Load balancing of molecular dynamics simulation with NWChem

by T. P. Straatsma
J. A. McCammon

NWChem is a computational chemistry software suite developed for massively parallel computers in the W. R. Wiley Environmental Molecular Sciences Laboratory at the U.S. Department of Energy's Pacific Northwest National Laboratory. This software integrates a range of modules for computational chemistry applications, including classical molecular dynamics simulations and quantum mechanical calculations. This contribution provides details of the classical molecular dynamics module and focuses on issues related to load balancing on massively parallel computers, in particular the IBM SP™ and the Cray T3E™ as examples of distributed and shared memory massively parallel architectures. The implementation of the molecular dynamics module of NWChem is based on a domain decomposition of the chemical system, taking advantage of the distribution of data to reduce the memory requirements and the locality of intermolecular interactions to reduce the communication requirements. This approach results in a more complex implementation because of the requirement of periodic atomic reassignments and the need for sophisticated load-balancing techniques.

Molecular dynamics simulations are an important tool in the study and rational design of molecular systems and materials, providing information about the behavior of chemical systems that can be difficult to obtain by other means.[1] These properties generally are obtained as statistical mechanical averages of atom trajectories. Unfortunately, these averages usually converge only slowly with the length of the simulation or the size of the molecular system. Considerable computational resources are required, even for small molecular systems (tens of thousands of atoms) and short simulation times (nanoseconds).

Several strategies have been suggested to take advantage of the development of massively parallel computer architectures. Molecular dynamics simulations generate the time evolution of chemical systems, which is an inherently sequential process. The calculation of pair-wise additive atomic forces needed to advance the system a discrete time step, however, does lend itself to parallelization. Parallelization approaches fall in two categories: data decomposition (including replicated data, atom decomposition, and force decomposition) and domain decomposition. Theoretical scaling of memory, calculation, and communication requirements for the different parallelization methods is given in Table 1, where N is the number of atoms in the molecular system, and p is the number of processors used. The scaling of communication in the domain decomposition is valid for systems with large numbers of atoms, a large number of processors, and use of a limited interaction range.

In the replicated data approach, the computational work is distributed over the available p processors, each of which holds a complete copy of the N-particle system. At each step, a processor calculates the interactions for N/p atoms and updates the forces and coordinates, requiring all-to-all communication.

0018-8670/01/$5.00 © 2001 IBM

Table 1 Theoretical parallel scaling of molecular dynamics simulation parallelization strategies

	Memory	Computation	Communication
Replicated data	N	N/p	N
Atom decomposition	N/p	N/p	N
Force decomposition	$N/\sqrt{p}$	N/p	$N/\sqrt{p}$
Domain decomposition	N/p	N/p	N/p

Atom decomposition is a variant in which the atom data are distributed, but all-to-all communication is still required. Replicated data and atom decomposition approaches have been used to parallelize existing sequential simulations codes.[2–6] However, the cost of communication makes these approaches less efficient on massively parallel computers. In replicated data and atom decomposition, the calculation of the force matrix is generally row-wise distributed over the processors. Block-wise distribution of the force matrix calculations leads to force decomposition approaches with a more balanced memory and communication requirement.[7] In all of these approaches, the list of atoms is directly distributed, and the memory requirements and cost of computation and communication can be precisely controlled for each processor. In this way the complexity of load balancing is significantly reduced.

In the second class of parallelization approaches, the physical space in the molecular system is distributed instead of the atom list.[8–13] In this scheme, each processor is responsible for the calculation of forces of the atoms that occupy the region or domain assigned to the processor. This scheme distributes the atoms over the available processors, reducing the memory requirements. For the calculations of local interactions, data are needed only for atoms in neighboring domains, thereby reducing the amount of communication. This approach is especially efficient for large molecular systems simulated on massively parallel computers. Implementations of this approach are much more complex than those based on data decomposition because of the fluctuating number of atoms per processor and the required periodic reassignment of atoms that move from the domain of one to the domain of another processor. A second complication is the need for dynamic load balancing.

The performance of a parallel algorithm on n processors is generally given in terms of the parallel speedup S or the parallel efficiency E, relative to a smaller number of processors m, defined as:

$$S = \frac{T_m}{T_n} \tag{1}$$

and

$$E = \frac{mT_m}{nT_n} \tag{2}$$

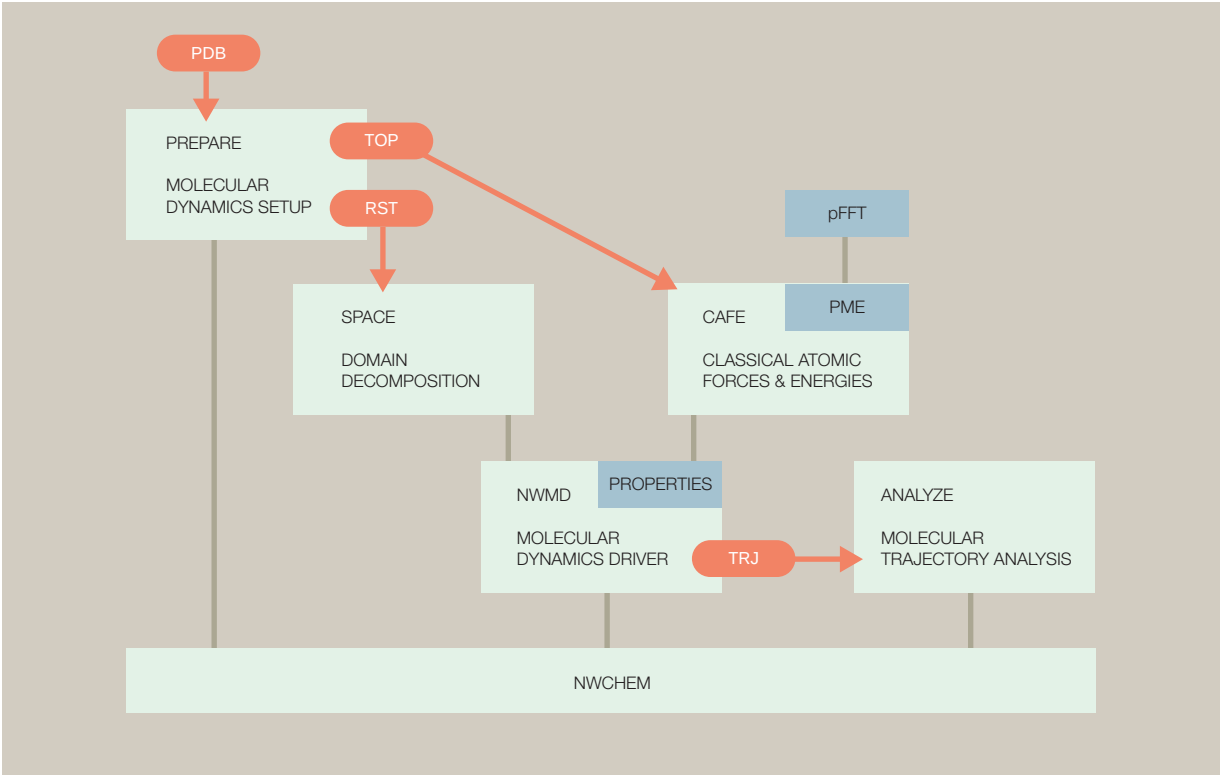
where T_m is the wall-clock time to complete the calculation on m processors. This measure of parallel efficiency not only depends on the algorithm but also on the computer architecture. In particular, the cost of communication is an important factor in parallel efficiency. In general, efficient parallel algorithms will minimize the amount of data to be communicated by data replication, data packing, or data recalculation, minimize the number of messages, and avoid simultaneous access to the same resource.

NWChem

The NWChem software suite provides many computational chemistry modules for molecular and periodic systems, ranging from quantum mechanical to classical mechanical methods.[14,15] These approaches may be combined to perform quantum molecular dynamics and QM/MM (quantum mechanical/molecular mechanical) simulations. The implemented methods are available on most of the high-performance, massively parallel computer architectures, but also on single, or clusters of, desktop workstations and personal computers running the Linux** operating system.

The computational modules for molecular electronic structure calculations include self-consistent field (SCF) or Hartree-Fock (RHF [restricted Hartree-Fock], UHF [unrestricted Hartree-Fock], high-spin ROHF [restricted open-shell Hartree-Fock]), Gaussian orbital-based density functional theory (DFT) using many local and nonlocal exchange correlation potentials, perturbation theory (MP2, or Moller-

Figure 1 NWChem modules for molecular dynamics simulation



Plesset 2nd order energy correction, RI-MP2), complete active space SCF (CASSCF), coupled-cluster singles doubles and noniterative triples correction (CCSD(T)) and selected-CI (configuration interaction). In addition, there are modules for pseudopotential plane-wave electronic structure and periodic Gaussian-based DFT calculations. The classical mechanics module includes molecular dynamics simulation and free energy perturbation and integration.

The chemistry modules in NWChem are built upon a common set of high-performance parallel software tools. The separation of these tools from the chemistry modules provides the flexibility to adapt and tune the code for different computer architectures. A run-time database (RTDB) provides the mechanism with which information is shared between modules. In general, there is a single database per calculation with parallel read/write access by all processors. The memory allocator (MA) provides a standardized tool for the dynamic allocation of local memory, i.e., memory not shared with other processors. The MA library includes routines for management, debugging, verification, usage statistics, and availability of heap and stack memory. The Global Array (GA) toolkit provides a distributed data mechanism for the allocation and access of nonlocal memory.[16] One of the most important features the GA tools provide is the one-sided asynchronous access to global memory elements. The GA memory model allows the application programmer to determine the locality of data and manage the memory hierarchy in the parallel application. Other parallel tools include parallel eigensolvers (PeIGS) and three-dimensional parallel Fast Fourier transformations (pFFT).

Molecular dynamics simulation

For molecular dynamics simulations, NWChem includes the five modules as illustrated in Figure 1. The PREPARE module is used to analyze the molecular structure provided as a Protein Data Bank (PDB) file and, using parameter database files for the specified force field, generates the two data files required for

the simulation: a topology file containing static information on the system, such as the force field parameters, lists of bonds, angles and torsions, and excluded pairs, and a restart file containing the dynamic information on the system, such as atomic coordinates and velocities.

The NWMD (Northwest Molecular Dynamics) module contains the driver routines for energy minimization, molecular dynamics simulations, thermodynamic perturbation, and thermodynamic integration calculations. This module relies on a module that deals exclusively with the calculation of classical atomic forces and energies (CAFE) and a module called SPACE that handles the data distribution and associated communications for domain decomposition. This separation of tasks will allow for the future implementation of other data decomposition algorithms in modules that can replace SPACE, without the need for modifying other modules.

The ANALYZE module contains routines for the analysis of molecular trajectories generated and currently includes root-mean-square deviation (RMSD) analysis, analysis of internal coordinates (Ramachandran plots),[17] and essential dynamics (ED) analysis.[18,19]

Domain decomposition offers the best theoretical scalability of memory use and communication cost for molecular dynamics simulations of large molecular systems on massively parallel computers. This decomposition is implemented in NWChem as follows. The complete molecular volume is defined as the periodic cell in systems with periodic boundary conditions and the smallest rectangular box that completely encloses the molecular system for nonperiodic systems. By slicing the molecular system in all three dimensions, the simulation volume is divided into rectangular subboxes. Next, these subboxes are assigned to the available processors. Each processor owns a range of subboxes in each of the three dimensions, i.e., the processors are logically arranged into a three-dimensional grid. In this way, each processor owns a subgrid of subboxes. This arrangement preserves locality of the subboxes and allows a unique number to be assigned to each processor, as well as to each subbox from which the location in the processor grid and the subbox grid can be determined without need for communication. Although currently not implemented, the assignment of subboxes to processors can be easily modified to take advantage of the fact that communication between same-node processors is more efficient than communication between off-node processors.

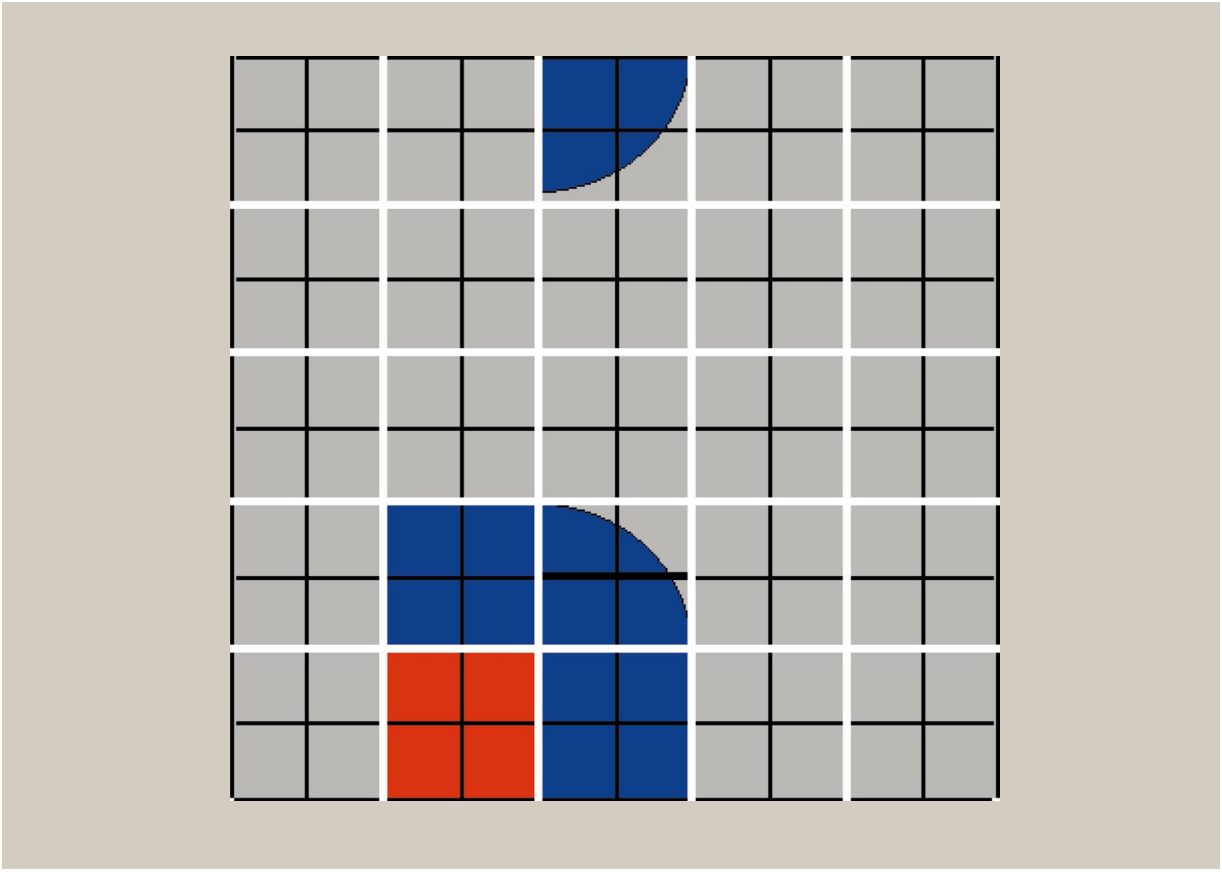
There is no restriction on the size of the subboxes, and the user has complete control over the number of subboxes that are defined. Based on the cutoff radius and the size of the subboxes, each processor generates a pair list of subboxes for which interactions need to be evaluated. Initially, each processor generates this list for subbox pairs in which local subboxes and nonlocal subboxes in half of the directions are involved, i.e., north, but not south, etc. Pairs of subboxes on the same processor are always handled by that processor. In this way all subbox pairs with interactions within the cutoff radius are included in one of the subbox pair lists without double counting.

Note that interactions may also extend beyond subboxes on the immediate neighboring processors. This is basically a distributed implementation of a cell-index list. Since the distribution and size of all subboxes is known on all processors, periodic boundary conditions are easily imposed. Since all data in a subbox are transferred between processors, subboxes should not be made too much larger than the cutoff radius. They can, however, be smaller than the cutoff radius. The current implementation will make the subboxes as large as possible under the restrictions that the size should be less than the cutoff radius times a factor slightly larger than one, and the restriction that each processor should at least own one subbox. This default decomposition can be changed by the user with explicit decomposition input directives.

Figure 2 depicts a two-dimensional representation of the distribution of subboxes. The space a molecular system occupies is divided into subboxes that are distributed over the available processors. This figure illustrates the two main advantages of this distribution: reduced memory requirement, and reduced communication. The processor in red owns a subset of the molecular system and needs to evaluate only interactions with atoms in subboxes within the cutoff radius on half of its neighboring processors, identified by the blue subboxes. In this figure the use of periodic boundary conditions is assumed, resulting in interaction between the red processor with the blue area on the processor in the top row. Note that in this figure each processor owns four subboxes, and that the cutoff radius extends two subboxes.

In NWChem, each processor sequentially processes the subbox pairs in its list, retrieving the atomic coordinates from the remote node, evaluating the con-

Figure 2 Two-dimensional representation of domain decomposition



tribution to forces and energies, and accumulating force contributions to the appropriate array on the remote processor. The subbox pair list on each processor is ordered such that atomic coordinates from a remote subbox are retrieved only once per molecular dynamics step. Similarly, atomic force contributions for a remote subbox are accumulated to the remote processor only once. For each subbox pair, an atom pair list is periodically evaluated and locally stored for the calculation of the atomic forces. All communication in the processing of the subbox pair lists is performed with one-sided asynchronous GA communication calls.

Domain decomposition molecular dynamics is efficient because of the locality of the atomic interactions when cutoff radii are used. However, especially for electrostatic interactions, the use of cutoff radii can be a serious approximation. To be able to account for electrostatic interactions beyond the cutoff radius in an approximate way, the particle-mesh Ewald (PME) method has been implemented.[20] In this method, short-range interactions are evaluated as usual, but with a scaling function that depends on the cutoff radius. Long-range contributions are approximated in the form of a discrete convolution on an interpolated grid of charges. This method requires the Fourier transform of the charge grid to perform the convolution efficiently. The pFFT implemented in NWChem distributes the grid in slabs for efficiency. The implementation in NWChem allows the use of a subset of the available processors to be used for the transforms while the other processors are evaluating local forces. Since all processors need the potential grid to calculate the PME forces, this part of the calculation is separated from the evaluation of the charge grid and performed after all local forces have been evaluated. In practical simulations, this

separation avoids synchronization of all processors before the PME forces evaluation.

Load balancing

After the calculation of the forces, a synchronization of all processors is required to ensure that all contributions from remote processors have been properly accumulated. Particularly in simulations of heterogeneous systems, the time to calculate the forces will be different on each processor, resulting in substantial synchronization time at this point in the calculation. To make molecular simulations of such systems efficient, dynamic load-balancing techniques are needed. Because the force evaluation in NWChem involves only asynchronous communication, the distribution of the idle time on each processor at this synchronization can be used to determine how the work can be more evenly distributed.

In NWChem, two methods for load balancing have been implemented. The first method is based on a redistribution of the subbox pair lists.[21] To avoid unnecessary communication or replication of data, in NWChem each subbox pair is handled by one of the processors that owns one of the subboxes. In the current implementation, each processor, in order of increasing synchronization time, transfers one interprocessor box pair to its least busy neighbor processor for which it handles the subbox pair. Further, each processor can only receive one additional subbox pair. This cascading subbox pair distribution avoids the back and forth exchange of a subbox pair between the two busiest processors if only one subbox pair is exchanged, as was observed for some simulations. The subbox pair redistribution requires only minimal additional communication. Furthermore, this method allows the interatomic calculations for each subbox pair to be evaluated in the same order, leaving the generated molecular trajectory unaffected by the load balancing.

The second load-balancing technique is based on a resizing of the processor domains.[22] The size of the subboxes on the busiest processor is reduced to reduce the workload. This reduction increases the size of the subboxes, and the corresponding load, on the other processors. A limit on the minimum size of the subboxes ensures that the subbox pair lists cover all interactions within the specified cutoff radius without increasing the total number of subbox pairs. This method of load balancing has the disadvantage of requiring the communication of atomic data because of the resulting redistribution of atoms, as well as divergence of the molecular trajectory caused by differences in numerical accuracy from the changed order in which contributions to the atomic forces are evaluated as a result of the subbox resizing.

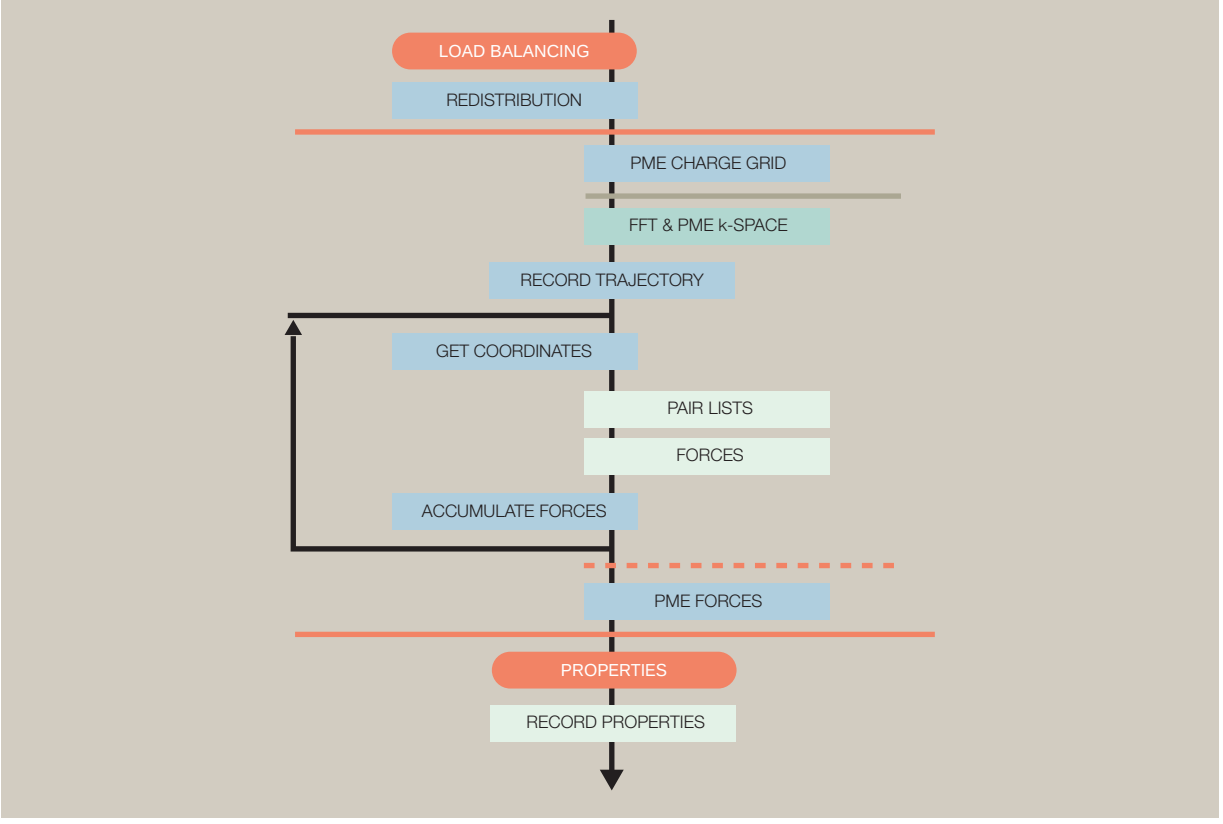
In NWChem, both load-balancing techniques can be used in a simulation. The subbox pair list redistribution does not change the order in which interactions are evaluated, and thus does not suffer from numerical differences that may lead to diverging trajectories. Moreover, the method exchanges subbox pair entries only, and thus requires a minimum of communication. Subbox pair resizing, however, requires a redistribution of atoms and thus significantly more communication. Also, since the number and order of atoms in each subbox is changing, the order of the calculations of interactions is changing. Consequently, numerical differences lead to diverging trajectories. Therefore, in the combined load balancing, subbox resizing is applied only after a specified number of load-balancing steps by subbox pair redistribution do not reduce the accumulated synchronization time.

Load-balancing efficiency of NWChem

The basic components in a molecular dynamics step as implemented in NWChem are depicted in Figure 3 from the top down and include the following: the load-balancing step, atom redistribution, global synchronization to ensure coordinates are current and available on all processors, PME charge grid evaluation, processor subset synchronization, reciprocal space calculations that include two three-dimensional Fast Fourier Transforms, recording of the trajectory, the calculation of interatomic forces in a loop consisting of retrieval of coordinates, calculation of pair lists and forces and accumulation of these forces, a global synchronization to ensure forces have all been evaluated, and the calculation and recording of properties. All steps between the two global synchronizations include only asynchronous communication, which allows time stamps around the second global synchronization to be used to determine the idle time on each of the processors due to load imbalance. This synchronization time is used in the dynamic load balancing based on the implementations described in the previous section.

In Figure 3, boxes off-center to the left are functions performed by the domain decomposition model SPACE, the center boxes are functions performed by the driver, and off-center boxes to the right are force field functions performed by the CAFE module. The

Figure 3 Flowchart of the calculation of forces and energies



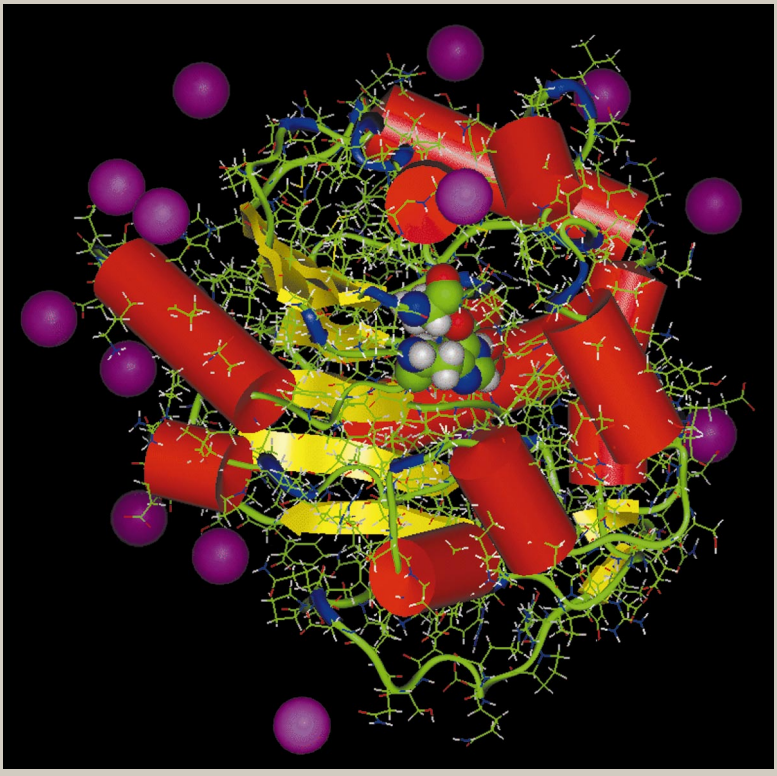
two horizontal red lines represent global synchronizations to ensure all coordinates are current before the forces are calculated, and to ensure that all forces are available before advancing the coordinates. Boxes in red contain global operations, the one in green contains processor subset communications, and those in blue contain asynchronous communications. The green horizontal line represents a processor subset synchronization to ensure that the PME charge grid calculation was completed. The red dashed line is a wait state on each processor until the PME potential grid is available. Note that recording of the trajectory is included in the load balancing and involves asynchronous communication initiated by the processor that performs I/O operations.

The efficiency of the load balancing $\mathcal{E}$ can be expressed in terms of the time t from immediately before the first global synchronization to immediately after the second global synchronization, which will be the same on all n processors, and the accumulated synchronization time $T^{synch} = \sum_{i=1}^{n} t_i^{synch}$ of the second global synchronization.

$$\mathcal{E} = \frac{nt - T^{synch}}{nt} \tag{3}$$

To determine the behavior and efficiency of the different load-balancing options in NWChem, molecular dynamics simulations of the enzyme haloalkane-dehalogenase in aqueous solution (see Figure 4, which shows the secondary structure elements, within space-filling representation, the Na^+ counter ions and the active site residues Asp124, Asp260, and His289) were carried out on 27 processors of the IBM SP* with single P2SC processor nodes, the IBM SP with four 604 processor silver nodes, and the Cray T3E**-900. The molecular system consists of the 4858-atom enzyme, 17 counterions, and 12128 water molecules in a periodic simulation volume. The simulations were carried out using the particle-mesh

Figure 4 Display of the enzyme haloalkanedehalogenase

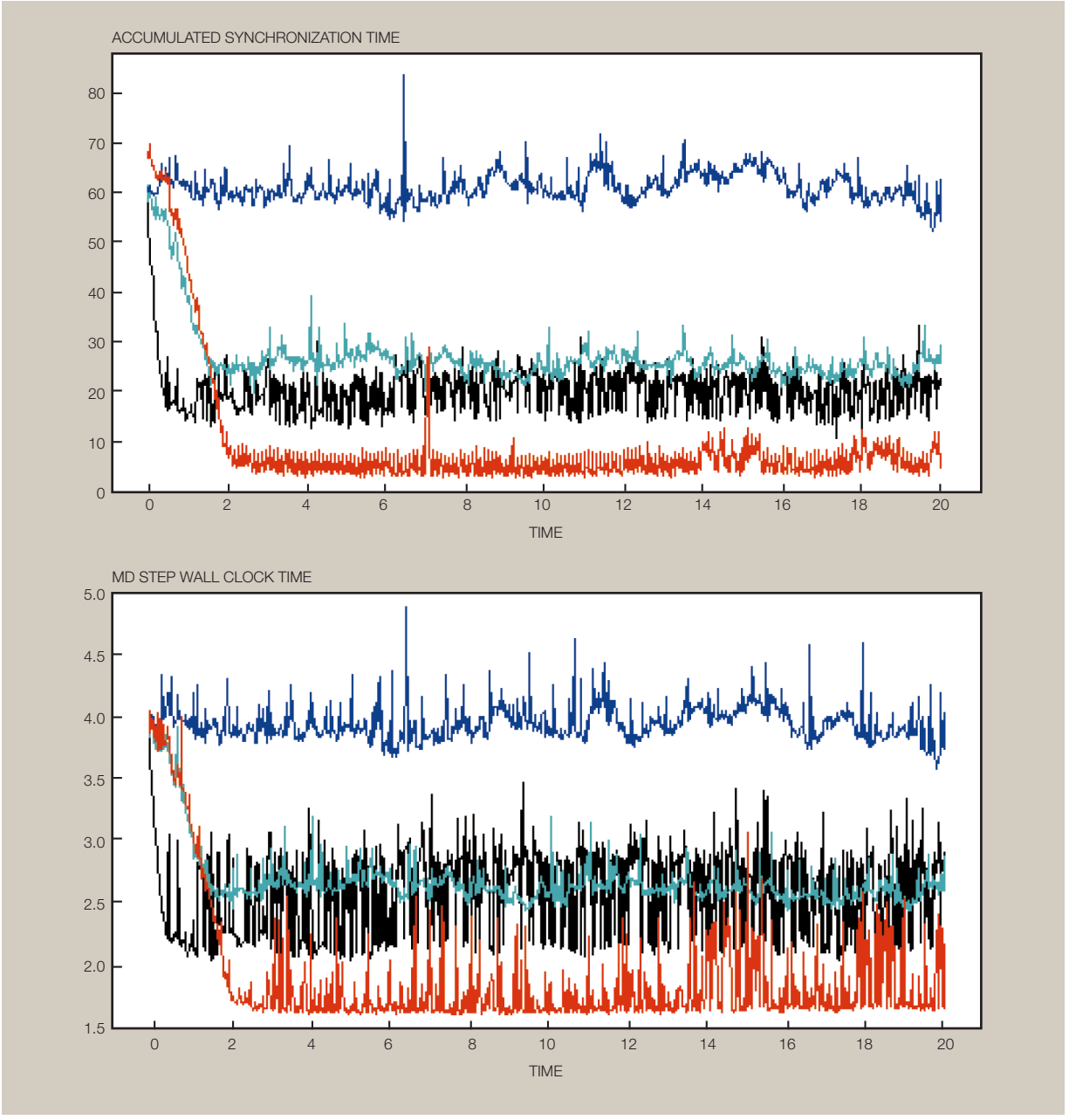


Ewald technique to account for electrostatic interactions beyond the cutoff radius, at a constant temperature of 298 K (Kelvin) and constant pressure of 1 atmosphere. Included in the load balancing is the recording of the coordinates of the system to a trajectory file, every ten molecular dynamics steps.

In Figures 5, 6, and 7, the measured accumulated synchronization times and total wall-clock times for a single molecular dynamics step in seconds versus simulation time in picoseconds (ps) are given. The times are for a 10000-step molecular dynamics simulation of solvated haloalkanedehologenase performed on 27 processors. They are shown for simulations without load balancing (blue curves), load balancing based on subbox redistribution only (green curves), subbox pair resizing only (black curves), and the combination of these (red curves). Figure 5 illustrates the times for the Molecular Science Computing Facility (MSCF) IBM SP with P2SC processors at the Pacific Northwest National Laboratory, and Figure 6 illustrates them for the MSCF experimental IBM SP with silver nodes. Figure 7 illustrates the times for the National Energy Research Scientific Computing Center (NERSC) Cray T3E-900 at the Lawrence Berkeley National Laboratory. In the simulations using the combined load balancing, subbox resizing was applied only after 10 successive subbox pair redistributions did not reduce the accumulated synchronization time.

On all three machines, the two load-balancing methods reduce the accumulated synchronization time to less than half the time measured for the simulations without load balancing. The combination of the two methods reduces the accumulated synchronization time by an order of magnitude, resulting in the re-

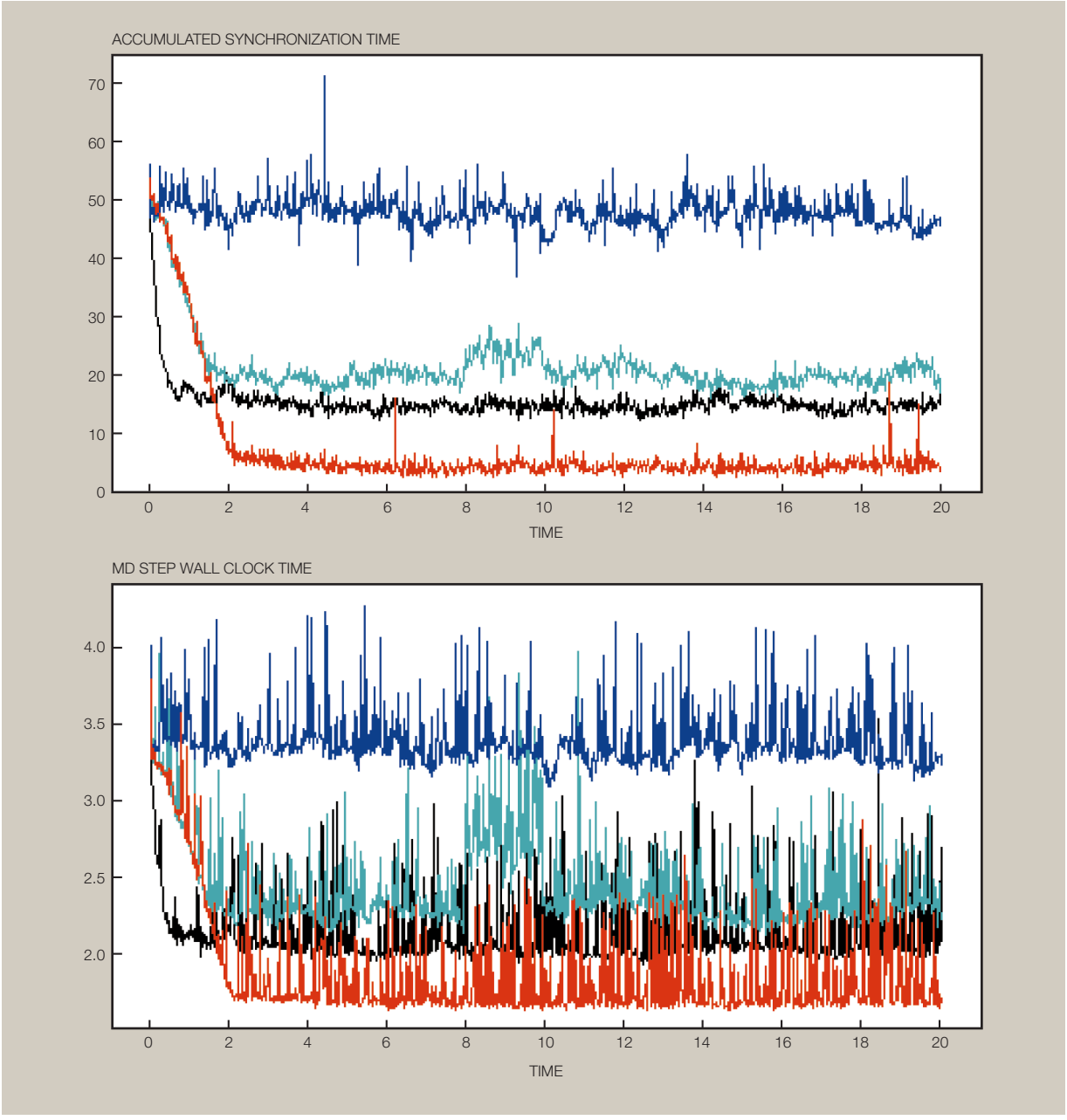
Figure 5 Times for molecular dynamics simulation performed on 27 processors of the IBM SP



duction of the wall-clock time per molecular dynamics step by a factor of two.

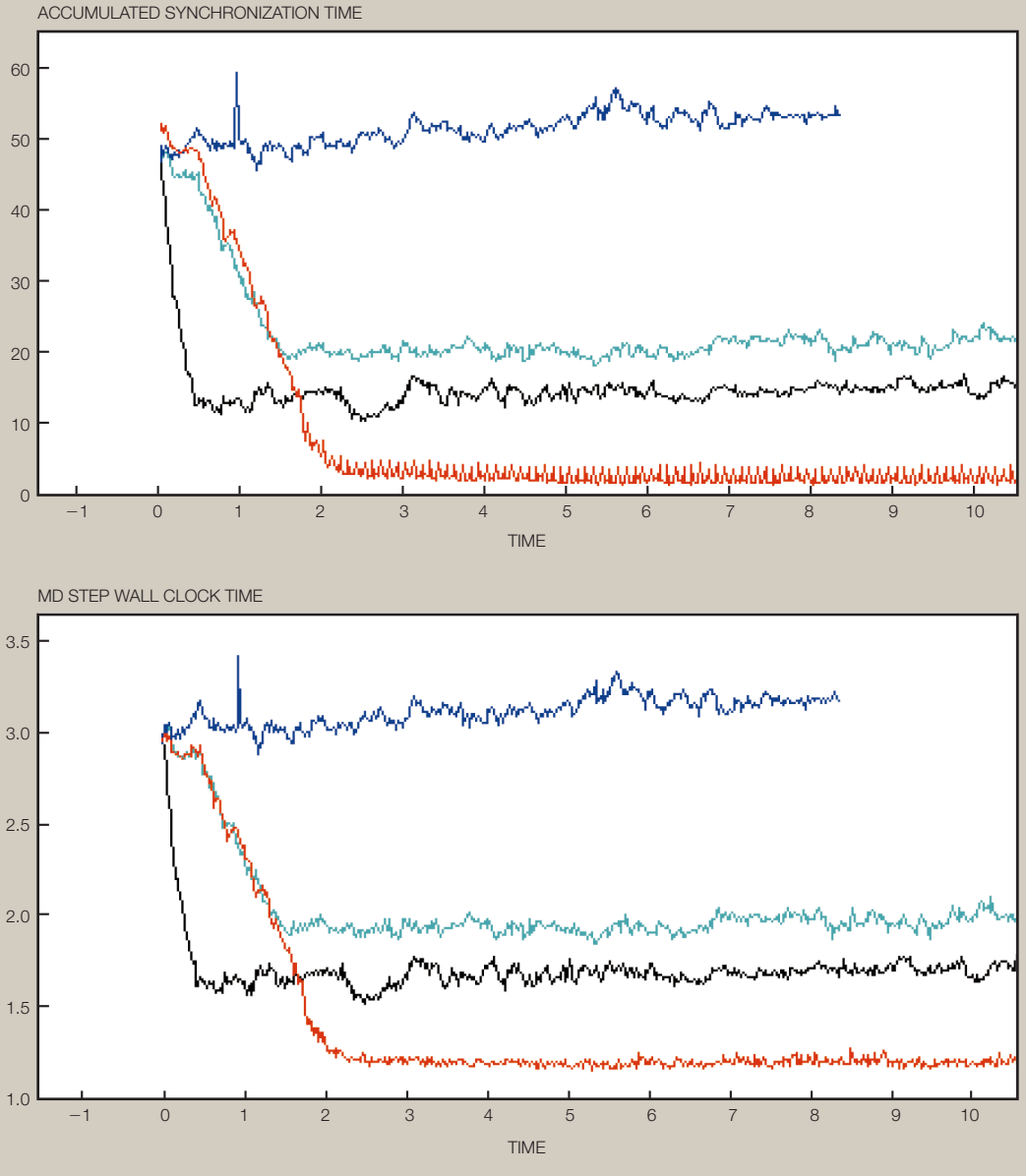
The fluctuations in the measured accumulated synchronization time presented a particular challenge in the design of the load-balancing algorithms, especially for the IBM SPs. Such fluctuations are expected for computer systems with higher latency and lower bandwidth, increasing the chances for contention. Another possible cause is the need for resources

Figure 6 Times for molecular dynamics simulation performed on 27 silver processors of the IBM experimental computing system



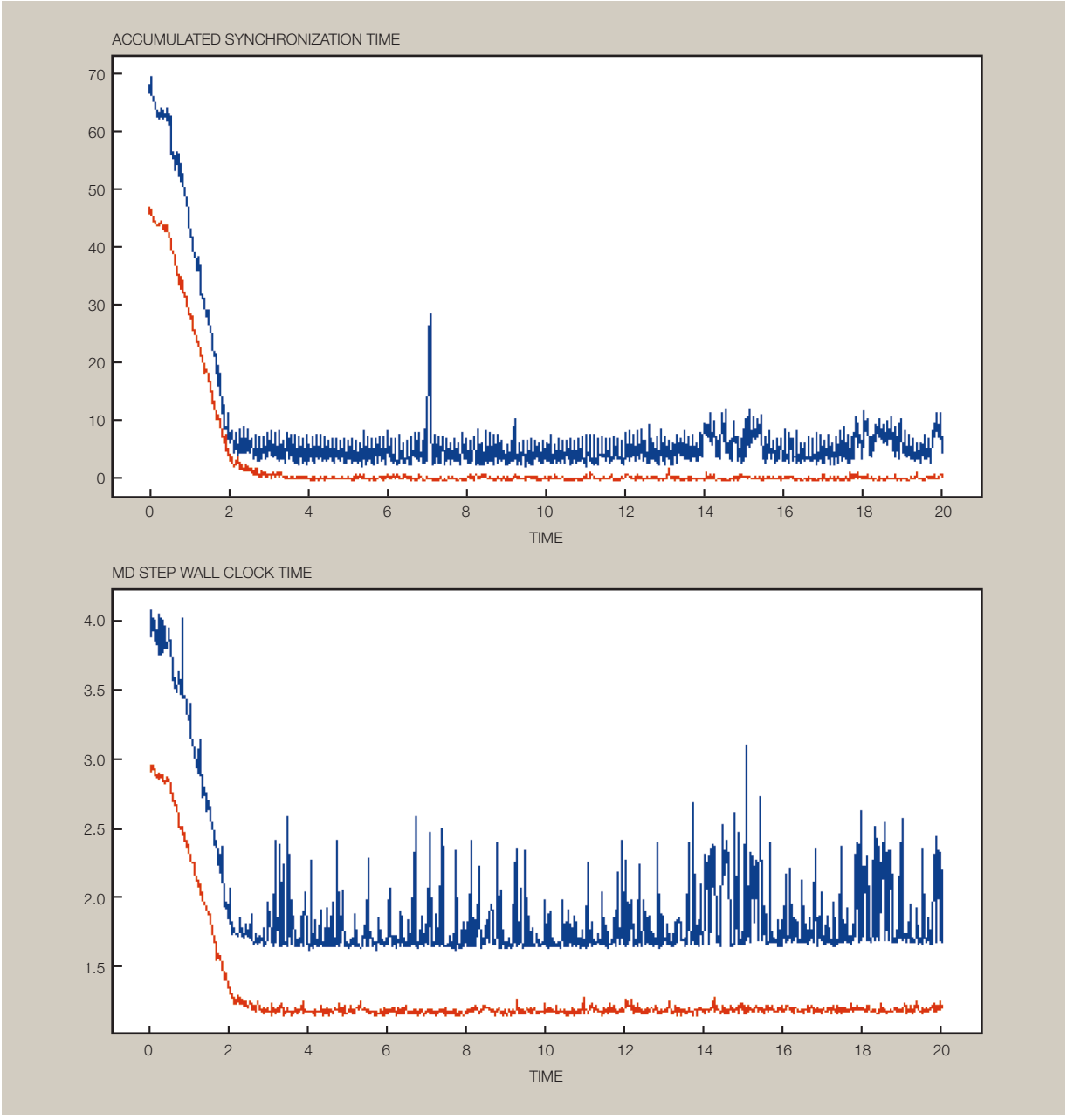
by deamons run by the operating system. This is especially problematic for molecular dynamics simulations, since the wall-clock time between synchronizations is usually very short. In the original implementation of the load-balancing methods, the fluctuation in the accumulated synchronization time was too large for the dynamic load balancing to be effective on the IBM SP. In the current implementation, the dynamic load-balancing algorithms use the minimum or the average accumulated synchroniza-

Figure 7 Times for molecular dynamics simulation performed on 27 processors of the Cray T3E-900



tion time over a number of time steps. Even with the averaging used in this study, the difference in these fluctuations between the IBM SP and Cray T3E are significant, as can be seen from Figure 8. It shows the accumulated synchronization time and wall-clock time for a single molecular dynamics step in seconds versus simulation time in ps for a 10000-step molecular dynamics simulation of solvated haloalkane-dehalogenase performed on 27 processors of the Cray T3E-900 (red) and IBM SP (blue), with com-

Figure 8 Times for molecular dynamics simulation performed on 27 processors of the Cray T3E-900 and IBM SP



bined subbox pair redistribution and subbox resizing.

In Table 2 the accumulated time for each component of the force calculation with combined load balancing is given, with the total wall-clock time per molecular dynamics step and the efficiency of the dynamic load balancing, which is around 0.9 or better. The components without communication, such as the calculation of local or remote forces, are

Table 2 Accumulated wall-clock time in seconds for different components

	Cray T3E	IBM SP P2SC	IBM SP 604
Replicated data interactions	0.000	0.001	0.002
PME charge grid calculation	1.969	4.561	3.486
PME subset synchronization 1	0.565	2.141	1.138
Reverse FFT	0.629	1.089	2.260
PME reciprocal work	0.112	0.294	0.215
PME subset synchronization 2	0.022	0.110	0.172
Forward FFT	0.608	1.167	2.235
I/O	0.034	0.058	0.025
Local coordinate retrieval	0.011	0.020	0.022
Local force calculation	13.418	17.718	14.122
Local force accumulation	0.009	0.018	0.019
Remote coordinate retrieval	0.250	0.956	1.384
Remote force calculation	10.467	14.560	10.858
Remote force accumulation	1.215	1.028	2.283
PME wait	0.002	0.002	0.002
PME forces calculation	1.111	2.749	4.638
Synchronization	2.359	6.814	4.653
MD step total	1.301	2.099	1.934
Load balancing efficiency	0.942	0.893	0.920

slightly faster on the Cray T3E. All communication is performed using the Global Array toolkit, and the observed differences in timings for data retrieval and remote data accumulation on the IBM SP and Cray T3E are consistent with reported Global Array performance on those machines.[23] Communication on the IBM SP 604 is less efficient than on the IBM SP P2SC. Since the parallel Fast Fourier Transform includes communication, its timing on the 604 processors is longer than on the P2SC.

Conclusion

The efficiency of molecular dynamics simulations on massively parallel computers depends critically on the availability of efficient dynamic load-balancing techniques. The molecular dynamics module of NWChem is based on a domain decomposition approach in which all remote data access during the evaluation of atomic forces is performed by one-sided asynchronous communication operations. This approach allows the dynamic load balancing to be determined by the measured synchronization time at the end of the force evaluation. A combination of redistribution of calculation of forces from interprocessor subbox pair interactions and resizing of processor domains is shown to lead to much improved efficiency of molecular dynamics simulations.

Acknowledgment

This work was supported under the auspices of the U.S. Department of Energy (DoE) Office of Biological and Environmental Research (OBER), which funds the Environmental Molecular Sciences Laboratory (EMSL). Pacific Northwest National Laboratory is a multidisciplinary laboratory operated by Battelle for the U.S. Department of Energy. Computer time on the IBM SPs was provided by the Molecular Sciences Computing Facility at EMSL and on the Cray T3E by the National Energy Research Scientific Computing Center (NERSC). J. A. McCammon's work is supported in part by a grant from the National Science Foundation (NSF).

Cited references

1. W. F. van Gunsteren and H. J. C. Berendsen, "Computer Simulation of Molecular Dynamics: Methodology, Applications and Perspectives in Chemistry," *Angewandte Chemie International Edition in English* **29**, 992–1023 (1990).
2. A. R. C. Raine, D. Fincham, and W. Smith, "Sistolic Loop Methods for Molecular Simulation Using Multiple Transputers," *Computer Physics Communications* **55**, 13–30 (1989).
3. J. E. Mertz, D. J. Tobias, C. L. Brooks, and U. C. Singh, "Vector and Parallel Algorithms for the Molecular Dynamics Sim-

ulation of Macromolecules on Shared Memory Computers," *Journal of Computational Chemistry* **12**, 1270–1277 (1991).
4. S. L. Lin, J. Mellor-Crummey, B. M. Pettitt, and G. N. Phillips, "Molecular Dynamics on a Distributed Memory Multiprocessor," *Journal of Computational Chemistry* **13**, 1022–1035 (1992).
5. S. E. DeBolt and P. A. Kollman, "(AMBERCUBE) (MD), Parallelization of (AMBER)'s Molecular Dynamics Module for Distributed-Memory Hypercube Computers," *Journal of Computational Chemistry* **14**, 312–329 (1993).
6. Z. Fang, A. D. J. Haymet, W. Shinoda, and S. Okazaki, "Parallel Molecular Dynamics Simulation: Implementation of PVM for a Lipid Membrane," *Computer Physics Communications* **116**, No. 2–3, 295–310 (February 1999).
7. S. Plimpton and B. Hendrickson, "A New Parallel Method for Molecular Dynamics Simulation of Macromolecular Systems," *Journal of Computational Chemistry* **17**, 326–337 (February 1996).
8. S. Y. Liem, D. Brown, and J. H. R. Clarke, "Molecular Dynamics Simulations on Distributed Memory Machines," *Computer Physics Communications* **67**, 261–267 (1991).
9. D. Brown, J. H. R. Clarke, M. Okuda, and T. Yamazaki, "A Domain Decomposition Parallelization Strategy for Molecular Dynamics Simulations on Distributed Memory Machines," *Computer Physics Communications* **74**, 67–80 (1993).
10. T. W. Clark, R. v. Hanxleden, J. A. McCammon, and L. R. Scott, "Parallelizing Molecular Dynamics Using Spatial Decomposition," *Proceedings of the Scalable High Performance Computing Conference*, Knoxville, TN (May 1994), pp. 95–102.
11. P. Ballestrero, P. Baglietto, and V. Ruggiero, "Molecular Dynamics for Proteins: Performance Evaluation on Massively Parallel Computers Based on Mesh Networks Using a Space Decomposition Approach," *Journal of Computational Chemistry* **17**, No. 4, 469–475 (1996).
12. T. P. Straatsma, "NWChem Molecular Dynamics Simulation," *Proceedings of High Performance Computing Systems and Applications* (1998), pp. 231–239.
13. T. P. Straatsma, M. Philippopoulos, and J. A. McCammon, "NWChem: Exploiting Parallelism in Molecular Simulations," *Computer Physics Communications* **128**, No. 1–2, 377–385 (2000).
14. J. Anchell, E. Apra, D. Bernholdt, P. Borowski, T. W. Clark, D. Clerc, H. Dachsel, M. O. Deegan, M. Dupuis, K. Dyall, G. I. Fann, H. Fruchtl, M. Gutowski, R. J. Harrison, A. Hess, J. Jae, R. A. Kendall, R. Kobayashi, R. Kutteh, Z. Lin, R. Littlefield, X. Long, B. Meng, J. A. Nichols, J. Nieplocha, A. Rendall, M. Stave, T. P. Straatsma, H. Taylor, G. Thomas, K. Wolinski, and A. T. Wong, *NWChem*, *A Computational Chemistry Package for Parallel Computers*, *Version 3.2.1*, Pacific Northwest National Laboratory, Richland, WA 99352-0999 (1998). Further information on NWChem can be found under http://www.emsl.pnl.gov:2080/docs/nwchem.
15. R. A. Kendall, E. Apra, D. E. Bernholdt, E. J. Bylaska, M. Dupuis, G. I. Fann, R. J. Harrison, J. Ju, J. A. Nichols, J. Nieplocha, T. P. Straatsma, T. L. Windus, and A. T. Wong, "High Performance Computational Chemistry: An Overview of NWChem, a Distributed Parallel Application," *Computer Physics Communications* **128**, No. 1–2, 260–283 (2000).
16. J. Nieplocha, R. J. Harrison, and R. J. Littlefield, "Field Global Arrays: A Portable "Shared-Memory" Programming Model for Distributed Memory Computers," *Proceedings of Supercomputing '94* (1994), pp. 340–349. Further information on the global array library can be found under http://www.emsl.pnl.gov:2080/docs/parsoft.
17. G. N. Ramachandran and V. Sasisekharan, "Conformation of Polypeptides and Proteins," *Advances in Protein Chemistry* **23**, 283–437 (1968).
18. A. Amadei, A. B. M. Linssen, and H. J. C. Berendsen, "Essential Dynamics of Proteins," *Proteins: Structure, Function, and Genetics* **17**, 412–425 (1993).
19. D. M. F. van Aalten, B. L. de Groot, J. B. C. Findlay, H. J. C. Berendsen, and A. Amadei, "A Comparison of Techniques for Calculating Protein Essential Dynamics," *Journal of Computational Chemistry* **18**, 169–181 (1997).
20. U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, T. H. Lee, and L. G. Pedersen, "A Smooth Particle Mesh Ewald Method," *Journal of Chemical Physics* **103**, No. 19, 8577–8593 (November 15, 1995).
21. L. V. Kale, R. Skeel, M. Bhandarkar, R. Brunner, A. Gursoy, N. Krawetz, J. Phillips, A. Shinozaki, K. Varadarajan, and K. Schulten, "NAMD2: Greater Scalability for Parallel Molecular Dynamics," *Journal of Computational Physics* **151**, No. 1, 283–312 (May 1, 1999).
22. L. Nyland, J. Prins, R. H. Yun, J. Hermans, H.-C. Kum, and L. Wang, "Achieving Scalable Parallel Molecular Dynamics Using Dynamic Spatial Domain Decomposition Techniques," *Journal of Parallel and Distributed Computing* **47**, No. 2, 125–138 (December 15, 1997).
23. G. Shah, J. Nieplocha, J. Mirza, C. Kim, R. Harrison, R. Govindaraju, K. Gildea, P. DiNicola, and C. Bender, "Performance and Experience with LAPI, a New High Performance Communication Library for the IBM RS/6000," *Proceedings of the First Merged International Parallel Processing Symposium and Symposium on Parallel and Distributed Processing*, Orlando, FL (1998), pp. 260–266.

Accepted for publication December 29, 2000.

T. P. Straatsma *Molecular Sciences Software Group, Theory, Modeling and Simulation Directorate, Environmental Molecular Sciences Laboratory, Pacific Northwest National Laboratory, P. O. Box 999, Richland, Washington 99352-0999 (electronic mail: tp_straatsma@pnl.gov).*

J. Andrew McCammon *Howard Hughes Medical Institute, University of California at San Diego, La Jolla, California 92093-0365 (electronic mail: jmccammon@ucsd.edu).* Dr. McCammon received his B. A. degree from Pomona College and his Ph.D. degree in chemical physics from Harvard University. He was Assistant Professor (1978–1981) and M. D. Anderson Professor (1981–1994) at the University of Houston. Since 1995, he has been J. E. Mayer Professor of Theoretical Chemistry, Professor of Pharmacology, and Fellow of SDSC, all at the University of California San Diego. His research is in the statistical mechanics of macromolecules and liquids; theory of protein structure, dynamics, and function; and development and application of computer models and simulation methods for molecular systems. His honors include the 1987 Hitchings Award for Innovative Methods in Drug Design from the Burroughs Wellcome Fund, the 1995 Award for Breakthrough Computational Science from the Smithsonian Institution, and Fellowships in the AAAS, the American Physical Society, and the Biophysical Society.