ELSEVIER

DL_POLY_2.0: A general-purpose parallel molecular dynamics simulation package

W. Smith and T.R. Forester

Theory and Computational Science, CCLRC Daresbury Laboratory, Daresbury, Warrington, U.K.

DL_POLY_2.0 is a general-purpose parallel molecular dynamics simulation package developed at Daresbury Laboratory under the auspices of the Council for the Central Laboratory of the Research Councils. Written to support academic research, it has a wide range of applications and is designed to run on a wide range of computers: from single processor workstations to parallel supercomputers. Its structure, functionality, performance, and availability are described.

BACKGROUND

The DL_POLY project originated in the United Kingdom Collaborative Computational Project, widely known as CCP5.[1] The objective was to develop a new molecular simulation package for the U.K. academic community that would exploit the emerging parallel computers to the fullest advantage as well as satisfy a number of other important criteria, particularly the demand for "open" software permitting verification and extension by the user (meaning that the package should be available in the form of source code). In addition, the package is required to support a wide range of applications; for example, macromolecules (both biological and synthetic), complex fluids, and ionic materials of high complexity. The DL_POLY package was the outcome of these requirements.

With support from the Engineering and Physical Sciences Research Council the DL_POLY package has been developed to a stage at which it is beginning to make a significant impact on molecular simulation science in the U.K. The first public version of the code (DL_POLY_1.1) was released in October 1994 and enjoys a rapidly growing user base. January 1996 saw the release of a much enhanced version: DL_POLY_2.0. The package is available free, under licence, to academic institutions on a world-wide basis.

This article describes the structure and capabilities of DL_POLY_2.0 and its public availability. The following section describes its structure and overall design. The subsequent sections describe its functionality and its performance on a number of model systems on a Cray T3D parallel computer. The final section outlines the availability of the package.

STRUCTURE

Overall design

As supplied, DL_POLY_2.0 consists of a single UNIX directory with several subdirectories as follows:

1. *source*—FORTRAN source code
2. *build*—Makefiles to build programs
3. *execute*—Program execution subdirectory
4. *data*—Input and output examples
5. *utility*—Analysis and data preparation programs
6. *public*—Contributed facilities
7. *manuals*—Documentation

The *source* subdirectory consists of approximately 150 subroutines. The basic design philosophy is that the users assemble these into tailor-made programs to meet their own requirements. However, to assist the inexperienced, facilities to build a "standard" version are supplied (in the *build* subdirectory—see below). To assist assembly all the subroutines have been written in a uniform style, in FORTRAN 77. The adopted style has been chosen to improve the legibility of the code and simplify the encoded algorithms, in keeping with the best coding practice. To simplify data transfer between subroutines, there are no COMMON blocks, expect those demanded by certain communication routines in the parallel implementation. An important feature of the subroutines is that they incorporate parallelism automatically (see below); data distribution between processors and the necessary communications are localized in a few dedicated subroutines.

The *build* subdirectory stores the UNIX makefiles that

Address reprint requests to: W. Smith, Theory and Computational Science, CCLRC Daresbury Laboratory, Daresbury, Warrington WA4 4AD, U.K.
Received 20 February 1996; accepted 30 April 1996.

Journal of Molecular Graphics 14:136–141, 1996
© 1996 by Elsevier Science Inc.
655 Avenue of the Americas, New York, NY 10010

S0263-7855/96/$15.00
PII S0263-7855(96)00043-4

assemble the executable versions of the code. The makefiles for several common workstations and parallel computers are available. To use, the required makefile is copied into the *source* subdirectory, where the code is compiled and linked. It is from the *execute* subdirectory that the program is run. Sample UNIX scripts are supplied to show how this is done on various machines. The input data for a given job are located in this subdirectory at run time. The output is written here also.

The *data* subdirectory contains examples of input and output from DL_POLY_2.0. These constitute the DL_POLY_2.0 benchmarks and test cases. This subdirectory is the recommended storage area for all data.

DL_POLY_2.0 requires a substantial arsenal of analysis and data preparation tools to be used effectively. Those written by the DL_POLY_2.0 authors are stored in the *utility* directory. Also, since the provision of open software is the aim, a second subdirectory, *public*, stores utilities donated by users. (Because of their origin the routines in the *public* subdirectory are not supported by the DL_POLY_2.0 authors.)

The final subdirectory is *manuals*, which houses the DL_POLY_2.0 documentation. The documentation takes the form of two manuals: a user manual and a reference manual. The former is a guide to using the code in scientific work, the latter is a much more detailed description suitable for software developers.

Parallelization

A special feature of DL_POLY_2.0 is parallelism. It was designed to run on *distributed memory* parallel computers (e.g., Cray T3D, Intel iPSC/860, and IBM SP2) and incorporates the necessary data distribution and communication structures. It is important to note, however, that the code can also be run on *serial* computers without modification.

The parallel strategy underlying DL_POLY_2.0 is the *Replicated Data* (RD) scheme.[2,3] In this scheme the fundamental data of the simulated system are reproduced on all processing nodes. (Auxiliary data, however, are fully distributed.) In molecular dynamics, the fundamental data consist of atomic position, velocity, and force arrays, together with data defining the force field and molecular topology. This volume of data is by no means prohibitive on current parallel computers. Our experience has shown the RD scheme to be perfectly viable on as many as 256 nodes (see Performance, below).

The RD scheme permits a simple communication strategy. Much of the communication between nodes is handled by global summation routines, which are commonplace on current computers. The additional communications are embedded within dedicated subroutines and are easily adapted for different parallel computers. The porting is greatly facilitated by communication harnesses such as PVM (Parallel Virtual Machine) and MPI (Message-Passing Interface), for which DL_POLY_2.0 contains the appropriate calling sequences. It also supports calls to native communication procedures on the Cray T3D and Intel iPSC/860 computers. Any of these options, including serial computers, can be obtained by selection of appropriate flags at compile time.

FUNCTIONALITY

Molecular systems

DL_POLY_2.0 incorporates most, if not all, of the functions required from a modern molecular simulation package, with the added bonus of an in-built parallel capability. Its potential range of applications is vast: from biological systems and polymeric materials, through to inorganic materials and solutions. An idea of the range of its applications can be obtained from the following list, which represents only a small fraction of the possibilities:

1. Atomic and molecular liquids and solids, e.g., Ar, SF_6, H_2O, etc.
2. Inorganic materials and melts, e.g., MgO, KNO_3, TiO_2, etc.
3. Electrolyte and polyelectrolyte solutions, e.g. PEO, DNA, etc.
4. Synthetic polymers, e.g., polyethylene, styrene, etc.
5. Biopolymers in crystals and solution, e.g., proteins, etc.
6. Surfactants, interfaces, and membranes
7. Silicate glasses and zeolites
8. Simple metals and alloys, e.g., Al, Ag, Ni, Cu, etc.
9. Polarizable systems, e.g., ceramics, H_2O, etc.

Within these broad categories, a wide selection of molecular models is possible. DL_POLY_2.0 offers a choice of flexible, rigid, or partially rigid molecules. Flexible molecules are described in terms of extensible bonds and variable bond angles (see the force field description below). Rigid molecular units are described by fixed geometry assemblies of atoms. Partially rigid molecules can be constructed from standard bond constraints[4] or by rigid units linked by flexible or constrained bonds. The dynamic treatment of these different models requires markedly different techniques (see below).

In the case of systems with Coulombic interactions, the charged entities are atoms, modeled either as point charges or as polarizable cores and shells.[5] The current code does not support the use of point multipoles. The simulated system may be treated with periodic or nonperiodic boundary conditions. Finer details of the models may be gleaned from the following section describing the force field.

Force field

A force field is a set of empirical functions and associated parameters that collectively describe the interactions between atoms and molecules.

DL_POLY_2.0 is not supplied with a parameterized force field of its own. In this respect it differs from other academic packages such as GROMOS[6] and AMBER.[7] However, it is sufficiently flexible to accommodate many of the functional forms of these force fields and leaves entirely open the possibility of the users implementing a force field of their own design. This flexibility is ensured by the generality of its subroutine design and the open access to the source code.

The following features are included:

1. All common forms of nonbonded atom–atom potential
2. Coulombic point charge and core–shell potentials
3. Valence angle potentials

4. Dihedral and improper dihedral angle potentials
5. Density dependent metal potentials
6. Three-body (angular) potentials

The number of forms of the nonbonded potentials supported by DL_POLY_2.0 is wide and covers all the commonly encountered forms (Lennard–Jones, n-m, Buckingham, Born–Huggins–Meyer, etc.). However, the code has been designed to make the incorporation of new forms particularly simple. New potentials are defined by FORTRAN statement functions that are used internally to construct interpolation arrays. This permits the existing force calculation routines to be used in all such calculations. The ease of incorporation of new potential terms is common to all aspects of the DL_POLY_2.0 force field. The forms available in DL_POLY_2.0 are as follows:

1. 12–6 potential:

$$U(r_{ij}) = \left\{\frac{A}{r_{ij}^{12}}\right\} - \left\{\frac{B}{r_{ij}^{6}}\right\} \tag{1}$$

2. Lennard–Jones:

$$U(r_{ij}) = 4\epsilon \left\{\left\{\frac{\sigma}{r_{ij}}\right\}^{12} - \left\{\frac{\sigma}{r_{ij}}\right\}^{6}\right\} \tag{2}$$

3. n–m potential:

$$U(r_{ij}) = \frac{E_0}{(n-m)}\left\{m\left\{\frac{r_0}{r_{ij}}\right\}^{n} - n\left\{\frac{r_0}{r_{ij}}\right\}^{m}\right\} \tag{3}$$

4. Buckingham potential:

$$U(r_{ij}) = A\exp\left(-\frac{r_{ij}}{\rho}\right) - \frac{C}{r_{ij}^{6}} \tag{4}$$

5. Born–Huggins–Meyer potential:

$$U(r_{ij}) = A\exp(B\{\sigma - r_{ij}\}) - \frac{C}{r_{ij}^{6}} - \frac{D}{r_{ij}^{8}} \tag{5}$$

6. Hydrogen-bond (12–10) potential:

$$U(r_{ij}) = \left\{\frac{A}{r_{ij}^{12}}\right\} - \left\{\frac{B}{r_{ij}^{10}}\right\} \tag{6}$$

7. Shifted force n–m potential[8]:

$$\begin{aligned} U(r_{ij}) = \frac{\alpha E_0}{(n-m)}\Bigg[& m\beta^{n}\left\{\left(\frac{r_0}{r_{ij}}\right)^{n} - \left(\frac{1}{\gamma}\right)^{n}\right\} \\ & - n\beta^{m}\left\{\left(\frac{r_0}{r_{ij}}\right)^{m} - \left(\frac{1}{\gamma}\right)^{m}\right\}\Bigg] \\ & + \frac{nm\alpha E_0}{(n-m)}\left(\frac{r_{ij} - \gamma r_0}{\gamma r_0}\right)\left\{\left(\frac{\beta}{\gamma}\right)^{n} - \left(\frac{B}{\gamma}\right)^{m}\right\} \end{aligned} \tag{7}$$

with

$$\gamma = \frac{r_{\mathrm{cut}}}{r_0} \tag{8}$$

$$\beta = \gamma\left(\frac{\gamma^{m+1} - 1}{\gamma^{n+1} - 1}\right)^{\frac{1}{n-m}} \tag{9}$$

$$\alpha = \frac{(n-m)}{\begin{aligned}&[n\beta^{m}(1 + (m/\gamma - m - 1)/\gamma^{m}) \\ &- m\beta^{n}(1 + (n/\gamma - n - 1)/\gamma^{n})]\end{aligned}} \tag{10}$$

DL_POLY_2.0 offers a wide choice of techniques for handling long-range electrostatic interactions. The preferred technique for periodic systems is Ewald's method,[9] but in other circumstances direct sum methods are suitable. The reaction field method[9,10] is an inexpensive alternative for periodic systems and is suitable for nonperiodic systems. The charge group method and the use of direct Coulombic sum with a distance-dependent dielectric are common in simulations of biological systems. DL_POLY_2.0 contains these options and also the less rigorous truncated Coulomb sum methods.

The force field in DL_POLY_2.0 is defined with reference to the atoms comprising the system. Molecules, however, imply the presence of both *intra-* and *inter*molecular interactions. It is usual (but not universal) for the nonbonded interactions between atoms in the same bond, valence angle, or dihedral terms to be neglected. DL_POLY_2.0 incorporates strategies to handle these exclusions (or sometimes partial exclusions) in accordance with the user's needs. In the case of electrostatic interactions, the appropriate corrections to the Ewald sum are automatically included.

The valence, dihedral, and improper dihedral angle potentials define the conformational structure of molecules. DL_POLY_2.0 supports a broad selection of empirical forms suitable for simple molecules, polymers, and zeolites. The three-body potentials available are angular functions, primarily for simulating silicate glasses,[11] but may be extended for other glasses and materials. The full set of angular potentials in DL_POLY_2.0 is as follows:

1. Harmonic:

$$U(\theta_{jik}) = \frac{k}{2}(\theta_{jik} - \theta_0)^2 \tag{11}$$

2. Quartic:

$$U(\theta_{jik}) = \frac{k}{2}(\theta_{jik} - \theta_0)^2 + \frac{k'}{3}(\theta_{jik} - \theta_0)^3 + \frac{k''}{4}(\theta_{jik} - \theta_0)^4 \tag{12}$$

3. Truncated harmonic:

$$U(\theta_{jik}) = \frac{k}{2}(\theta_{jik} - \theta_0)^2\exp[-(r_{ij}^8 + r_{ik}^8)/\rho^8] \tag{13}$$

4. Screened harmonic:

$$U(\theta_{jik}) = \frac{k}{2}(\theta_{jik} - \theta_0)^2\exp[-(r_{ij}/\rho_1 + r_{ik}/\rho_2)] \tag{14}$$

5. Screened Vessal[12]:

$$\begin{aligned} U(\theta_{jik}) = & \frac{k}{8(\theta_0 - \pi)^2}\{[(\theta_0 - \pi)^2 - (\theta_{jik} - \pi)^2]^2\} \\ & \exp[-(r_{ij}/\rho_1 + r_{ik}/\rho_2)] \end{aligned} \tag{15}$$

6. Truncated Vessal[13]:

$$U(\theta_{jik}) = k\left[\theta_{jik}^{\alpha}(\theta_{jik} - \theta_0)^2(\theta_{jik} + \theta_0 - 2\pi)^2 - \frac{\alpha}{2}\pi^{\alpha-1}\right.$$
$$\left.(\theta_{jik} - \theta_0)^2(\pi - \theta_0)^3\right]\exp[-(r_{ij}^8 + r_{ik}^8)/\rho^8] \quad (16)$$

7. Cosine dihedral potential:

$$U(\phi_{ijkn}) = A[1 + \cos(m\phi_{ijkn} - \delta)] \quad (17)$$

8. Harmonic improper dihedral:

$$U(\phi_{ijkn}) = \frac{1}{2}k(\phi_{ijkn} - \phi_0)^2 \quad (18)$$

The density-dependent potentials in DL_POLY_2.0 are those of Sutton and Chen,[14] which in turn are based on the model of Finnis and Sinclair.[15] These forms are highly suitable for simulating metals and alloys.

$$U_{sc} = \epsilon\left\{\sum_{j}\sum_{i<j}\left(\frac{\alpha}{r_{ij}}\right)^{n} - C\sum_{i}\rho_i^{1/2}\right\} \quad (19)$$

where the *local density* ρ_i is given by

$$\rho_i = \sum_{i}\left(\frac{\alpha}{r_{ij}}\right)^{m} \quad (20)$$

It is important to note that in DL_POLY_2.0 all pair interactions are calculated using a neighbor list,[9] which keeps record of the interactions within a spherical cutoff of a given "central" atom. This list is updated at intervals as a simulation proceeds. The DL_POLY_2.0 parallel implementation distributes the list over processing nodes.[2]

Boundary conditions

A choice of boundary conditions is an important feature of any simulation package. DL_POLY_2.0 will permit the following boundary conditions:

1. None, e.g., isolated biopolymer in space
2. Cubic periodic boundaries[9]
3. Orthorhombic periodic boundaries[9]
4. Parallelepiped periodic boundaries[9]
5. Truncated octahedral periodic boundaries[16]
6. Rhombic dodecahedral periodic boundaries[16]
7. Slab (x,y periodic, z nonperiodic)

Solvents

As with the force field, DL_POLY_2.0 has no default model solvent. The user defines the required solvent model as another molecular species in the system of interest. In the case of aqueous solutions, most of the commonly used water models are acceptable, provided the electrostatic model does not depend on a point–multipole representation. Both rigid and flexible water molecules are permitted. Thus long-established water models such as SPC,[17] ST2,[18] and MCY[19] can be handled. The availability of polarizability and three-body forces within DL_POLY_2.0 offers a facility to extend beyond these models.

Molecular dynamics algorithms

DL_POLY_2.0 offers a variety of molecular dynamics simulation algorithms, which differ in terms of their dynamic and thermodynamic properties. All the algorithms, however, are based on the leapfrog variant of the Verlet scheme.[9]

Different algorithms are required to simulate different molecular models. Flexible molecules and unbonded atoms are treated with the basic leapfrog algorithm. Molecules defined with bond constraints alone are handled by a parallel variant of the SHAKE algorithm[4] known as RD-SHAKE.[3] Rigid molecules and rigid units connected by extensible bonds are handled with Fincham's implicit quaternion algorithm (FIQA).[20] Rigid units connected by rigid bonds are handled by a new algorithm (QSHAKE) devised especially for DL_POLY_2.0.[21]

Algorithms differ in regard to the thermodynamic *ensemble* they represent. Strictly speaking, not all algorithms represent valid ensembles, but are often labeled by the most closely related ensemble. In DL_POLY_2.0 the ensembles available include the following:

1. NVE—Fixed particle number, volume, and energy (the microcanonical ensemble)
2. NVT—Fixed particle number, volume, and temperature (the canonical ensemble)
3. NPT—Fixed particle number, pressure, and temperature
4. N$\underline{\underline{\sigma}}$T—Fixed particle number, stress ($\underline{\underline{\sigma}}$), and temperature

In practice the ensembles are distinguishable by the use of a thermostat (conserving temperature) or a barostat (conserving pressure). Choosing one or both is what determines the thermodynamic identity of the ensemble.

Within DL_POLY_2.0 the following algorithms are available:

NVE_0	Verlet leapfrog
NVE_1	Verlet leapfrog with RD-SHAKE
NVEQ_1	Rigid units with FIQA and RD-SHAKE
NVEQ_2	Linked rigid units with QSHAKE
NVT_B0	Constant T (Berendsen et al.[22]) with Verlet leapfrog
NVT_B1	Constant T (Berendsen et al.[22]) with RD-SHAKE
NVT_E0	Constant T (Evans and Morriss[23]) with Verlet leapfrog
NVT_E1	Constant T (Evans and Morriss[23]) with RD-SHAKE
NVT_H0	Constant T (Hoover[24]) with Verlet leapfrog
NVT_H1	Constant T (Hoover[24]) with RD-SHAKE
NVTQ_B1	Constant T (Berendsen et al.[22]) with FIQA and RD-SHAKE
NVTQ_B2	Constant T (Berendsen et al.[22]) with QSHAKE
NVTQ_H1	Constant T (Hoover[24]) with FIQA and RD-SHAKE
NVTQ_H2	Constant T (Hoover[24]) with QSHAKE
NPT_B1	Constant T, P (Berendsen et al.[22]) with FIQA and RD-SHAKE
NPT_B3	Constant T, $\underline{\underline{\sigma}}$ (Berendsen et al.[22]) with RD-SHAKE
NPT_H1	Constant T, P (Hoover[24]) with RD-SHAKE

NPT_H3	Constant T, $\underline{\underline{\sigma}}$ (Hoover[24]) with RD-SHAKE
NPTQ_B1	Constant T, P (Berendsen et al.[22]) with FIQA and RD-SHAKE
NPTQ_B2	Constant T, P (Berendsen et al.[22]) with QSHAKE
NPTQ_B3	Constant T, $\underline{\underline{\sigma}}$ (Berendsen et al.[22]) with FIQA and RD-SHAKE
NPTQ_B4	Constant T, $\underline{\underline{\sigma}}$ (Berendsen et al.[22]) with QSHAKE
NPTQ_H1	Constant T, P (Hoover[24]) with FIQA and RD-SHAKE
NPTQ_H2	Constant T, P (Hoover[24]) with QSHAKE
NPTQ_H3	Constant T, $\underline{\underline{\sigma}}$ (Hoover[24]) with FIQA and RD-SHAKE
NPTQ_H4	Constant T, $\underline{\underline{\sigma}}$ (Hoover[24]) with QSHAKE

DL_POLY_2.0 also provides a multiple timestep algorithm for economical simulation of large systems. The scheme requires two cutoff radii; the primary and secondary cutoffs. The smaller primary radius specifies the range within which interactions are explicitly calculated *every timestep.* For distances larger than this, and smaller than the secondary cutoff, the interactions are calculated less frequently, with a consequent saving in execution time. At intervals between the explicit calculations of these longer ranged interactions, a numerical extrapolation from the first two timesteps of the interval is used to estimate the long-range interaction. The algorithm, due to Tildesley et al.[25,26] has been extended to treat Coulombic systems.[27]

PERFORMANCE

In this section we outline the performance of DL_POLY_2.0, with reference to particular molecular systems on a Cray T3D parallel computer. The test cases are composed of the following systems.

1. A total of 19 652 Lennard–Jones atoms with a cubic periodic cell at density $\rho^* = 1$ and with cutoff $r_{\text{cut}} = 8.6$ Å (2.25σ). A link cell algorithm[9] is used to construct the neighbor list.
2. A 15-peptide chain in 1 247 SPC water molecules in a cubic periodic cell, at normal temperature and pressure (NTP) and with $r_{\text{cut}} = 8$ Å. The electrostatic forces are calculated using charge groups. Overall the system is composed of 3 993 atoms and incorporates 1 264 charge groups.
3. The same 15-peptide in water system as previously but with a cutoff of 12 Å applied to the nonbonded potentials and cutoff of 16 Å applied to the electrostatic potential.
4. The protein transferrin in 8 102 TIP3 water molecules in a cubic periodic cell, at NTP, with $r_{\text{cut}} = 8$ Å and charge group electrostatics. The system contains 27 593 atoms and 8 441 charge groups.
5. A total of 16 000 sodium chloride in the NTP crystal, with a cubic periodic cell and a real-space cutoff $r_{\text{cut}} = 48$ Å. The Ewald sum is used to handle the electrostatics, Approximately 348 k vectors are used in reciprocal space (equivalent to 696 k vectors if inversion symmetry is ignored).
6. The same sodium chloride system as in the previous case, but using the multiple timestep algorithm described above,[27] with $r_{\text{prim}} = 10$ Å and $r_{\text{cut}} = 48$ Å and a multistep of 5.

These test systems were run of the Edinburgh Cray T3D, using a version of the code incorporating SHMEM memory-to-memory communications. The results of the simulations, for a nominal 100 timesteps (neglecting the input and output phases of the simulation), are presented in Figure 1. Figure 1 is conveniently plotted as the logarithm (base 10) of the time (T in seconds) to execute 100 timesteps versus the logarithm (base 2) of the number of processing nodes (P). A reduction of 1 vertical unit on any plot corresponds to a factor of 10 increase in computing speed. A straight line indicates good parallel scaling characteristics, while a departure from linearity for larger numbers of processing nodes indicates a growing importance of communication and serial overheads.[28] Typically this becomes significant more rapidly in calculations with a lower compute intensity, for example, cases 1 and 2. The excellent linearity of cases 5 and 6 reflects the high numerical processing demanded by the Ewald sum. Cases 2 and 3, and 5 and 6, show how essentially the same simulation can be achieved much more efficiently (and with little loss in accuracy) if care is taken to select the appropriate algorithms. The most problematic case in this set is case 4, in which the unavoidable communications requirements of the parallel SHAKE algorithm contribute strongly to the imperfect scaling.[3] The success of DL_POLY_2.0 in reducing this burden in cases 2 and 3 is apparent.

Overall, DL_POLY_2.0 is demonstrably able to accommodate a broad range of applications with reasonable efficiency.

AVAILABILITY

DL_POLY_2.0 is available free to academic institutions under licence from Daresbury Laboratory. The Laboratory is the sole center for the distribution of DL_POLY_2.0.

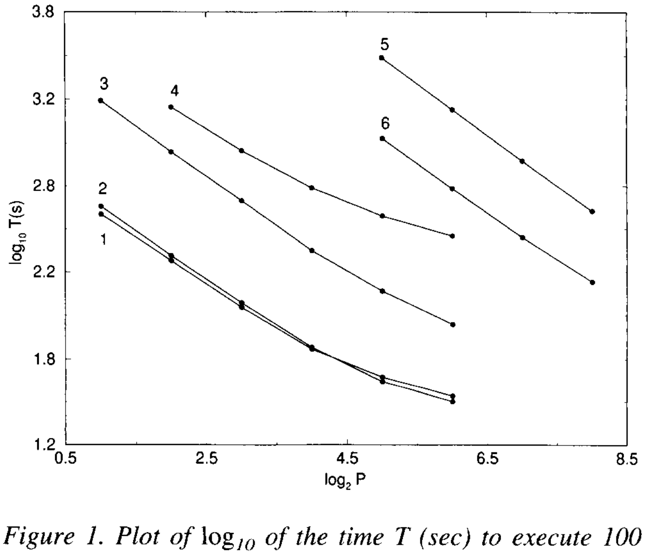

Figure 1. Plot of $\log_{10}$ *of the time T (sec) to execute 100 time steps of DL_POLY_2.0 on test cases 1 to 6 (see text)* versus $\log_2$ *of the number of processing nodes P. (Note the horizontal axis represents the* hypercube dimension *of the parallel computer*[28].

A copy of the licence form may be obtained in two ways: either by selecting the licence button on the World Wide Web page (http://www.dl.ac.uk/TCSC/Software/DL_POLY/main.html) and downloading and printing the file; or by using FTP to copy the postscript file from the CCP5 Program Library at Daresbury Laboratory in the following manner:

→ move to the desired directory on YOUR machine,

→ type: ftp 148.79.80.10 or ftp ftp.dl.ac.uk

→ enter userid: anonymous

→ enter passwd: (use your name and site)

→ change to the CCP5 directory: cd ccp5

→ change to the DL_POLY directory: cd DL_POLY

→ type: binary (for postscript files)

→ type: get LICENCE.ps.Z

→ type: quit

The licence file will need to be uncompressed (using the UNIX *uncompress* command) before printing.

Once the licence form has been signed by the applicant it should be returned to the authors at the above address by post (FAX is not acceptable).

When the signed licence has been received DL_POLY_2.0 source code will be sent by FTP. For this purpose (and this only) the applicant must supply a guest ID and password on the receiving computer. Please note that access to Daresbury computers for this purpose is not available.

These restrictions do not apply to DL_POLY_2.0 *manuals* subdirectory or the *public* utility subdirectory. These subdirectories will be freely available by FTP, in the same manner and location as the licence form. The *DL_POLY_2.0 User Manual* will be available on the World Wide Web at the above address.

Last, there is the question of the availability of DL_POLY_2.0 to commercial organizations. DL_POLY_2.0 is free to academic institutions, but is not freely available to other organizations. All commercial rights of the package are owned by the Council for the Central Laboratories of the Research Councils (U.K.) and commercial organizations wishing to obtain and use the code must first establish an appropriate licencing agreement with CCLRC. Contract should be with the authors in the first instance.

All enquiries regarding DL_POLY_2.0 should be directed to the authors.

ACKNOWLEDGMENTS

This project has been financially supported by grants from the EPSRC's former Science and Materials Computing Committee and the Computational Science Initiative. Daresbury Laboratory is thanked for time on the Intel iPSC/860 and IBM SP2 parallel computers. The Edinburgh Parallel Computing Centre is thanked for assistance in porting DL_POLY to the Cray T3D. The advice and encouragement of the participants in CCP5 and the EPSRC's Materials Consortium is gratefully acknowledged.

REFERENCES

1 Smith, W. *Molecular Graphics*, 1987, **5,** 71

2 Smith, W. and Forester, T.R. *Comput. Phys. Commun.* 1994, **79,** 52

3 Smith, W. and Forester, T.R. *Comput. Phys. Commun.* 1994, **79,** 63

4 Ryckaert, J.P., Ciccotti, G., and Berendsen, H.J.C. *J. Comput. Phys.* 1977, **23,** 327

5 Fincham, D. and Mitchell, P.J. *J. Phys. Condens. Matter.* 1993, **5,** 1031

6 van Gunsteren, W.F. and Berendsen, H.J.C. *Groningen Molecular Simulation (GROMOS) Library Manual.* BIOMOS, Groningen. The Netherlands, 1987 [Standard GROMOS reference]

7 Weiner, S.J., Kollman, P.A., Nguyen, D.T., and Case, D.A. *J. Comput. Chem.* 1986, **7,** 230

8 Clarke, J.H.R., Smith, W., and Woodcock, L.V. *J. Chem. Phys.* 1986, **84,** 2290

9 Allen, M.P. and Tildesley, D.J. *Computer Simulation of Liquids.* Clarendon Press, Oxford, 1989

10 Tironi, I.G., Sperb, R., Smith, P.E., and van Gunsteren, W.F. *J. Chem. Phys.* 1995, **102,** 5451

11 Vessal, B., Amini, M., Leslie, M., and Catlow, C.R.A. Mol. Simul. 1990, **5,** 1

12 Vessal, B. *J. Non-Cryst. Solids.* 1994, **177,** 103

13 Smith, W., Greaves, G.N., and Gillan, M.J. *J. Chem. Phys.* 1995, **103,** 3091

14 Sutton, A.P. and Chen, J. *Philos. Mag. Lett.* 1990, **61,** 139

15 Finnis, M.W. and Sinclair, J.E. *Philos. Mag.* 1984, **50,** 45

16 Smith, W. and Fincham, D. *Mol. Simul.* 1993, **10,** 67

17 Berendsen, H.J.C., Postma, J.P.M., van Gunsteren, W., and Hermans, J. In: Pullman, *Intermolecular Forces* (Pullman, B., ed.). Reidel, Dordecht, 1981, p. 331. [SPC water standard reference]

18 Stillinger, F.H. and Rahman, A. *J. Chem. Phys.* 1974, **60,** 1545

19 Matsuoka, O., Clementi, E., and Yoshimine, M. *J. Chem. Phys.* 1976, **64,** 1351

20 Fincham, D. *Mol. Simul.* 1992, **8,** 165

21 Forester, T. and Smith, W. 1996 (in preparation)

22 Berendsen, H.J.C., Postma, J.P.M., van Gunsteren, W., DiNola, A., and Haak, J.R. *J. Chem. Phys.* 1984, **81,** 3684

23 Evans, D.J. and Morriss, G.P. *Comput. Phys. Rep.* 1984, **1,** 297

24 Hoover, W.G. *Phys. Rev.* 1985, **A31,** 1695

25 Tildesley, D.J., Streett, W.B., and Saville, G. *Mol. Phys.* 1978, **35,** 639

26 Tildesley, D.J. and Streett, W.B. Multiple time step methods and an improved potential function for molecular dynamics simulations of molecular liquids. In: *Computer Modelling of Matter* (Lykos, P., ed.). ACS Symposium Series No. 86. 1978

27 Forester, T. and Smith, W. *Mol. Simul.* 1994, **13,** 195

28 Smith, W. *Comput. Phys. Commun.* 1991, **62,** 229