IMD – A MOLECULAR DYNAMICS PROGRAM AND APPLICATIONS

J. ROTH

Institut für Theoretische und Angewandte Physik, Universität Stuttgart, Pfaffenwaldring 57, 70550 Stuttgart, Germany

E-mail: johannes@itap.physik.uni-stuttgart.de

The intention of this article is two-fold: First, we want to describe IMD (**I**TAP **M**olecular **D**ynamics), a software package for classical molecular dynamics simulations on massively-parallel computers. The software is a general purpose package which can be used for any kind of condensed matter, but it contains a number of special features for simulation of the mechanical properties of solids, especially of quasicrystals and silicon interfaces. Second we want to describe a representative application of IMD to the simulation of shock waves in binary icosahedral quasicrystals. These simulation have been carried out to demonstrate the feasibility of metacomputing (clustering of massively-parallel supercomputers over intercontinental distances).

1 Introduction

IMD is a software package designed for classical molecular dynamics (MD) simulations in two or three dimensions. In MD simulations the hamiltonian equations of motion for a system of point masses are solved iteratively for a given time interval. In the equilibrium case one may then extract thermodynamical averages of the system under study. But MD simulations are not limited to this case: non-equilibrium simulations like heating, cooling, deformation of the sample can also be carried out.

For the basic design of IMD we refer to a previous paper by Jörg Stadler et al.[1] (called paper I in the following).

With the first release 1.0 of IMD it was only possible to simulate particles interacting via pair potentials. Several thermodynamical ensembles were allowed, but the volume of the sample had to be kept constant.

Meanwhile IMD has been extended and improved. The current release 1.2[2] allows the particles to interact via pair, three-body and EAM[a] potentials. The particles can be of an arbitrary number of types and may differ in mass and interaction potential. The interaction potentials are read from tables and therefore are not limited to simple analytical expressions. The calculations can be carried out in a number of thermodynamical ensembles[4] like microcanonical (NVE, total energy is constant), canonical (NVT, temperature and volume constant, NPT, temperature and pressure constant, and NσT, temperature and uniaxial stress constant). For NVT a stochastic Andersen thermostat[4] is available. In general a Nosé-Hoover thermostat is used for NVT, NPT and NσT. In addition to MD simulations it is possible to optimize the potential energy of a sample with the microconvergence method[5]. Mechanical boundary conditions allow the sample to be pulled apart or sheared at a constant rate or at certain time intervals.

IMD contains routines to compute self-correlation functions and to generate

[a] EAM means **E**mbedded **A**tom **M**ethod, first described by Daw and Baskes[3].

several crystal or quasicrystal structures. In addition there are special setups for fracture simulations to damp out the shock waves emitted from a crack, and in the shock wave simulations to initiate a shock wave.

Large simulations on massively-parallel supercomputers require completely new concepts to deal with the overwhelming flood of data: If there are ten to hundred million particles and the simulation is carried out for several hundred thousand or million time steps it is impossible to store the complete particle configurations, even at longer time intervals. IMD includes several features to deal with this problem: It is possible to generate pictures of the potential or kinetic energy directly. The program provides the possibility of online visualization, where the simulation on the supercomputer is monitored during its run on a local workstation (even if there is a firewall in between).

In metacomputing simulations it may not even be possible to generate the initial and final data sets and store them. However, it is often necessary first to optimize the potential energy of the generated structure with the microconvergence method, then to equilibrate it with NPT to correct the volume, and finally to carry out an NVT or NVE simulation to collect the data for the thermodynamical averages. With IMD it is possible to carry out such a multiphase simulation in a single run of one program.

The package's main design goals were to create flexible and modular software that reaches a high performance on contemporary computer architectures, while it is still as portable as reasonably possible. IMDs target platforms are RISC workstations and massively-parallel computers. It does not perform well on vector machines. It is actively supported on DEC and SGI workstations and on PCs running Linux as well as on the Cray T3E. It has been run on IBM workstations, on the Intel Paragon and on the Hitachi SR2201. IMD is written in C and uses the widely available MPI library for message passing. It has been adapted to metacomputing via the PACX[6] library. For the online visualization a special version of VolRend[7] has been adapted.

In paper I the concept of linked cells for simulations of short-range interactions and its the extension to parallel computers has been described. Paper I further deals with communication modes and the performance of IMD in large scale and high speed simulations. This paper is organized as follows: in the first section we will summarize paper I and extend it to the concept of dynamical linked cells and many-body interaction. In the second part we will describe the online visualization and the direct generation of pictures. In the third section we will talk about metacomputing and in the final section present an example, the simulation of shock waves in quasicrystals.

IMD has been applied mainly for simulations of the mechanical stability of quasicrystals. Examples are simulations of cracks, fracture, shear deformations and dislocations in quasicrystals[8,9,10] in two and three dimensions, shock waves in crystals and quasicrystals[11] in three dimensions, and semiconductor interfaces in compound materials.

2 Basic Concepts of Short Range Massively-Parallel MD Simulations

In MD we solve the hamiltonian equations of motion for a system of N particles over a time T_{sim}. The integration is done stepwise in small time intervals Δt. At each time step the force F_i acting on particle i must be calculated. Since the particle may interact with all other particles, the complexity of the problem grows with N^2. Thus the force calculation is the most time consuming part of the simulation. If the forces are short ranged, however, then the complexity for the necessary force calculations grows with NN_c, where N_c is the number of atoms in the range of the interactions. This is true for two-body pair potentials as well as for many-body interactions like EAM for example, since the many-body interactions are usually also short ranged.

For short-range interactions the computation of forces is a local problem and lends itself naturally to parallelization by domain decomposition. The simulation box is geometrically divided into cells which represents an easy way to find the neighbors of any particle.

If the interactions are long-ranged like Coulomb interactions we cannot use domain decomposition. There are other methods like computing part of the interaction in reciprocal space and using Ewald sums, force and atom decomposition, and hierarchical tree methods. We will not discuss these methods here in detail, they are mentioned only since it may not be possible in a straightforward way to extend IMD to such interactions.

The total force F_i acting on particle i is the sum over all forces f_{ij} of the particles j within the interaction range r_c. To calculate F_i we must know the positions r_j of all particles closer than r_c to particle i. To find them we subdivide the simulation box into small cells with a diameter that is at least twice as long as r_c (see Fig. 1). The cells need not be squares or cubes, they may even be parallelepipeds, but they must fill the sample without gaps and overlaps. The atoms can easily be assigned to the cells if their coordinates are divided by the length of the simulation box and then multiplied by the number of boxes. Since the diameter of the cells is $2r_c$, all interactions take place within a cell or between neighboring cells. What remains to be done is to set up a linked list of the neighbors of each cell. Then the whole computation can be carried out by computing the interaction between atoms in the same cell and between atoms in neighboring cells. Since $f_{ij} = -f_{ij}$ by Newton's law only half the number of neighbors has actually to be taken into account (See Fig. 1). Since the assignment of the atoms to the cells is a linear process the complexity of our simulation has been reduced to N.

For NVE and NVT simulations the cell structure is set up at the beginning of the simulation and will never change. This may not be true for NPT or NσT simulations. Since the number of cells is constant it may happen that the diameter of the cells too small if the volume shrinks for example at a transition from a gas to a liquid. Or for the opposite transition it may happen that the number of cells is too small and the simulation becomes very inefficient. IMD checks in NPT and NσT simulations if the cells division is appropriate, and if not, it calculates a new cell setup.

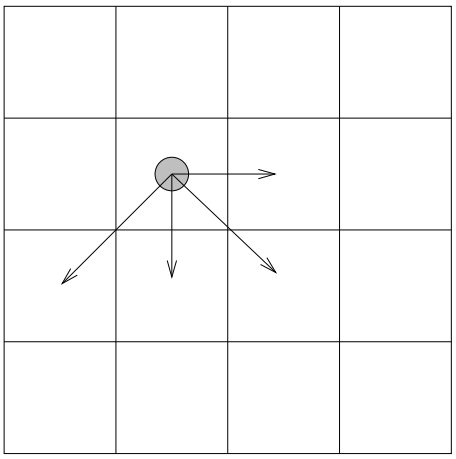

Figure 1. Decomposition of the simulation box into cells. The arrows indicate the neighbor cells that must be taken into account for the compuation of the interaction in the central cell.

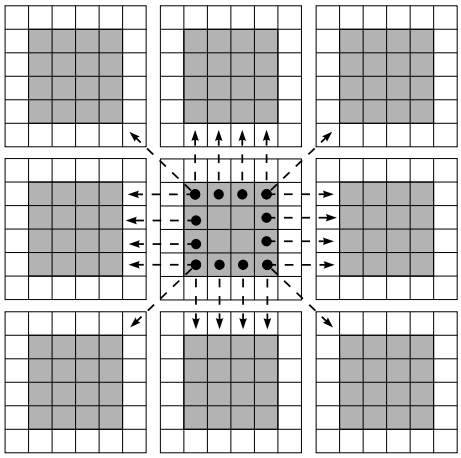

Figure 2. Assignment of the cells to processors in two dimensions. The white cells are *buffer cells*, the adjacent cells (dotted in the central processor) are the *surface cells*.

2.1 Cells on Parallel Computers

The generalization of the cell structure to parallel computers is rather straightforward: IMD assumes that the processors are arranged on a three-dimensional cartesian grid with periodic boundaries, so each processor has 26 neighbors in 3d or 8 in 2d. Our algorithm subdivides the array of cells in equally sized parts which then are assigned to the processors. Figure 2 shows a two-dimensional example of how the cells are associated with the processors: A 12×12 array of cells (shaded) is distributed within a 3×3 grid of processors. A layer of empty cells (non-shaded) is added to each processor. These are used as temporary stage for particle data received from neighbors. We call the extra cells *buffer* cells. The outermost layer of non-empty cells at the surface of a processor's part are called *surface* cells.

Particles in non-surface cells interact only with particles on the same processor, whereas those in the surface cells also interact with particles in surface cells on the neighboring processors. To calculate the forces for them, data from the surface cells are copied to the buffer cells of the neighbors, as indicated by the arrows in Fig. 2.

Three communication steps have to be performed at each time step: First, the particle positions have to be exchanged with the neighbor processors for the force calculation, second, the forces accumulated due to Newton's third law by particles in the buffer cells are sent back to the original processor, and third, after the positions have been updated, all atoms that have left or entered a processors's column must be transferred to their new processor.

As indicated in Fig. 2 by the arrows, data for each surface cell could be communicated cell by cell. This is, however, not optimal since it creates a large number of small messages. IMD first collects the data for a given neighbor into an intermediate data buffer and then translates the whole buffer with only one single MPI call.

We have tested the cell by cell and the buffer scheme. Details are given in paper I.

3 Online Visualization and the Generation of Pictures

Until recently MD simulations were carried out in the following way: The simulation was run, data were collected and stored for analysis *after* the simulation had been finished. With the large massively-parallel supercomputers such a scheme is no longer feasible, since in many cases the amount of data is too large to be stored.

One way out of this problem is to visualize the data online. More advanced schemes even permit to steer the simulation online, for example to change the applied pressure or temperature, to alter the frequency of picture production or data storage or to interrupt the simulation in case of errors.

With IMD online visualization is possible, and the basic interfaces for online steering are also given. Before we go into details we have to provide a few definitions: the supercomputer (or workstation) which runs the simulation will be called (visualization) server and the workstation, which is used to visualize the data, is called client. IMD uses ordinary TCP sockets for communication. There may be problems if the supercomputer is hidden behind a firewall like it is the case at the HLRS in Stuttgart. For this reason IMD uses port numbers above 31913 (which are open), but it may still be necessary to ask the computing center explicitly to open up ports for communication. Let's assume that you have compiled IMD with the option for online visualization. Then one starts the simulation interactively or sends the job to a batch queue. It is possible at any time during the simulation, and in any mode, to connect the client to the simulation and to get pictures. If the simulation is running for several days, one may connect to the server once a day and check the status of the simulation job.

The communication scheme implemented in IMD works in the following way: The central part of a MD simulation is the main loop for integration of the equations of motions. At a given time interval a function, `check_sockets` is carried out which connects to the client and looks whether certain flags have been set. If this is the case, the program terminates or sends a picture of the atoms or a picture of the potential or kinetic energy distribution as required. (Changing of parameters is not yet implemented). Due to the firewall it is important that the server contacts the client. Without a firewall the opposite way would also be possible.

3.1 Pictures

The pictures produced by the online visualization routines are raw data written in binary format. This has the advantage of being faster and more flexible than writing formatted output. The pictures actually consists of histograms. At the beginning of the simulation one has to define an array of bins, and then the data are collected into the bins and written in binary format. There are tools to convert the binary data into ppm[b]-files. In three dimensions this can be done with the help of VolRend[7]. In two dimensions it is possible to write ppm-files directly, but then

[b] ppm: portable pixel map.

one looses the possibility to manipulate the pictures for example by redefining the color map.

Here we may add a few words on the picture options currently implemented in IMD: We found that for our purposes it is helpful to know the instantaneous distribution of the kinetic and potential energy, or more specific, the change of their values. The potential energy of surface atoms, for example, is smaller than the potential energy of the bulk atoms due to the missing bonds. If a crack opens, bonds are broken, and the potential energy of the atoms near the crack boundary changes drastically. Thus the potential energy allows to locate the crack. The kinetic energy, on the other hand, allows to see the shock wave fronts emitted by the crack tip. For simple crystals it is sufficient to plot the atoms with a lower energy, but this is not possible for complicated structures like quasicrystals. IMD permits to store reference energies which can then be subtracted from the actual energies. Atoms with large energy change can thus be identified easily.

4 Metacomputing

Metacomputing is the ultimate extension of massively-parallel (super-)computing: it denotes the intercontinental clustering of massively-parallel supercomputers. Currently, metacomputing is a hot topic and is still under development. It is not yet possible for the ordinary user to sent jobs to an intercontinental batch queue, only demonstration have been carried out. Not every kind of massively-parallel application is suitable for metacomputing, and intercontinental metacomputing will certainly stay exotic for quite some time since the intercontinental data lines are too expensive. But metacomputing has its applications: if an application consists of a part more suitable for massively-parallel systems and another part more suitable for vector machines, it could easily be distributed on two computers with different architectures using PACX.

IMD is prepared for metacomputing with the PACX[c] library, developed at the HLRS in Stuttgart[6]. If necessary PACX replaces the ordinary MPI routines by new routines. To use PACX one simply has to relink IMD with the PACX library prior to the MPI library, so that the original MPI routines are overwritten by the PACX routines. Currently only a subset of the most important MPI routines is implemented. The cartesian topology used for the cell division in IMD for example is not provided and has been replaced by self-written routines.

In the following we will first describe how metacomputing with PACX works, and then discuss what special requirements exist for the simulation program.

4.1 Metacomputing with PACX

On each of the participating supercomputer nodes two processors are set aside for communication, one for sending and one fore receiving. Figure 3 shows the communication between two hosts: The communication within one node is unchanged. But if a processor on one node wants to communicate with a processor on the other node, a two-step process is initiated: first the message is broadcasted locally to

[c]PACX: **PA**rallel **C**omputer e**X**tension.

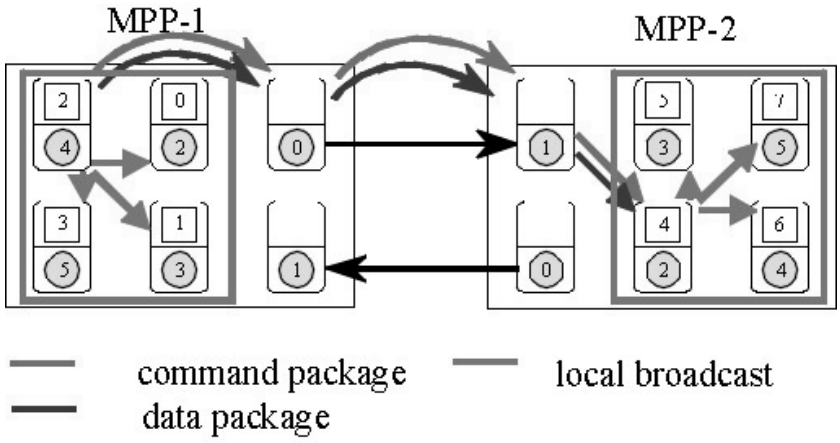

Figure 3. Communication between two supercomputers.The labels in the squares and circles are the global and local processor addresses, respectively. The arrows indicate how the communication processors are connected.

the communication processor, then it is sent to the communication processor on the other node and transmitted again over the local network to the destination processor.

The communication with PACX was originally limited to two participants, meanwhile it has been extended to any number of supercomputer nodes as displayed in Fig. 4 for example for three nodes. All communication processors lie on one loop, thus two communication processors per supercomputer node are sufficient.

4.2 Special Requirements for Metacomputing

IMD has been written for massively-parallel supercomputers like the Cray T3E, where the communication between the processors is very fast and and the communication time is short compared to the time required for a time step. For metacomputing this is no longer the case. The internal communication time within the T3E is 18 μs, with PACX it is 21 μs. Communication between the supercomputer nodes is at least one order of magnitude slower than within a supercomputer, even if high-speed and high-bandwidth dedicated ATM lines are used. The time to send a the message between two partitions within the same T3E is 3.5 ms, between Stuttgart and Manchester it is 45 ms, and between Stuttgart and Pittsburgh it is 75 ms! This phenomenon is called latency. To avoid wrong results the computations have to be stopped until the inter-supercomputer communication has been finished. This is very unsatisfactory and a way to interlace calculation and communication has to be found. The solution is latency hiding which is currently not included in IMD, but its implementation is rather straightforward.

In section 2.1 we have described how to set up cells and how to distribute them on parallel computers. A glance at figure 2 tells us that it is not necessary to wait

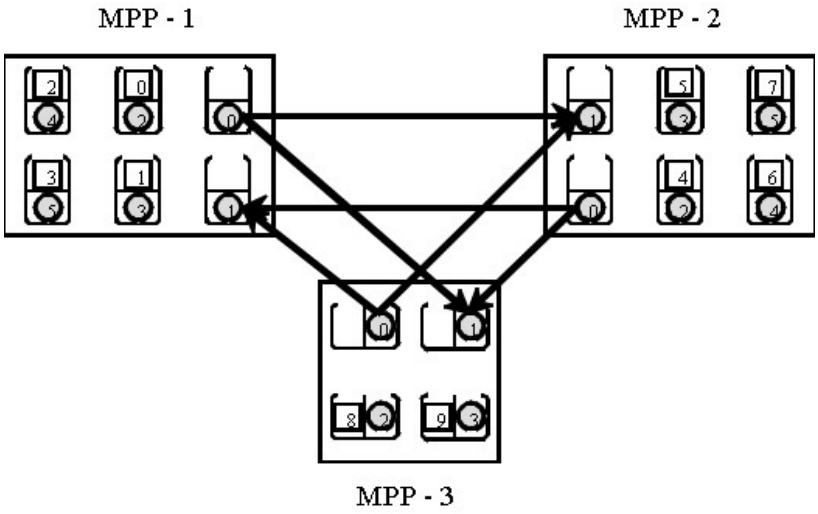

Figure 4. Communication between three supercomputers. Description as in Fig. 3.

for communication until the computations in all cells are completed. After the computations in a surface cell and in its neighbors have been finished the result can be sent immediately to the other nodes. In the buffered scheme one would have to wait nevertheless until the surface cells belonging to one coordinate direction have been completed. During communication time the processors will compute the forces and displacements for the non-surface and non-border cells since these cells are independent from the status of the cells on the other processors. If the computation for the central cells is done before the communication exchange has ended, one still has to wait some time, but the latency time will now be much shorter than before.

5 Applications: Shock Waves in Quasicrystals and Crystals

The last section is dedicated to applications. Quasicrystals are one of the main topics at the ITAP, and so it is quite natural that most of the applications of IMD up to now deal with quasicrystals. We are especially interested in the mechanical properties of quasicrystals. The first studies using IMD were concerned with cracks in two-dimensional quasicrystals[8,9]. The material in a notch is removed from the sample, and then the upper and lower boundaries are moved apart by a fixed amount until a crack is emitted the notch. In a crystal the crack would propagate smoothly. In quasicrystals a crack is meandering around, thereby avoiding clusters of strongly bound atoms(Fig. 5). At the crack tip bonds are breaking and emitting shock waves. To avoid an interaction of the shock waves reflected at the boundary of the sample, we have generated a stadium around the crack. Waves meeting the stadium are damped out[12].

Further studies are concerned with shear simulations of quasicrystals. In this

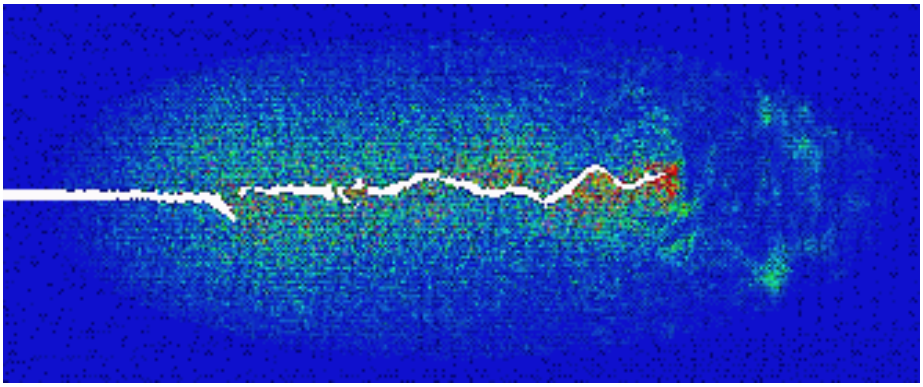

Figure 5. Crack in a quasicrystal. Dark shading indicates low energies, light shading high energies.

case the upper border is shifted parallel to the lower border but in opposite direction. Dislocations are created which move through the quasicrystal. In contrast to the crystal case the sample is not perfect again behind a dislocation but contains a stacking-fault-like defect.

In three dimensions it was not possible to create dislocations by shear distortion since the structure was destroyed completely before dislocations were formed. Therefore the dislocation was generated by cutting the sample into two pieces up to the center, and then shifting the upper part with respect to the lower part[13]. Along the surface cut through the sample a phason wall is formed. Figure 6 displays atoms belonging to this stacking-fault-like defect. The irregularity results from the quasiperiodicity of the structure. The behavior of the dislocation is again different from the crystal case: The dislocation is not straight, and it climbs when it moves. We believe that this is caused again by strongly-bound clusters.

Quantum dot models are currently studied in two dimensions. We are also interested in the interface between two hexagonal crystals with different lattice spacings, especially how the distortions propagated into the bulk.

5.1 Shock waves

There have been two main reasons to study shock waves in three dimensional quasicrystals. The first is that shock waves are interesting by themselves. With shock waves a large variety of defects can be created, from point like over stacking faults up to complete amorphization. The second reason is that shock waves are suitable for metacomputing simulations. Normally an ideal sample first has to be relaxed so that the atoms move to the equilibrium position. Then it has to be equilibrated at the specific temperature and volume or pressure at which the simulation shall take place. And finally the simulation has to be carried out and the data have to be collected. In the case of the dislocation study in three dimensions additional manipulations by hand had to be carried out. To run such a complicated schedule is currently not possible for metacomputing, even with multiphase simulations feature implemented in IMD (see Sec. 1). We had to find a setup where relaxation

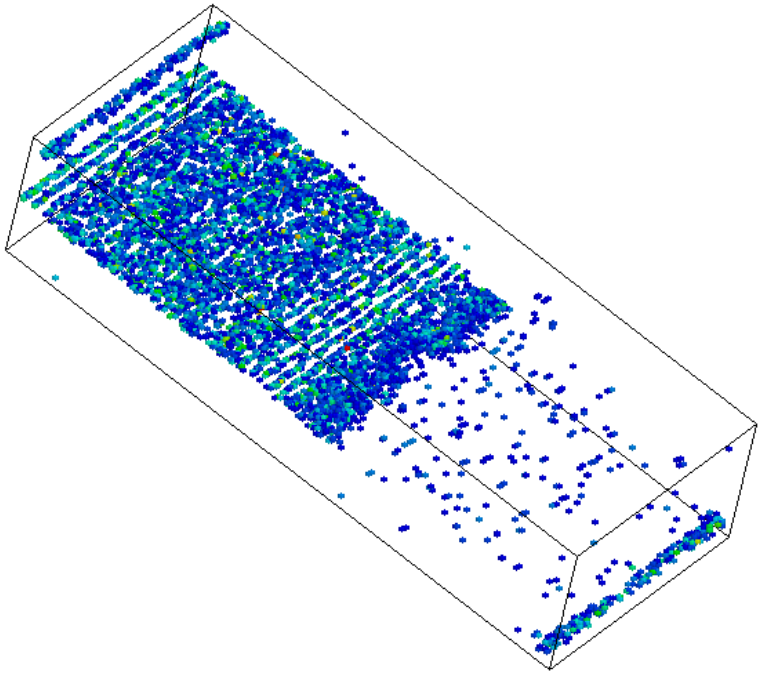

Figure 6. Stacking-fault-like defect behind a dislocation in teh center of a three-dimensional quasicrystal. Only interface atoms are shown. The upper part of the sample has been shifted with respect to the lower part.

and equilibration can be avoided. This is the case for shock wave simulations since the kinetic energy of the shock wave is usually large compared to the energies of the atoms in the sample.

The shock waves were generated in the following way: The rodlike sample had periodic boundaries in two directions and was open at the other sides. At one side in a small slab the velocities of all atoms were set to a large value with a direction pointing along the long axis of the sample.

For comparison we have studied two samples: A binary quasicrystal with TI structure[14], consisting of two structural motives, a prolate rhombohedron and an oblate rhombohedron, and a crystal built of prolate rhombohedra only. Both samples contained about 1 million atoms. Since the composition of the two samples is slightly different, their potential energy also slightly differs. But otherwise they are as similar as possible. Their behavior, however, is strikingly different. First of all, the propagation velocities of the crystal were much larger than the velocities in the quasicrystal for shock waves of the same intensity (Fig. 7). We assume that the aperiodicity of the quasicrystal decelerates the shock wave.

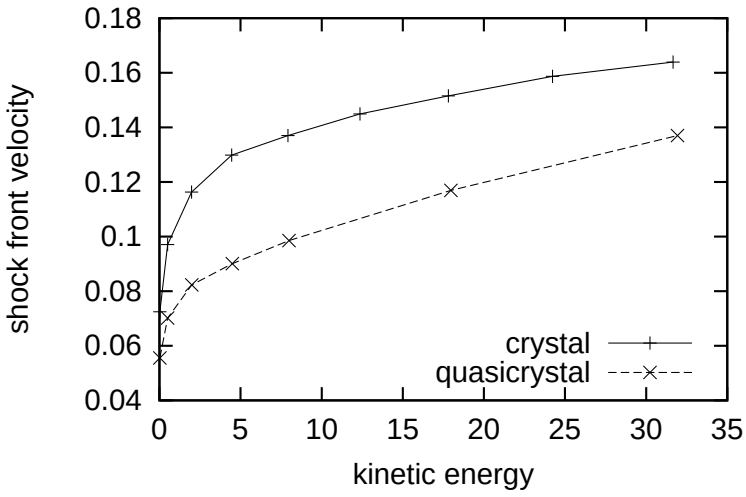

Figure 7. Velocity of the shock wave (in arbitrary units)*vs.* strength of the shock wave, expressed in units of the average kinetic energy.

A second difference is observed for the strength of the material: the crystal is much stronger than the quasicrystal despite the similarity of the structures. Below an average kinetic energy of about 3 units no defects occur in the quasicrystal. Between 3 and about 13 shear bands show up which consist of parallel lines of dislocations at 45 degree with respect to the shock wave direction. Above 13 the sample becomes amorphous. The crystal is still perfect at a kinetic energy of 12 units, at 18 domains of disordered material show up, and at 24 it starts to become amorphous. At an energy of 32 the whole sample has turned amorphous. Shear bands have not been found in the crystal. As with the shock wave velocity we again are surprised by the strong difference between the crystal and the quasicrystal. A straightforward explanation is not yet at hand.

In summary we have found that it is indeed possible to create a zoo of defects in quasicrystals with the help of shock waves. The results have turn out to be much more complicated to understand than expected.

Acknowledgments

IMD has originally been written by Jörg Stadler and was funded by project A2: "Molecular Dynamics Simulations of Quasicrystals" of the Sonderforschungsbereich 382: "Procedures and Algorithms for the Simulation of Physical Processes on Supercomputers". Meanwhile IMD has been augmented by new features and maintained by a team of programmers including Marco Brunelli, Franz Gähler, Jutta Hahn, Martin Hohl, Christof Horn, Ralf Mikulla, Gunther Schaaf and myself in the quasicrystal group led by Prof. Dr. H.-R. Trebin.

The help of the PACX group at the HLRS, led by Michael Resch and Edgar

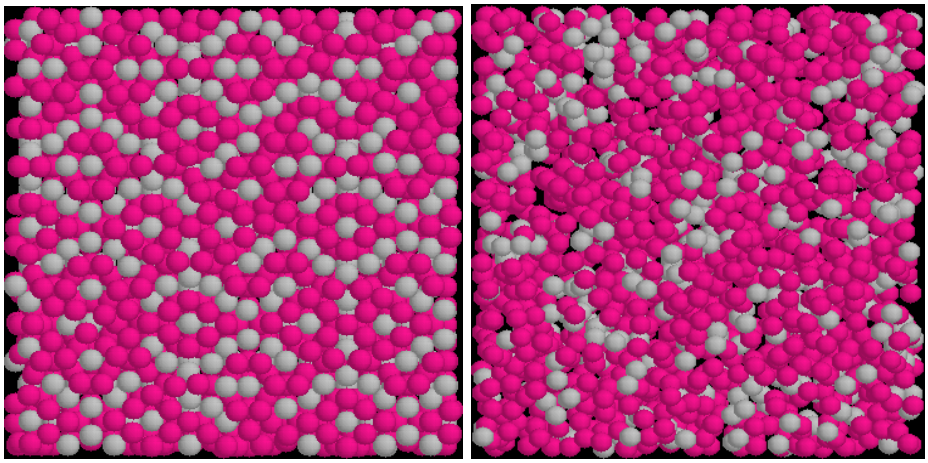

Figure 8. The quasicrystal structure at an energy of 32 units. The shading indicates two kinds of atoms. Left: before the shock wave has passed, right: amorphous state behind the shock wave.

Gabriel, is also greatfully acknowledged.

References

1. J. Stadler, R. Mikulla, H.-R. Trebin, Int. J. Mod. Phys. C **8**,1131 (1997).
2. Homepage for IMD: www.itap.physik.uni-stuttgart.de/~johannes/imd-home.html.
3. M.S. Daw, M.T. Baskes, Phys. Rev. B **29**, 6444 (1984).
4. M.P. Allen, D.J. Tildesley, "Computer Simulation of Liquids", Oxford Science Publications, Oxford (1987).
5. J.R. Beeler, "Radiation Effects Computer Simulations", North Holland, Amsterdam, p 27 (1983).
6. Homepage for PACX: http://www.hlrs.de/structure/organisation/par/projects/pacx-mpi/.
7. VolIMD is a special version of VolRend (See Ref. [2]). Homepage for VolRend: www.uni-stuttgart.de/RUSuser/vis/People/roland/volrend/volrend.html.
8. R. Mikulla, J. Stadler, F. Krul, H.-R. Trebin, P. Gumbsch, Phys. Rev. Lett. **81**, 3163 (1998).
9. R. Mikulla, J. Stadler, P. Gumbsch, H.-R. Trebin, in "Proceedings of the 6th International Conference on Quasicrystals", World Scientific, Singapore, eds. S. Takeuchi, and T., Fujiwara, p. 485, (1998).
10. R. Mikulla, P. Gumbsch, H.-R. Trebin, to appear in: Phil. Mag. Lett.
11. www.hlrs.de/news/events/1998/sc98/sc98_hlrs_booth.html.
12. S.J.Zou, P.S. Lohmdahl, R. Thomas, B.L. Holian, Phys. Rev. Lett. **76**, 2318 (1996).
13. G. Schaaf, Diploma Thesis, Stuttgart (1998).
14. J. Roth, R. Schilling, H.-R. Trebin, Phys. Rev. B **41**, 2735 (1990).