, a data-parallel PM/P^{3}M reference code has been implemented that, aside from the overloading, is a close algorithmic copy. This reference code is based on our earlier MC^2 (Mesh-based Cosmology Code) suite which has been tested and used extensively [8]. The reference code is fully operational and is being used for validation of MC^3 in a phased manner. An example is shown in Fig. 4, where the density fluctuation power spectrum is compared (as a ratio) for a medium-resolution test run, primarily as a check of the particle overloading implementation.

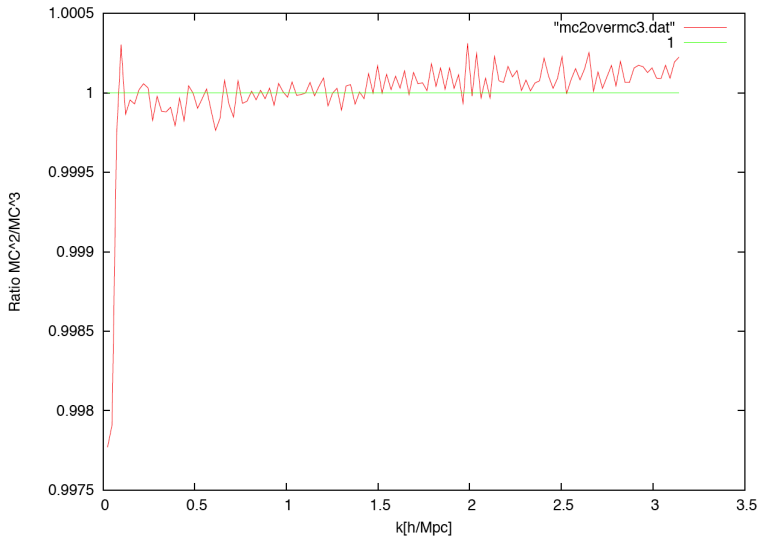

Figure 4. Ratio of the matter power spectrum for the reference code relative to MC^3 as a function of wave number for a medium resolution run (particle overloading test). The relative error is only 2×10^{-4}. The greater error at the lowest k modes is a finite sampling artifact.

A key aspect of the MC^3 code suite is a capability for very fast "on the fly" data analysis. Because the raw data from each run can easily exceed hundreds of TB, it is essential that as much analysis as possible be already performed before the code output is dumped to the file system. The data reduction required is roughly a factor of two orders of magnitude. This data reduction is achieved primarily by a very fast halo finder that can identify clustered regions and distill the particle information contained therein into a much-reduced data vector for individual halos. The halo finder runs on the Opteron layer: The serial part of this halo finder is based on a KD tree algorithm and the parallelization is based on the particle overloading already implemented for the time-stepping. The halo finder's overloading zone is taken to be sufficiently

large compared to the size of the largest halos, ~ 5 Mpc being typically sufficient. Because only near-neighbor communication is involved (26 nodes for a given single node), the halo finder is arbitrarily scalable: scaling tests indicate that a trillion particle analysis on Roadrunner should take no more than several minutes.

5. Summary and discussion

To summarize, it is often stated that programming the Roadrunner hybrid system proceeds in either a top-down fashion (put code on the Opterons and transfer the computationally intensive parts to the Cell) or via a bottom-up approach (view the machine primarily as an MPI cluster of SPUs). MC^3 represents a third alternative where the machine is considered as an entire system and the code is designed around it, not based on the particular ideologies mentioned above, but using elements of both. If this is done flexibly enough, successful ideas can be reused – perhaps in different forms – on other architectures. In our case, the particle overloading infrastructure can be used to make MC^3 run on an architecture like that of the Cray XT5 with a different serial solution for the local force solver, such as a tree-based method, instead of the direct particle-particle force evaluation.

Validation against the reference code and convergence testing of MC^3 is currently in progress, as the Roadrunner computer comes closer to achieving full operational status. Timing tests have so far yielded results very close to the initial estimates. The science program for the Roadrunner Universe runs is described in the project Blue Book to be released shortly.

Future development of MC^3 will be in several different directions. The two most obvious capability extensions are the addition of hydrodynamics (both SPH and grid-based) solvers for the Cell BE and the ability to handle small-volume simulations with very high mass and force resolution. In addition, we aim to add sub-grid modules (e.g., star formation, supernova feedback) and post-processing capabilities such as mock catalogs for galaxies and the modeling of galaxy clusters based on semi-analytic prescriptions [9]. More radical extensions include modifications of general relativity on large scales (as a possible explanation of cosmic acceleration) and self-consistent inclusion of dark energy fluctuations. Different methods for introducing non-Gaussian initial conditions will also be implemented.

6. Acknowledgements

We are indebted to Cornell Wright, Sriram Swaminarayan, and Scott Pakin for their generous help with Roadrunner programming issues. SH wishes to acknowledge many discussions on particle codes with Andreas Adelmann, Viktor Decyk, Robert Ryne, and Martin White, and to extend a special note of appreciation to Andy White for his encouragement. AP was supported by a LANL Director's Fellowship. This work was supported by the LDRD program at Los Alamos National Laboratory.

[1] Komatsu E et al. [WMAP Team] 2009 *Astrophys. J. Supp.* **180**, 330
[2] See, e.g., the report of the tri-agency Dark Energy Task Force, Albrecht A et al. 2006 *arXiv:astro-ph/0609591*
[3] Habib S, Heitmann K, Higdon D, Nakhleh C and Williams B 2007 *Phys. Rev.* D **76**, 083503
[4] Hockney R W and Eastwood J W 1988 *Computer Simulation Using Particles* (New York: Adam Hilger)
[5] Hamming R W 1998 *Digital Filters* (Mineola: Dover)
[6] For a similar application in beam physics, see Qiang J, Ryne R D, Habib S and Decyk V 2000 *J. Comp. Phys.* **163**, 434
[7] Frigo M and Johnson S G 2005 *Procedings of the IEEE* **93** 216
[8] Heitmann K et al. 2008 *Comp. Sci. Disc.* **1**, 015003
[9] See, e.g., Bode P, Ostriker J P and Vikhlinin A 2009 *arXiv:0905.3748*