A NEW PROGRAMME PACKAGE FOR STRUCTURAL ANALYSIS OF COMPUTER SIMULATED SOLIDS

GRZEGORZ BERGMAŃSKI[1], JAROSŁAW RYBICKI[1,3]
AND GIORGIO MANCINI[2]

[1] *Department of Solid State Physics,*
Faculty of Technical Physics and Applied Mathematics,
Technical University of Gdansk,
Narutowicza 11/12, 80-952 Gdansk, Poland
tuptus@task.gda.pl, ryba@task.gda.pl

[2] *INFM, UdR Camerino, Istituto di Matematica e Fisica, Università di Camerino,*
Madonna delle Carceri, Camerino (MC), Italy
mancini@pcman.unicam.it

[3] *TASK Computer Centre, Narutowicza 11/12, 80-952 Gdansk, Poland*

(Received 10 March 2000)

Abstract: The opportunity to gain detailed information on representative set of rings and chains of atoms is of great importance in the analysis of medium-range order in the computer simulated structures of solids. In the paper, a new program package (ANELLI) for ring and chain analysis of computer simulated atomic structures is presented. The paper includes detailed description of input and output files. The package presentation is followed by exemplary results obtained using our programs.

Keywords: ring analysis, structure of matter, molecular dynamics, computer modelling

1. Introduction

For many years computer modelling has been a widely used tool of research within solid state physics (*e.g.* [1–3]). Recently, due to a remarkable increase of computational power, the simulations of multi-component systems with 10^5 atoms are commonly performed, and one faces a non-trivial problem of structure recognition. As an output from the simulation programs one usually obtains the co-ordinates of atom positions within the simulation box. Such kind of information on the system, although theoretically complete, requires further data processing, *e.g.*, calculation of radial and angular distribution functions (RDFs, and ADFs, respectively). Having obtained the RDFs and ADFs, and using some other tools, as common neighbours analysis (CNA) [4], one can readily establish the structure of the first co-ordination shell for each element, *i.e.* the short-range order in the simulated structure. However, as far as the medium-range order is concerned, it is often very difficult to extract unambiguous information from RDFs and ADFs, and in order to describe the second, and further co-ordination shells, one should use much more advanced methods of the structural analysis.

One of the possible approaches consists of the analysis of properly constructed clusters of edge and/or face sharing Voronoi polyhedra [5, 6]. This method, although very efficient in the detection of crystalline regions of various symmetry [6–8], works well only for close packed systems. In open systems serious problems appear in the construction of the Voronoi network, and in the procedures eliminating short edges and small faces. Thus, the ring analysis seems to be a more general method, since it works for both close-packed, and open structures. In this contribution a new program package for the ring and chain analysis is presented.

As far as ring perception is concerned, a great number of algorithms have been published. An extensive critical review on the early algorithms was given by Downs *et al.* [9]. However, before early nineties neither of them was efficient enough to be applied effectively for the structural analysis for systems containing thousands of atoms. Only in 1993 Balducci and Pearlman [10] presented an algorithm (BP algorithm), which appeared to be sufficiently effective. However, it has probably never been implemented [11]. While analysing the BP algorithm one of the authors came to the conclusion that some of the rings directed to the selector will certainly not add another ring to the minimum basis, and a part called *pre-filter* was added to the algorithm [12, 13]. The modified algorithm has been implemented, and the *anelli* program has been used successfully for the description of medium range order in some oxide glasses [14–16]. Our ring perception program *anelli* has been recently supplemented by several other programs, forming a program package (ANELLI) for the medium-range order analysis of computer simulated solids.

In Section 2 we describe in detail all the programs of the ANELLI package. Since all the input and output files described, this section is a kind of user's manual. Several examples of the results obtained using our package are presented in Section 3. Section 4 contains concluding remarks.

2. The ANELLI package

The package includes 8 programs written in C language: *fnlg*, *ggsplit*, *gbi*, *anelli*, *frs*, *recover*, *geom*, and *pdb*. Figure 1 presents the dependence among the individual programs. By the arrows between the program names, standard input and output file extensions are given. The *fnlg*, and *pdb* programs are strongly user-dependent, being rather input and output interfaces of the proper package. However, for completeness, we will describe in turn all eight programs.

2.1. Full neighbour-list generation (fnlg)

To begin with the structural analysis, the adjacency matrix should be calculated. We have chosen a neighbour-list representation. The *fnlg* program constructs the neighbours lists for all the atoms from the simulation box, according to a simple adjacency criteria.

The input file with atom positions has a simple 4-column ASCII format. In what follows, the $*$*.asc* extension will be used for this file. Its first column contains the atom type labels ($1,2,\ldots,K$ for K various atomic species), whereas columns 2–4 contain the x, y, z co-ordinates (from the interval $[-1.0,1.0]$) of subsequent atoms. For each pair of atoms the distance between them is calculated taking into account the standard periodic boundary conditions, and the atoms are considered to be neighbours if the distance between them is

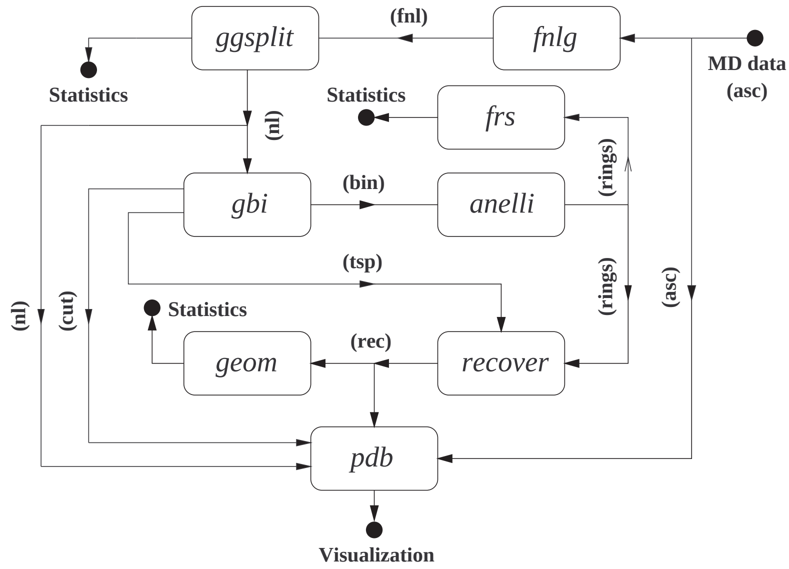


Figure 1. *ANELLI package scheme*

smaller than, or equal to the cut-off radius r_c given by the user. The output file defines a graph, which represents inter-atomic bonds, containing full lists of neighbours of all atoms specified in the input file (Figure 2). Of course, the set of all bonded atoms need not form a single connected graph.

PROGRAM USAGE:	*fnlg input_file* *fnlg* (if the default name *fnlg.in* of the *input_file* is used).

INPUT FILE:

The program reads a file containing the following data:

LINE 1: character $*$ 60 – name of the file containing atom positions ($*$.*asc*);

LINE 2: character$*$60 – prefix of the output file. Default extension $*$.*fnl* (*f*ull *n*eighbour *l*ist) will be added to it;

LINE 3: integer – number of different atomic species, K;

LINE 4: $K*$integer – numbers of atoms of subsequent atomic species (separated by commas);

LINE 5: $K*$integer – bonds specification. The line contains L ‘1’ and $K-L$ ‘0’ values (separated by commas). When 0 appears in i-th position, the atoms of i-th kind will be skipped during the neighbour searching. For example, if $K=3$, the sequence $1,0,1$ means that only the atoms of the first and the third kind can be neighbours ($L=2$);

LINE 6: $3*$double – *bl_x_half*, *bl_y_half*, *bl_z_half* – halves of the lengths of the simulation box edges (parallelopipedal);

LINE 7: double – maximum bond length r_c (in the same units as simulation box dimensions).

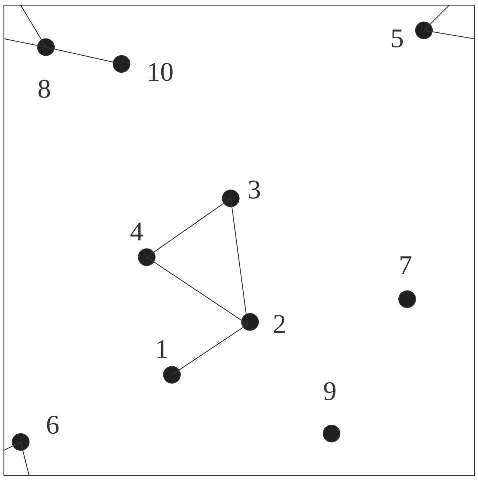
OUTPUT FILE: *prefix.fnl*

The format is as follows:

COLUMN 1: integer∗8 – subsequent number i of atom within the simulation box, $i = 1,\ldots,N\_at$;

COLUMN 2: integer∗8 – number of its neighbours, *i_neigh*;

COLUMNS 3 TO $2+i\_neigh$: integer∗8 – reference numbers of neighbouring atoms.



```
 1      1      2
 2      3      1      3      4
 3      2      2      4
 4      2      2      3
 5      2      6      8
 6      2      5      8
 7      0
 8      3      5      6     10
 9      0
10      1      8
```

Figure 2. *Exemplary atom configuration and the corresponding ∗.fnl file*

2.2. Connected sub-graph construction (ggsplit)

Before further data elaboration the full bonding graph should be split into separate connected sub-graphs. This is performed by the *ggsplit* (*g*eneral *g*raph *split*ting) program, which transforms the full neighbour list file (∗*.fnl*) into files (∗*.nl*) containing neighbour lists of atoms for subsequent connected sub-graphs. Each of the ∗*.nl* output files contains data of one connected cluster (Figure 3). Moreover, the program gives information on sizes of the detected sub-graphs.

PROGRAM USAGE: *ggsplit input_file*
ggsplit (if the default name *ggsplit.in* of the *input_file* is used).

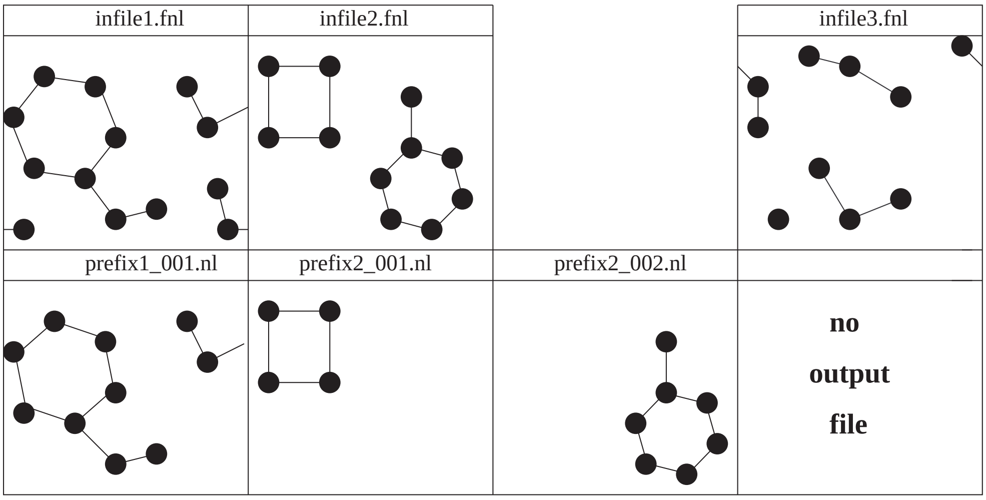


Figure 3. *The ggsplit program functioning for three exemplary atom configurations*

INPUT FILE:

The program reads a file containing the following data:

LINE 1: character$*$60 – prefix of the name of input file (default extension $*$.*fnl* is expected). During the program execution many files may appear (a separate file for each connected sub-graph). The file names will be *prefix_nnn.nl*, where *nnn* is an integer in the range from 0 to 999, and *nl* stands for '*n*eighbour *l*ist';

LINE 2: integer$*$2 – number of different atomic species in the $*$.*fnl* file;

LINE $2+1$: character$*$2, integer$*$6, integer$*$6 – label of the first atomic species (*e.g.* chemical symbol of the element), reference number of the first atom of the first atomic species, and the number of atoms of the first species;

…

LINE $2+L$: character $*$ 2, integer $*$ 6, integer $*$ 6 – label of the last (L-th) atomic species, reference numbers of the first atom of the L-th species, and the number of atoms of the L-th species;

LINE $3+L$: $3*$double – *bl_x_half*, *bl_y_half*, *bl_z_half* – halves of the lengths of the simulation box edges;

LINE $4+L$: integer – minimum cluster size to be written into output file, M.

The order of the atom appearance in lines $3\div2+L$ must agree with the order of atoms in the $*$.*asc* file. For example, if the simulation box contains 500 Pb, 500 Ge, and 1500 O atoms, the third, fourth, and fifth line should read:

```
Pb   1     500
Ge   501   500
O    1001  1500
```

OUTPUT FILES: *prefix_nnn.nl, split.out*

prefix_nnn.nl: the file name consists of the *prefix* defined in the first line of the input file, and of a *_nnn* string, where *nnn* is a number (integer, $0 \le nnn \le 999$)

of a successive connected sub-graph. Each ∗*.nl* file contains reference numbers of atoms forming one connected sub-graph written in the ∗*.fnl* file. The ∗*.nl* files are the input files for the *gbi* and *pdb* programs;

split.out: contains information concerning the sub-graph statistics. It consists of three blocks (Figure 4). The first one repeats the input data. The second block gives information on the smallest sub-graphs (1-, 2-,..., and $(M-1)$-atom clusters), and on the maximum sub-graph size. Block 3 lists the contents of the output ∗*.nl* files. The ∗*.nl* files are created only for sub-graphs containing M or more nodes.

```
ATOMIC SPECIES UNDER CONSIDERATION:
Ge    500
O     1500
    SIMULATION BOX DIMENSIONS:
               X= 44.932 A
               Y= 44.932 A
               Z= 44.932 A
NUMBER OF 1-ATOM CLUSTERS: 123
   SPECIFICATION:
ATOMS Ge:                  0
ATOMS O:                   123
NUMBER OF 2-ATOM CLUSTERS: 3
   SPECIFICATION:
      Ge-O                 3
NUMBER OF 3-ATOM CLUSTERS: 2
   SPECIFICATION:
      Ge-O-Ge              1
      O-Ge-O               1
 TOTAL NUMBER OF CONSIDERED ATOMS :      2000
 ATOMS CONNECTED TO AT LEAST ONE ATOM :  1495
 THE BIGGEST CLUSTER CONTAINS :    1007  ATOMS
 WHICH IS  50.350% OF ALL ATOMS IN THE INPUT CONFIGURATION
*****************************************************
CONTENTS OF OUTPUT *.nl FILES:
           21 CLUSTERS OF SIZE     4
           47 CLUSTERS OF SIZE     5
           15 CLUSTERS OF SIZE     6
           11 CLUSTERS OF SIZE     7
           10 CLUSTERS OF SIZE     8
            4 CLUSTERS OF SIZE     9
            3 CLUSTERS OF SIZE    12
            1 CLUSTERS OF SIZE    23
            1 CLUSTERS OF SIZE    98
            1 CLUSTERS OF SIZE   103
            1 CLUSTERS OF SIZE  1007
*****************************************************
```

Figure 4. *An exemplary split.out file. The ∗.asc file contained 500 Pb, 500 Ge, and 1500 O atoms. Only the Ge-O bonds were constructed (line 5 in fnlg.in file: 0,1,1)*

2.3. Elimination of dangling structures with no rings (gbi)

The *gbi* program *g*enerates *b*inary *i*nput file (default ∗*.bin* extension) for the *anelli* program. Since the execution time and the memory needed by the *anelli* program depend exponentially on the graph size, one should reduce the latter as much as possible. Thus, the *gbi* program eliminates from the connected sub-graphs all dangling structures (DSs), that is all the atoms which do not belong to any ring nor to paths between rings (Figures 5a, 5b). Then all the remaining atoms are re-numerated from 0 to $N-1$, where N is the number of all the atoms, which remained after the elimination of all dangling structures. When a sub-graph does not contain any ring, no ∗*.bin* output files will be created (Figure 5c). Since the geometry of eliminated DSs is also of interest, for the sake of further analysis they are all written to files.

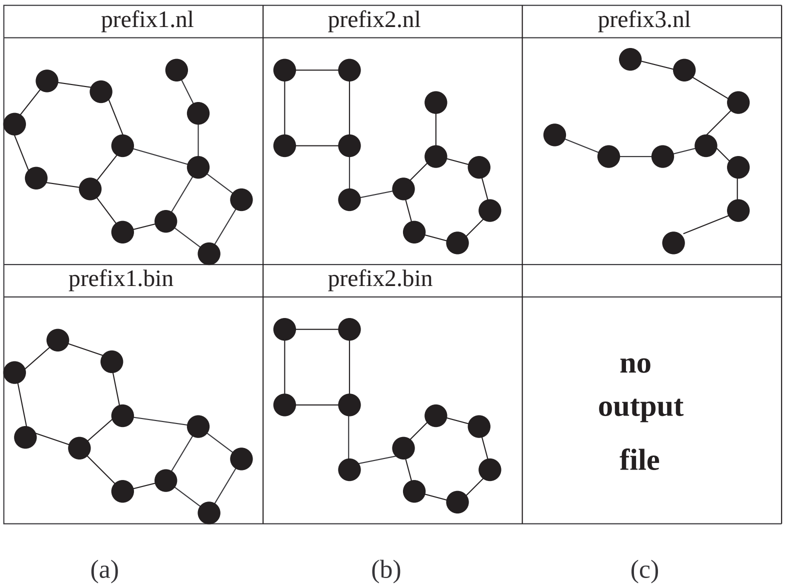


Figure 5. *The gbi program functioning for three exemplary atom configurations*

PROGRAM USAGE: *gbi input_file*
gbi_all prefix – the *gbi* program will be executed for all the input files *prefix_nnn.nl* in the current directory;

INPUT FILE: *prefix_nnn.nl*

OUTPUT FILES: *prefix_nnn.bin*, *prefix_nnn.tsp*, *prefix_nnn_txxx_nyyy.cut*, *prefix_nnn.dss*, *prefix.bif*

prefix_nnn.bin: a binary file containing full information on the reduced graph. Its format is as follows:

RECORD 1: integer equal to 1. This piece of data is necessary to establish whether the numerical representation used in the input file is the same as in the computer actually executing the program. If these representations are different, the data are being properly converted by the program itself;

RECORD 2: number of nodes (*N_node*);

RECORD 3: number of neighbours of the first node, neigh(1);

RECORD 3+1: number of the first neighbour of the first node;

…

RECORD 3+neigh(1): number of the neigh(1)-th neighbour of the first node;

…

…

…

If the neighbour number is higher than the current central node number, the 0 value should be written in the corresponding record. No data for the last node should be given.

prefix_nnn.tsp: During the re-numeration of atoms in the program information about atom numbers is lost: the node numbers in *prefix_nnn.bin* file are no longer atom numbers in the simulation box. The *prefix_nnn.tsp* file contains information indispensable to restore the original atom numeration later on (by the *recover* program).

prefix_nnn_txxx_nyyy.cut: Contains data on one DS with at least two atoms that have been eliminated from the original graph. Here *xxx* and *yyy* are integers ranging from 0 to 999. *xxx* indicates the size (number of atoms) of (in) the dangling structure written to the file. *yyy* is the subsequent number of the *xxx*-atom structure cut from the *nnn*-th sub-graph. The file format is as follows:

COLUMN 1: successive DS number, *yyy*;

COLUMN 2: the size of this DS, *xxx*;

COLUMNS 3 TO 2+*xxx*: the reference numbers of atoms in the simulation box forming this DS.

prefix_nnn.dss: contains a concise statistics over all DSs (Figure 6).

LINE 1: the name of the ∗*.nl* file being considered;

LINE 2: the total number of DSs found in the ∗*.nl* file, *N_DS_tot*;

LINE 3: the number of DSs longer than 1;

LINE 4: the total number of atoms in all DS's;

LINE 5: the fraction [%] of atoms in all DS's (*N_DS_tot*/*N_graph_tot*);

LINE 6: the average size of DSs;

LINE 7: the average size of DSs longer than 1;

LINE 8: the total number of bifurcation nodes in all DSs, *N_DS_bif* (a bifurcation node being defined as a node with at least three neighbours);

LINE 9: the fraction [%] of bifurcation nodes (*N_DS_bif*/*N_DS_tot*);

LINE 10: the average bifurcation density, defined as the ratio of the total number of atoms in DSs longer than 1 to the total number of bifurcations (B parameter).

prefix.bif: generated only by the *gbi_all* version. The file contains individual B parameters calculated for all files considered and the average value of all B's.

```
******** STATISTIC OF DS's IN FILE: input_002.nl  ******
Total number of DS's                               = 27
Number of DS's longer than 1                       = 7
Total number of atoms in all DS's                  = 44
This is 34.92% of all atoms in the input_002.nl file.
Average number of atoms per DS                     = 1.63
Average number of atoms per DS longer than 1       = 3.43
Total number of bifurcations                       = 5
This is  11.36% all atoms in all DS's.
B parameter                                        = 0.208
```

Figure 6. *An exemplary prefix_nnn.dss file*

2.4. Rings basis calculation (anelli)

This is the main program in the package, which determines the ring basis using the Balducci-Pearlman-Mancini algorithm [12, 13]. The output file contains full information on all basal rings and gives the ring statistics.

PROGRAM USAGE: *anelli prefix_nnn* (without the *.bin* extension);
anelli_all prefix – the *anelli* program will be executed for all the *prefix_nnn.bin* files in the current directory;

INPUT FILE: *prefix_nnn.bin* (output file of the *gbi* program, see Subsection 2.3)

OUTPUT FILE: *prefix_nnn.rings*. The file lists all the rings found, *N_rings*, specifying the ring edges in terms of node numbers (the first *N_rings* lines). The list is followed by a simple ring statistics (Figure 7).

2.5. Full ring statistics (frs)

The *anelli* program calculates the ring statistics for one sub-graph only. The *frs* program calculates the *f*ull *r*ing *s*tatistics, *i.e.* the global statistics over all subsequent subgraphs in the current directory. This program requires at least one *prefix_nnn.rings* file.

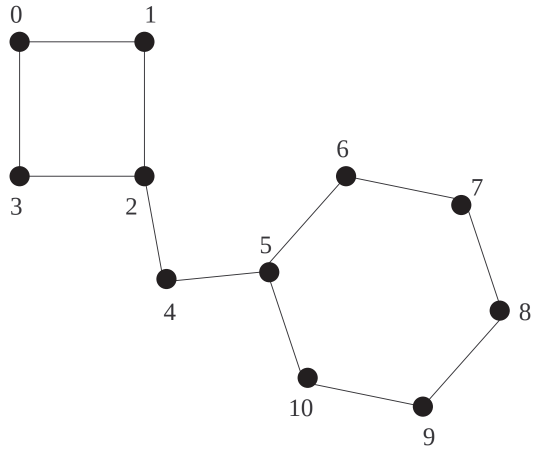
PROGRAM USAGE: *frs prefix* – the program will elaborate all the files in the current directory with names beginning with *prefix*, and having the ∗*.rings* extension;
frs – the program will elaborate all the files with the ∗*.rings* extension in the current directory, independently of the file names.

INPUT FILES: *prefix_nnn.rings* (output files from the *anelli* program, for format specification see Subsection 2.4).

OUTPUT FILES: *prefix_all_rings.stat* (when run with the *prefix* argument);
all_rings.stat (when run with no arguments);
The output file format is similar as in the final section of the *prefix_nnn.rings* file.

2.6. Ring node identification (recover)

All the basal rings are listed in the *prefix_nnn.rings* files in terms of reduced-graph node numbers. In order to make physical analysis possible, it is necessary to restore the



```
[0-1][1-2][2-3][3-0]
[5-6][6-7][7-8][8-9][9-10][10-5]
Input file: prefix_002.bin
Nodes:                               000011
Edges:                               000012
Expected rings:                      000002
Ring(s) found:                       000002
Total rings processed:               000002
Rings rejected by r_selector:        0.00%
Ring(s):
000001 ring(s) of length    000004 (  50.00% )
000001 ring(s) of length    000006 (  50.00% )
```

Figure 7. *An output file of the anelli program for a graph shown*

true atom numbers in the nodes. This can be done using the *recover* program, which uses two input files: ∗*.rings* (output file from the *anelli* program), and ∗*.tsp* (generated by the *gbi* program). The *recover* program generates a file containing all the rings listed in the ∗*.rings* file, but the numbers in it are now the original reference numbers of atoms within the simulation box.

PROGRAM USAGE: *recover prefix_nnn.rings prefix_nnn.tsp*

recover_all prefix – the program will be executed for all pairs of files: the ∗*.rings* file, and the corresponding (the same *nnn*) ∗*.tsp* file (in the current directory).

INPUT FILES: *prefix_nnn.rings*, *prefix_nnn.tsp*

These are the output files from the *anelli* and *gbi* programs, respectively.

OUTPUT FILE: *prefix_nnn.rec*

The ∗*.rec* format is as follows:

COLUMN 1: successive ring number;

COLUMN 2: the length of this ring, *ir*;

COLUMNS 3 TO $2+ir$: the reference numbers of atoms in the simulation box forming the ring.

2.7. Geometrical properties of the basal rings (geom)

The *geom* program calculates:

- distribution of distances of the first neighbours along the rings;
- distribution of distances of second neighbours along the rings;
- distribution of angles in the rings;
- distribution of the sums of all angles along the rings.

The distributions of angles along the rings are calculated taking into account the diversity of all the atom kinds, whereas the distribution of the sums of all interior angles along the rings is calculated regardless of the atom type. The distributions are being calculated separately for all the rings of the same length, and are normalised to the maximum value.

PROGRAM USAGE: *geom input_file*
geom (when the default name *geom.in* of the input file is used)

INPUT FILE:

The input file should contain the following data:

LINE 1: character∗60 – the name of the file with atom positions (∗*.asc*);
LINE 2: character∗60 – the name of the corresponding ∗*.rec* file;
LINE 3: character∗60 – the name of the corresponding ∗*.nl* file;
LINE 4: integer∗2 – number of different atomic species in ∗*.asc* file;
LINE 5: integer∗6 – number of atoms in ∗*.asc* file;
LINE 6: 3∗double – *bl_x_half*, *bl_y_half*, *bl_z_half*
LINE 7: integer – *ring_length* – the lengths of rings to be analysed (≤ 30);
LINE 8: real – *range* – the maximum distance between atoms for the first neighbours distance distribution (the maximum distance for the second neighbours is set automatically to 2∗*range*);
LINE 9: real – *delta_range* – a step for calculating the distance distributions;
LINE 10: real – *angle_step* – a step for angular distributions (in grades). Angles are in the range from 0 to 180°;
LINE 11: real – *sum_angle_step* – an angular step for calculation of distribution of the sums of all the angles along the rings (in grades).

OUTPUT FILES: *prefix_nnn_rmm.adf*, *prefix_nnn_rmm.rdf*, *prefix_nnn_rmm_cl.sec*, *prefix_nnn_mm.sum*

prefix_nnn_rmm.adf: distributions of inter-bond angles along the rings. A string *_rmm* is added automatically to the *prefix_nnn*-particle: *r* stands for ‘ring’, and *mm* indicates the length of the considered rings (specified in line 7 of the input file). For example *_r04* means that the file contains angular distributions calculated over all 4-node rings. In the ∗*.adf* file, the first line contains the successive column labels. In the first column the angular argument sub-intervals are given (the lower sub-interval limit is printed). Further columns contain normalised angle distributions for various atom triples specified by the column labels. The label *k*-*l*-*m* refers to the distribution of angles between *l*-*k* and *l*-*m* bonds; *k*, *l* and *m* are indices of various atom kinds from the ∗*.asc* file. The number of columns in the file depends obviously on the number of different atomic species appearing in the rings.

prefix_nnn_rmm.rdf: distributions of the first neighbours distances. Its format is similar as of the ∗*.adf* file, and the column labels are self-explained.

prefix_nnn_rmm_cl.sec: distributions of the second neighbours distances. In the *_cl* string, added automatically to the file name, the *c*- character is fixed, and stands for *c*entral atom, whereas *l* is an integer (≤ 5) indicating the atom kind, from which the second neighbours distances are being calculated. The file format is similar as for ∗*.adf* and ∗*.rdf* files. The number of columns is equal to the number of various atomic species that turn out to be second neighbours of the *l*-th species.

prefix_nnn_rmm.sum: distribution of the sums of all the interior angles along the rings. The file contains two columns. The first one gives lower limits of successive angle sub-intervals, and the second one the corresponding angle occurrence.

2.8. Generation of input data for graphic programs (pdb)

The last program in the package prepares data for visualisation programs, and so its output format depends on the user's preferences. Below we describe the *pdb* program, which generates files in the PDB format (Protein Data Bank) [17]. The *pdb* program has no *_all* version, since being executed in such a way it would generate usually a great number (up to thousands) output files. Thus, a manual selection of structures to be visualised is preferable.

PROGRAM USAGE: *pdb input_file*
pdb (when default name *pdb.in* of the input file is used).

INPUT FILE:

The input file should contain the following data:

LINE 1: character∗60 – the name of the file with atom positions (∗*.asc*);

LINE 2: character∗60 – the name of the corresponding ∗*.rec* or ∗*.cut* file;

LINE 3: character∗60 – the name of the corresponding ∗*.nl* file;

LINE 4: integer∗2 – number of different atomic species in ∗*.asc* file;

LINE 5: integer∗6 – number of atoms in ∗*.asc* file;

LINE 6: integer∗1 – *par_1* equal to 0 or 1. For 0 only the rings will be visualised (DSs remain masked); for 1 the whole structure contained in the ∗*.nl* file will be visualised (the program uses also all the corresponding ∗*.cut* files present in the current directory). This parameter must be equal to 0 if the program is executed for the ∗*.cut* files;

LINE 7: integer∗1 – *par_2* equal to 0 or 1. For 0 all the rings are written into one file; for 1 a separate file for each ring is created. This parameter must be equal to 0 if the program is executed for the ∗*.cut* files;

LINE 8: integer$*$1 – *par_3* equal to 0 or 1. For 0 only the rings of length stated in line 9 will be taken into account; for 1 the actual value of *ring_length* in line 9 will be ignored, and all the rings specified in the $*$.*rec* file will be considered;

LINE 9: integer$*$2 – *ring_length*, the lengths of rings to be analysed (≤ 30);

LINE 10: 3$*$double – *bl_x_half*, *bl_y_half*, *bl_z_half*;

LINE $10+1$: character$*$2 – label of the first atomic species;

…

LINE $10+L$: character$*$2 – label of the L-th atomic species.

Lines 6 to 8 are ignored if the program is run for the $*$.*cut* files. An exemplary *pdb.in* file is shown in Figure 8.

```
dn_3000.asc
dn_3000_001.rec
dn_3000_001.nl
3
2500
0                  : par_1
0                  : par_2
0                  : par_3
4                  : ring length
22.4660, 22.4660, 22.4660
Pb                 : label_1
Ge                 : label_2
O                  : label_3
```

Figure 8. *An exemplary pdb.in file. All the 4-node rings found in the given* $*$*.nl structure are to be visualised, the data will be written into one file, and the DS's will be masked*

OUTPUT FILES: *prefix_aa_xx_yyy.pdb*

The names of the output $*$.pdb files will be created in dependence on the *par_1*, *par_2* and *par_3* values in the input file. The *aa* string can assume two values: *r*, and *rds*, which means that the file contains data for *r*ings only (*par_1* $=0$), and for *r*ings together with *d*angling *s*tructures (*par_1* $=1$), respectively. The *xx* string is an integer (≤ 30), and denotes the length of the rings elaborated (see line 9 of the input file). If *par_2* in the input file equals to 1, then the *yyy* string will give the successive ring number. If *par_2* equals to 0, then the *yyy* string will assume the value *all*. In the case when all the rings are to be introduced into one file, regardless of their length (*par_3* $=1$), the *xx* string does not appear in the name of the output $*$.*pdb* file, and the *yyy* string assumes the *all* value.

3. Examples of the ANELLI package usage

Below we give two examples of the usage of our package. The examples refer to the classical MD simulations of lead-silicate glasses of compositions xPbO $(1-x)$SiO$_2$ and

xPb $(1-x)$SiO$_2$, with $0.0 \le x \le 0.67$ [14–16]. The simulations have been performed using the MDSIM program [18], which is the implementation of the Anderson method [19]. The atoms were assumed to interact by a two-body potential (Born-Mayer repulsive forces, and Coulomb forces due to full ionic charges, calculated with the aid of the standard Ewald technique). The potential parameterisation was taken from [20]. The numbers of atoms within the simulation box, in dependence on the system stoichiometry, ranged from about 2500 to about 4000 (the edge of the cubic simulation box from ≈ 30Å to ≈ 45Å).

The room temperature (RT) structures were obtained by cooling at two different rates. In the 'slow cooling' scheme (average cooling rate $\approx 10^{13}$K/s) the system was initially prepared in a well equilibrated molten state at 6000K, and then slowly cooled down to 300K, passing equilibrium states at 5000K, 4000K, 3000K, 2500K, 2000K, 1500K, 1000K, 600K. At each temperature the system was being equilibrated during 30000 fs time steps, and sampled during other 10000 fs time steps. The simulations in the temperature range from 6000K down to 2500K were performed in constant volume regime, and from 2000K on in the constant pressure regime (external pressure equal to zero). In a 'fast cooling' scheme the average cooling rate amounted 10^{14}K/s, and the system was initially prepared below the glass transition temperature.

EXAMPLE 1. Thermal relaxation of lead silicate (xPbO $(1-x)$SiO$_2$) glasses

Most crystalline forms of pure silica (SiO$_2$) consist of corner sharing network of SiO$_4$ tetrahedra. Topology of this network can be characterised in terms of Si–O–Si–O... rings. For instance in quartz 6-, and 8- member rings appear, whereas in crystoballite and trydimite only 6-member rings are present. In α-SiO$_2$ relaxed 5-, 6-, and 7-member rings dominate [21]. These, and longer rings are considered as 'relaxed' ones. The shorter rings are considered as 'strained' ones. The shortest possible silicon-oxygen ring, the 2-member ring, corresponds to edge sharing tetrahedra. Let us see how the Si–O–Si–O–... ring structure depends on increasing PbO contents and the glass cooling rate.

The silicon-oxygen rings were calculated for the final RT structures using the cut-off radius for the Si–O bond equal to 2.0Å (minimum of the radial distribution function between two first peaks).

On increasing PbO concentration, because of the presence of O–Pb–O bridges, the silica network gradually looses its connectivity. This manifests itself by the appearance of more and more long rings with increasing x (Figure 9). It has been also observed, that for higher PbO contents some structural defects appear (2-, and 3-member rings). Figure 10 shows the influence of the cooling rate on the glass middle-range structure. As expected, the slowly cooled sample has more relaxed rings. It should be noted, that the first co-ordination shell of the silicon atom is identical for both cooling rates, and the differences in the shape of the second RDF peak do not lead to any conclusions on the difference in the middle range order of the silica subsystem in the two final configurations. As is seen from Figure 10, the differences in the middle-range order, due to different cooling rates, could be easily detected within the ring analysis.

The PbO subsystem becomes practically connected for $x \ge 0.25$. The 2-member rings dominate for all the stoichiometries. Let us focus on the $x = 0.4$ composition. The sums of all the internal angles along the Pb–O–Pb–O rings, calculated from the slowly cooled structure, are sharply peaked between 355° and 360°. Similar, but somewhat wider angular distributions were obtained for the fast cooling scheme. Since the Pb–O–Pb–O rings corresponding to two edge sharing PbO_4 pyramids in red PbO (P4/nmm space group) are flat, the sum of internal angles amounts to 360°. The accordance between the two numbers suggests that the configuration of adjacent edge sharing PbO_4 pyramids in crystalline PbO is preserved in lead-silicate glasses.

More pronounced difference in the structure of the PbO subsystem between quickly and slowly cooled glasses appears in the dangling Pb–O structures. The average sizes of DS's longer than 1 amount to ≈ 6, and ≈ 5 in the quickly and slowly quenched glasses, respectively. The corresponding B parameters equal to ≈ 1.5, and ≈ 1.0, respectively.

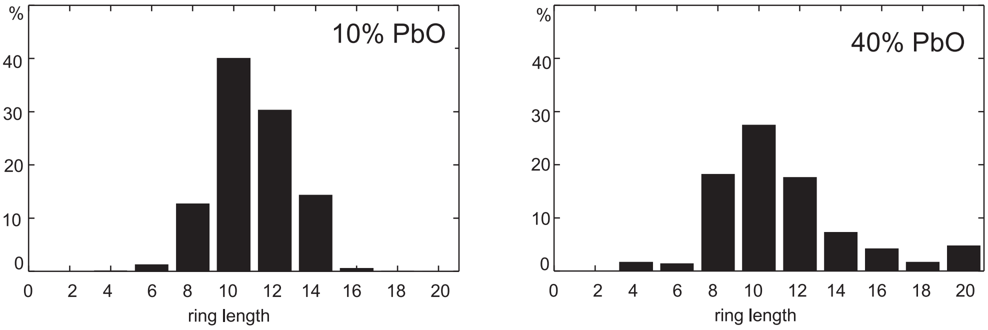


Figure 9. *Distribution of the Si–O–Si–O–... ring lengths in xPbO (1−x)SiO_2 glasses, x = 0.1 and 0.4. Slow cooling scheme*

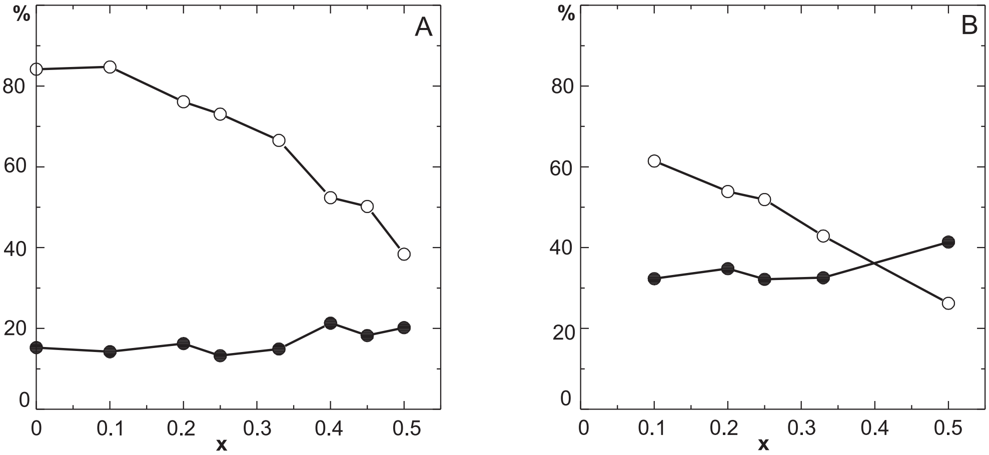


Figure 10. *Fractions of relaxed (5-, 6-, and 7- member) rings (○) and strained (2-, 3-, and 4- member) rings (●) in xPbO (1−x)SiO_2 glass. A – slow cooling, B – fast cooling*

EXAMPLE 2. Influence of hydrogen reduction on the silica subsystem in $PbOSiO_2$.

Lead-silicate glasses, submitted to the reduction process (*e.g.* in hydrogen atmosphere), undergo dramatic changes of optical and electrical properties, related to the reduction-induced structural reconstruction [22, 23] (Figure 11). In the unreduced lead-silicate glasses some of the SiO_4 tetrahedra vertices are bridged by lead ion. In totally reduced glasses neutral Pb atoms agglomerate, forming metallic granules.

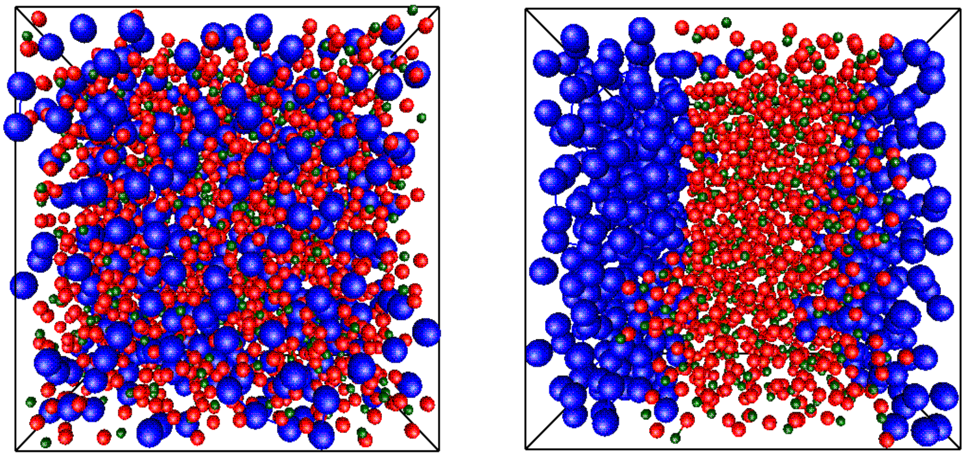


Figure 11. *Atom distributions in unreduced $PbSiO_3$ (left panel) and reduced $PbSiO_2$ (right panel) glasses. Pb – blue, Si – green, O – red. The simulation box contains 500 Pb atoms, 500 Si atoms and 1500 O atoms (1000 O atoms in the reduced glass)*

Both in unreduced, and totally reduced glasses, the regular tetrahedron is a basic structural unit of SiO_2. The overall (calculated for all atoms in the simulation box) angular distribution functions for triples Si–O–Si and Si–Si–Si are somewhat different for unreduced and reduced glasses, but as in the previous example the differences in the middle range order can not be treated quantitatively. However, within the ring analysis one can easily characterise these differences. Figure 12 shows the difference between the rings length distributions for unreduced, and reduced glasses (for $x = 0.1$ and 0.5). For $x = 0.1$, although the ring length statistics is rather similar in both unreduced, and reduced glasses, one can easily see the predominance of longer rings in unreduced material. For $x = 0.5$ the differences are much more pronounced: in the reduced glass there are many more 4- and 6-member rings than in the unreduced one. Simultaneously, in the unreduced glass the long rings are more frequent than in the reduced glass. Figure 13 is analogous to Figure 10. In comparison with the unreduced glasses, the x-dependence of the relative contribution of relaxed rings is much weaker in the reduced glasses.

Due to high contents of O–Pb–O bridges in the unreduced glass, the Si–O–Si–O–… graph is weakly connected. On the other hand, due to the phase separation in the reduced glass this graph is strongly connected. These obvious - from qualitative point of view – facts can be discussed quantitatively using the ANELLI package. In the reduced glass, $PbSiO_2$, a single connected Si–O graph was found. All the DSs in the graph contain only one node: these are apical oxygen atoms adjacent to the lead granules. On the other hand, the graph describing Si–O bonds in the unreduced glass, $PbSiO_3$, splits into many (29) separate connected sub-graphs. Most of them (28) are small groups of atoms: single SiO_4 tetrahedra, or 2–3 corner sharing SiO_4 tetrahedra, and do not contain any ring. However, most of the Si and O atoms (90% of all the Si and O atoms in the simulation box) form one large cluster. Only about 40% of atoms in this connected structure belong to any ring. All the rings are shown in Figure 14, left panel. The remaining 60% belong to dangling structures

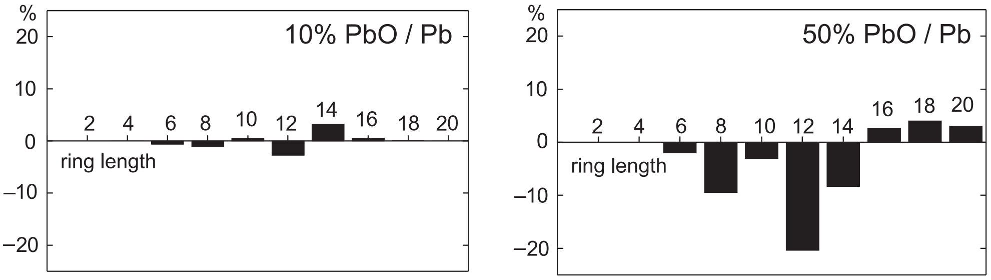


Figure 12. *Difference between Si–O–Si–O–... rings length distributions in unreduced and reduced lead-silicate glasses*

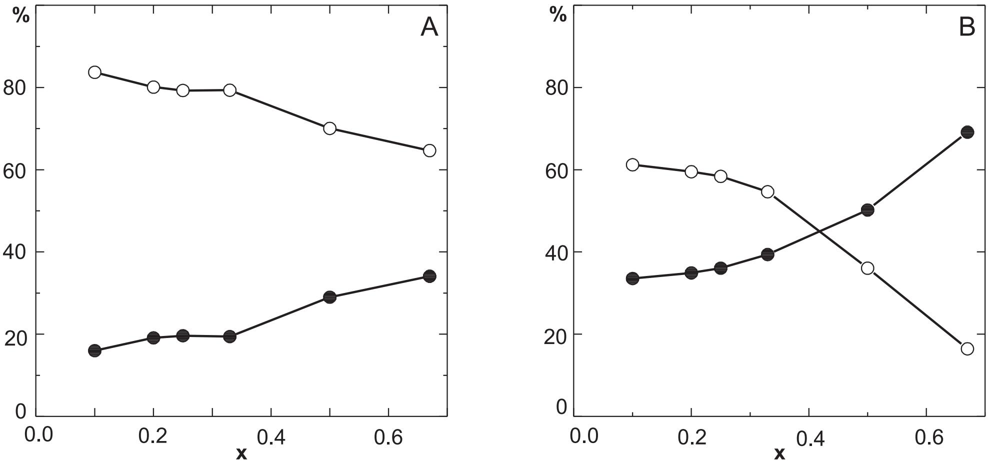


Figure 13. *Fractions of relaxed (○) and strained rings (●) in totally reduced xPbO (1−x)SiO$_2$ glass. A – slow cooling, B – fast cooling*

(82 DS's longer than 1, with average length of about 8 and typically with 2 bifurcations. Several dangling structures are shown in right panel of Figure 14.

4. Concluding remarks

In the paper a new program package ANELLI, designed for the medium range order analysis in computer simulated solids, has been described. Although the presented exemplary results referred to MD-simulated structures, the package applicability is not limited to molecular dynamics methods. One can analyse the particle spatial correlations for any atomic model, independently of the simulation method.

Using the ANELLI package one can find a representative set of rings and investigate their geometrical properties. All the linear structures (not containing any ring) can be identified and various statistical data on the atom chains are accessible. Simple-format files, containing complete data on every ring or chain expressed in the node-number language, can be generated on request. These files can be used to extract any kind of geometrical information according to current needs. Moreover, for any structure of interest a PDB-format file can be generated, enabling the structure direct visualisation by many easily accessible graphic programs.

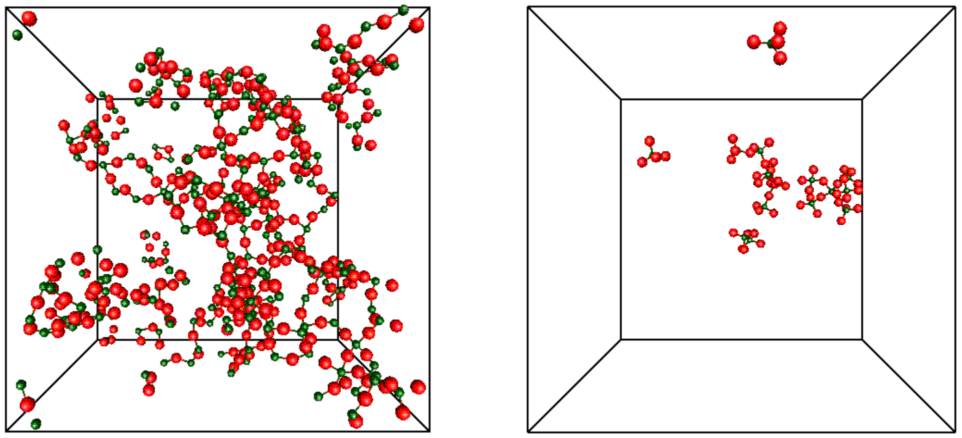


Figure 14. *All the Si–O–Si–O–… rings of lengths from 4 to 20 found in the PbSiO$_3$ structure (left panel) and several groups of Si and O atoms with no rings (right panel)*

In most practical cases, the *anelli* program execution is the most time consuming. For about 10^3 nodes the CPU charge amounts to about 10 minutes of the R10000 processor. For about $3 \cdot 10^4$ particles the ring basis calculation requires about 15 hours. All other programs can be run interactively.

The package has been installed at the TASK Computer Centre and is accessible to all the users. It is also available by the authors. We believe that the ANELLI package can be useful for everyone who works in the field of computer modelling of solid structure.

Acknowledgements

The MD calculations cited in the paper have been performed at the TASK Computer Centre in Gdansk (Poland).

References

[1] Hockney R W and Eastwood J W 1987 *Computer Simulations using Particles*, McGraw-Hill

[2] Allen M P and Tildesley D J 1997 *Computer Simulation of Liquids*, Clarendon Press, Oxford

[3] Rapaport D C 1995 *The Art of Computer Simulation*, University Press, Cambridge

[4] Falken D and Jonsson H 1994 *Comput. Matter. Sci.* **2** 279

[5] Finney J L 1970 *Proc. Roy. Soc. London*, Ser. A **319** 479, *ibid.* 495

[6] Brostow W, Chybicki M, Laskowski R and Rybicki J 1998 *Phys Rev*. **B57** 13448

[7] Rybicki J, Laskowski R and Feliziani S 1996 *Comp. Phys. Commun.* **97** 185

[8] Rybicki J, Laskowski R, Alda W and Mancini G 1996 *Proc. Int. Conf.on Physics Computing*, Cracow, p. 167

[9] Downs G M, Gillet V J, Holliday J D and Lynch M F 1989 *J. Chem. Inf. Comput. Sci.* **29** 172

[10] Balducci R and Pearlman R S 1994 *J. Chem. Inf. Comput. Sci.* **34** 822

[11] Balducci R 1998 (private information)

[12] Mancini G 1997 *TASK Quartertly* **1** 89

[13] Mancini G 2000 *Comp. Phys. Commun.* (submitted)

[14] Rybicka A 1999 *Structure and properties of lead-silicate and lead-germanate glasses: simulations using particles*, PhD Thesis, Technical University of Gdansk, Gdansk

[15] Rybicka A, Chybicki M, Mancini G, Feliziani S and Bergmański G 1997 *Proc. of the Fourth Int. Conf. on Intermolecular Interactions in Matter*, Lublin, p. 42

[16] Rybicka A, Rybicki J, Witkowska A, Mancini G and Feliziani S 1999 *Comp. Meth. Sci. Techn.* **5** 67

[17] http://www.rcsb.org/pdb/

[18] Alda W 1999 unpublished

[19] Andersen H C 1980 *J. Chem. Phys.* **72** 2384

[20] Damodaran K V, Rao B G and Rao K J 1990 *Chem. Phys. Glasses* **31** 212

[21] Hamann D R 1997 *Phys. Rev.* **B 55** 14784

[22] Pan Z, Henderson D O and Morgan S H 1994 *J. Non-Cryst. Solids* **171** 134

[23] Witkowska A, Rybicki J, Trzebiatowski K, Di Cicco A and Minicucci M 2000 *J. Non-Cryst. Solids* **276** 19