Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Physics Communications 165 (2005) 59–96
Computer Physics Communications
www.elsevier.com/locate/cpc

ARVO: A Fortran package for computing the solvent accessible surface area and the excluded volume of overlapping spheres via analytic equations ☆

Ján Buša [a,b,c], Jozef Džurina [b,c], Edik Hayryan [a,c], Shura Hayryan [a], Chin-Kun Hu [a,*], Ján Plavka [b,c], Imrich Pokorný [b,c], Jaroslav Skřivánek [b,c], Ming-Chya Wu [a]

[a] *Institute of Physics, Academia Sinica, Nankang, Taipei 11529, Taiwan*
[b] *Department of Applied Mathematics, Technical University in Košice, 040 01 Košice, Slovak Republic*
[c] *Laboratory of Computing Technology and Automation, Joint Institute for Nuclear Research, Dubna, Russia*

Received 25 September 2003; accepted 4 August 2004

Available online 2 November 2004

Abstract

In calculating the solvation energy of proteins, the hydration effects, drug binding, molecular docking, etc., it is important to have an efficient and exact algorithms for computing the solvent accessible surface area and the excluded volume of macromolecules. Here we present a Fortran package based on the new exact analytical methods for computing volume and surface area of overlapping spheres. In the considered procedure the surface area and volume are expressed as surface integrals of the second kind over the closed region. Using the stereographic projection the surface integrals are transformed to a sum of double integrals which are reduced to the curve integrals. MPI Fortran version is described as well. The package is also useful for computing the percolation probability of continuum percolation models.

Program summary

Title of program: ARVO
Catalogue identifier: ADUL
Program summary URL: http://cpc.cs.qub.ac.uk/summaries/ADUL
Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland
Operating system under which the program has been tested: LINUX system and Windows system

☆ This paper and its associated computer program are available via the Computer Physics Communications homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).

* Corresponding author.
E-mail addresses: jan.busa@tuke.sk (J. Buša), jozef.dzurina@tuke.sk (J. Džurina), edik.hayryan@jinr.ru (E. Hayryan), shura@phys.sinica.edu.tw (S. Hayryan), huck@phys.sinica.edu.tw (C.-K. Hu), jan.plavka@tuke.sk (J. Plavka), imrich.pokorny@tuke.sk (I. Pokorný), jaroslav.skrivanek@tuke.sk (J. Skřivánek), mcwu@phys.sinica.edu.tw (M.-C. Wu).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2004.08.002

Programming language used: FORTRAN
Computer: PC Pentium; SPP'2000;
Number of bytes in distributed program, including test data, etc.: 322 633
Number of lines in distributed program, including test data, etc.: 5051
Distribution format: tar.gz
Card punching code: ASCII
Nature of physical problem: Molecular mechanics computations, continuum percolations.
Method of solution: Numerical algorithm based on the analytical formulas, after using the stereographic transformation.
Restriction on the complexity of the problem: The program does not account explicitly for cavities inside the molecule.
Typical running time: Depends on the size of the molecule under consideration.
Unusual features of the program: No

Keywords: Solvent accessible area; Excluded volume; Proteins; Analytical equations; Stereographic projection; Fortran package; Percolation probability

1. Introduction

To perform its biological function the protein molecule has to fold into a certain spatial (tertiary) structure, called "native structure". This structure is a result of many types of competing intramolecular and intermolecular physical interactions and is believed to be the structure of global minimum of the free energy. The most interesting feature of the folding phenomenon is that the proteins with the given primary structure (aminoacid sequence) almost always fold to the same native structure from any disordered conformation when the environmental parameters (temperature, solvent composition, etc.) are restored to their physiological level. In other words, the tertiary structure is encoded in the primary structure by genetic information. There are two basic problems which are most interesting from point of view of physics: (i) determination of the tertiary structure with a given aminoacid sequence (*structure prediction problem*); and (ii) understanding the kinetic pathways leading to the native structure (*protein folding problem*). The usual way to solve the first problem is to try to minimize the free energy function against the atomic coordinates of the molecule while the second one needs exploration of the very complicated energy profile. In both cases the crucial point is to define the energy function of the protein–solvent system as fully and as exactly as possible. Since the protein molecule is designed by nature to function in the water solution the interactions between protein atoms and the surrounding solvent particles play a special role. Unfortunately, the exact definition of the potential of protein–solvent interactions still remains a difficult problem. Solution of the corresponding explicit equations is impossible because of the tremendous number of degrees of freedom of the protein–water system. For this reason different kinds of approximations are used to model the energy function which help to simplify the simulations.

In so-called "explicit solvent" models thousands of water molecules are involved and molecular dynamics or Monte Carlo simulation is performed. This approach is more exact but computationally very expensive. In "continuum solvent" approximation the water is modeled by some averaged medium with continuous electrostatic properties. One of this kind of models is the atomic solvation parameters approach proposed by Eisenberg and McLachlan [1], in which it has been assumed that the solvation energy of atoms or atomic groups is proportional to the area of the part of atomic surface exposed to the solvent. The total solvation energy of the whole protein–solvent system is then the sum of individual contributions from all atoms:

$$\Delta G = \sum_i \sigma_i A_i, \tag{1}$$

where A_i and σ_i are, respectively, the conformation-dependent solvent accessible surface area and atomic solvation parameter of the atomic group i; and σ_i can be determined from experiments with model compounds which have low molecular weight.

The notion of the solvent accessible area was introduced in [2] in an effort to calculate quantitatively the hydrophobic burial [3] of the protein side chains into the solvent. The accessible surface is a locus of the center of probe sphere when it rolls over the Van der Vaals surface of the molecule. It may be considered as a Van der Waals surface of a system in which all atomic radii are increased by the probe radius. The union of these expanded overlapping atomic spheres was called excluded volume [4]. It represents an envelope enclosed by the accessible surface. The study of the accessible surface area and the excluded volume is very important in computing the protein–water interaction energy [1], theory of gases and liquids [5], in drug binding problem, etc.

The problem of computing volume and the surface area of the union of overlapping spheres was approached both numerically [6–14] and analytically [4,15–21]. The molecular surface was first computed by Greer and Bush [22]. The list of the related literature is so vast that it is beyond our ability to cite even the most important of them. We strongly recommend the interested reader to read a review paper with comprehensive bibliography prepared by Conolly [23], the author of [24,25]. The package GEPOL by Silla et al. [26,27] for computing the molecular area and volume is referred there. At the website of Ref. [28], one can find Conolly's molecular surface package presentation. The basic tool for analytic surface calculations traditionally has been the global Gauss–Bonnet theorem [4,29].

In our recent work [30] we have proposed a new approach for analytic surface calculations using a simple stereographic projection method [29] which leads to more simple formulas and allows to reduce the computations. In this work we extend the developed method to calculate the excluded volume as well [31]. We also present the corresponding Fortran code. Note, that the idea of stereographic projection was used also in [24] albeit in a different aspect.

This paper is organized as follows. In Section 2, we present equations for computing the volume of overlapping spheres. In Section 3, we briefly review equations for computing the surface area of overlapping spheres [30]. In Section 4, we introduce the content of the FORTRAN package: **ARVO**. In Section 5, we present a package **ACCAR** for computing the surface area of overlapping spheres and its derivatives with respect to coordinates of spheres. In Section 6, we present some examples about using **ARVO**. In Section 7, we discuss potential applications of **ARVO**, including simulation of proteins and continuum percolation models. In Appendix A, we present typical outputs from **ACCAR** and that of **GETAREA** [32,33] for the peptide with PDB code *1j4m*, which show that two outputs are consistent.

2. Analytical method for computing the volume

We describe the molecule M as a union of n spheres (atoms) $S_1, \ldots, S_n$, i.e., $M = \bigcup_{j=1}^{n} S_j$. Let (x_i, y_i, z_i) be Cartesian coordinates of the center of the ith sphere and r_i be its radius, where $1 \leqslant i \leqslant n$. For $j \neq i$ we say that S_j is *a neighbor of* S_i if $\mathrm{In}(S_i) \cap \mathrm{In}(S_j) \neq \emptyset$, where $\mathrm{In}(S)$ denotes the interior of the set S.

We will compute the volume according the following scheme

$$\text{Volume integral} \Longrightarrow \text{Surface integral} \Longrightarrow \text{Double integral} \Longrightarrow \text{Line integral}$$

or

$$V(M) = \iiint_{V(M)} \mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z = \iint_{B(M)} z\,\mathrm{d}x\,\mathrm{d}y = \sum_{i=1}^{n} \iint_{B_i(M)} z\,\mathrm{d}x\,\mathrm{d}y, \tag{2}$$

where $B(M)$ is the boundary (surface) of M and $B_i(M)$ is a part of the surface of S_i which is *outside of all its neighbors* ("free" surface of the sphere S_i). Here the Gauss–Ostrogradsky theorem was used to reduce the evaluation of volume $V(M)$ to the surface integrals of the second kind. All integrals at the right-hand side of Eq. (2) can be calculated separately. At this point the problem of computing $V(M)$ is reduced to computing n surface integrals.

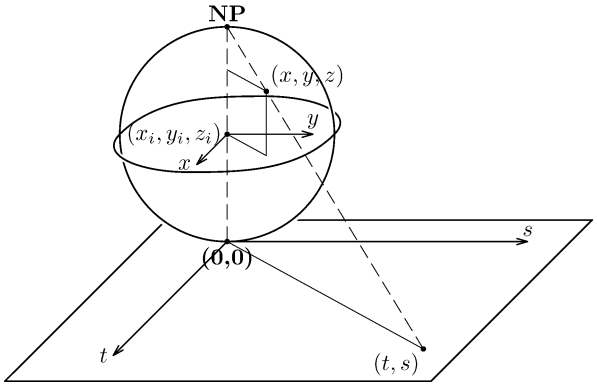

Fig. 1. Stereographic projection of the spherical surface points onto the tangential plane.

Next step is the transformation of the surface integral over the particular surface $B_i(M)$ into the double integral. This can be done by projecting the surface $B_i(M)$ from some top point of the sphere (North Pole) into the plane tangent to the sphere at the diametrically opposite point (the South Pole of S_i) [30].

2.1. Stereographic projection

The points (x, y, z) on the surface of ith sphere satisfy the equation

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = r_i^2. \tag{3}$$

One can easily calculate from Fig. 1 that the point (x_i, y_i, z_i) on the ith sphere is projected from the top point (NP—North Pole) of the sphere onto the point $(t, s) \in \mathbb{R}^2$ through the following equations

$$t = -\frac{2r_i(x - x_i)}{z - z_i - r_i}, \qquad s = -\frac{2r_i(y - y_i)}{z - z_i - r_i}. \tag{4}$$

This an one-to-one transformation except the top point $(x_i, y_i, z_i + r_i)$ itself. It follows from Eqs. (3) and (4) that the inverse transformation can be written as

$$x = x_i + \frac{4r_i^2 t}{t^2 + s^2 + 4r_i^2}, \qquad y = y_i + \frac{4r_i^2 s}{t^2 + s^2 + 4r_i^2}, \qquad z = z_i + r_i - \frac{8r_i^3}{t^2 + s^2 + 4r_i^2}. \tag{5}$$

The points which are not inside the jth sphere satisfy the following inequality

$$(x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2 \geqslant r_j^2. \tag{6}$$

The points of the ith sphere's surface which are outside of the jth sphere or on its surface, satisfy Eqs. (3) and (6). Transformation of those points onto (t, s)-plane using Eq. (5) leads to

$$a_j^i(t^2 + s^2) + b_j^i t + c_j^i s + d_j^i \geqslant 0, \tag{7}$$

where

$$\begin{aligned} a_j^i &= (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i + r_i - z_j)^2 - r_j^2, \\ b_j^i &= 8r_i^2(x_i - x_j), \\ c_j^i &= 8r_i^2(y_i - y_j), \\ d_j^i &= 4r_i^2\big[(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - r_i - z_j)^2 - r_j^2\big]. \end{aligned} \tag{8}$$

Let us denote by Ω_i the set of the points on (t,s)-plane which correspond to $B_i(M)$ by the stereographic projection and by $\mathcal{C}_i$ the complementary to the index i in the set of indices, i.e. $\mathcal{C}_i = \{1,2,\ldots,n\}\backslash\{i\}$. Then

$$\Omega_i = \left\{(t,s);\ a_j^i(t^2+s^2) + b_j^i t + c_j^i s + d_j^i \geqslant 0 \text{ for all } j \in \mathcal{C}_i\right\}. \tag{9}$$

It is obvious that in practical calculations we may use in Eq. (9) only the set of indices $\mathcal{N}_i = \{j \in \mathbb{N};\ S_j \text{ is a neighbor of } S_i\} \subset \mathcal{C}_i$ instead of $\mathcal{C}_i$. So, $\mathcal{N}_i$ is a set of indices of neighbors of S_i.

Let us denote the (t,s)-image of the points of the ith sphere which do not fall inside the jth sphere as $\Im_j^i$. Several different types of $\Im_j^i$ are possible [30]. (a) If the ith sphere is the subset of the jth sphere then the image $\Im_j^i$ is an empty set $\emptyset$. (b) If the ith sphere touches the jth sphere at a single point from inner side then the image $\Im_j^i$ consists of only one point. In both (a) and (b) the corresponding volume and area will be zero. When the ith sphere touches the jth sphere from outside then $\Im_j^i$ is the whole (t,s)-plane excluding one point. (c) If the NP of the ith sphere lies inside the jth sphere then $\Im_j^i$ is the interior of the circle, which is the image of the intersection circle of the ith and the jth spheres in 3D space. (d) When the NP lies on the intersection circle of the ith and the jth spheres, the image of the intersection circle is a straight line and $\Im_j^i$ is a half plane (see Remark 1 below). (e) If the NP of the ith sphere lies outside the jth sphere then $\Im_j^i$ is the interior of a circle. And, finally, (f) if the ith sphere lies outside the jth one then $\Im_j^i$ is the whole plane. Corresponding volume and area are the volume and area of the whole ith sphere, respectively.

So it is easy to see that if S_i has no neighbors then $\Omega_i = \mathbb{R}^2$ and $B_i(M)$ is the whole surface of S_i and the corresponding surface integral is equal to $(4/3)\pi r_i^3$ (the volume of sphere). On the other hand, if the whole *surface* of the sphere S_i is the subset of the union of its neighbors, $S_i \subset \bigcup_{j\in\mathcal{N}_i} S_j$, then $B_i(M) = \emptyset$, $\Omega_i = \emptyset$ and the integral equals to zero. We need to calculate the integrals only if the ith sphere has a (nontrivial) neighbor and is not a subset of some other sphere. So we need deal only with cases (c), (d), and (e).

Remark 1. Using the rotations of the whole molecule one can avoid the case when NP of some sphere lies on the surface of some other sphere. This approach gives us the possibility to work only with cases (c) and (e). In both cases the boundaries of $\Im_j^i$ will be circles. In the following let us consider only such situation. The general case is discussed in [31].

Since inequality (7) represents either interior of a circle ($a_j^i < 0$) or exterior of a circle ($a_j^i > 0$) then Ω_i is an intersection of such parts of (t,s)-plane (see Figs. 2 and 4).

2.2. Computation of the integral $\iint_{B_i(M)} z\,\mathrm{d}x\,\mathrm{d}y$

For computing the surface integrals in Eq. (2) we will use the known formula which transforms the surface integral into double integral. In view of Eq. (5) we have the Jacobian

$$\mathcal{J}_i(t,s) = \begin{vmatrix} \frac{\partial x}{\partial t} & \frac{\partial x}{\partial s} \\ \frac{\partial y}{\partial t} & \frac{\partial y}{\partial s} \end{vmatrix} = 16 r_i^4 \frac{4r_i^2 - t^2 - s^2}{(t^2+s^2+4r_i^2)^3}.$$

Consequently,

$$\begin{aligned} \iint\limits_{B_i(M)} z\,\mathrm{d}x\,\mathrm{d}y &= -\iint\limits_{\Omega_i} \left[z_i + r_i - \frac{8r_i^3}{t^2+s^2+4r_i^2}\right] \mathcal{J}_i(t,s)\,\mathrm{d}t\,\mathrm{d}s \\ &= 128 r_i^7 \iint\limits_{\Omega_i} \left[\frac{\partial Q(t,s)}{\partial t} - \frac{\partial P(t,s)}{\partial s}\right] \mathrm{d}t\,\mathrm{d}s = \mathcal{I}_i, \end{aligned} \tag{10}$$

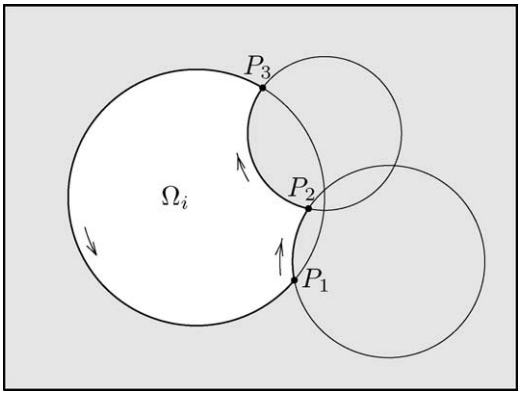

Fig. 2. The bounded plane image Ω_i of the surface part $B_i(M)$.

where

$$Q(t,s) = \frac{t}{3(t^2+s^2+4r_i^2)^3} + \frac{t}{192r_i^4(t^2+s^2+4r_i^2)} + \frac{t}{(t^2+s^2+4r_i^2)^2}\left[\frac{1}{48r_i^2} - \frac{z_i+r_i}{16r_i^3}\right],$$

$$P(t,s) = \frac{-s}{3(t^2+s^2+4r_i^2)^3} + \frac{-s}{192r_i^4(t^2+s^2+4r_i^2)} + \frac{-s}{(t^2+s^2+4r_i^2)^2}\left[\frac{1}{48r_i^2} - \frac{z_i+r_i}{16r_i^3}\right].$$

The Green's theorem says that

$$\iint_{(S)} \left[\frac{\partial Q}{\partial t} - \frac{\partial P}{\partial s}\right] \mathrm{d}t\,\mathrm{d}s = \int_{(K)} P\,\mathrm{d}t + Q\,\mathrm{d}s, \tag{11}$$

where functions $P(t,s)$ and $Q(t,s)$ and their partial derivatives are continuous function, (K) is the positive (counter clockwise) oriented piecewise smooth boundary of the region (S).

We first assume that Ω_i is *bounded* (cf. Fig. 2). Applying Green's theorem to Eq. (10) we transform the double integral into the curve integrals in the following way

$$\mathcal{I}_i = \frac{128r_i^7}{3} \oint_{B(\Omega_i)} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{(t^2+s^2+4r_i^2)^3} + \frac{2r_i^3}{3} \oint_{B(\Omega_i)} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{(t^2+s^2+4r_i^2)} - \frac{8r_i^4(3z_i+2r_i)}{3} \oint_{B(\Omega_i)} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{(t^2+s^2+4r_i^2)^2}, \tag{12}$$

where $B(\Omega_i)$ is the boundary of Ω_i. Fig. 2 corresponds to the case, when some sphere S_i has three neighbors. So the number of indices in $\mathcal{N}_i$ is $|\mathcal{N}_i| = 3$. Each sphere S_{j_k}, $j_k \in \mathcal{N}_i$, $k = 1, 2, 3$ (k is the label of circles in Fig. 2) surface has the nontrivial intersection with the sphere S_i. This intersection is the circle in 3D space. The stereographic image of this circle is again a circle in the (t,s)-plane (see Fig. 2) described by the equation

$$a_j^i(t^2+s^2) + b_j^i t + c_j^i s + d_j^i = 0, \quad \text{for some } j \in \mathcal{N}_i, \tag{13}$$

where $(a_j^i \neq 0)$. Eq. (9) implies that Ω_i is the intersection of the domains with circular boundaries (disks or complements to disks), hence, $B(\Omega_i)$—the boundary of Ω_i—consists of circular arcs.

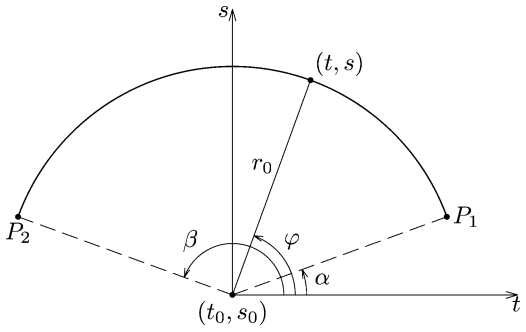

Fig. 3. Parameterization of a circular arc in the (t,s)-plane.

2.3. Computation of $\oint_{B(\Omega_i)} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{(t^2+s^2+4r_i^2)^k}$ along the circular arcs

The boundary $B(\Omega_i)$ consists of circular arcs with the ends at the intersection points of circles in the (t,s)-plane (Fig. 2). If some circle has no intersection with other circles then we consider the whole circle as one arc, which may be a part of the boundary $B(\Omega_i)$. In Figs. 2 and 4 each circle has only one arc from the boundary $B(\Omega_i)$. But more than one arcs are possible on the same circle which come from one boundary $B(\Omega_i)$ (see both pictures in Fig. 5). Let Λ_j^i be the number of arcs which generate partially the boundary of Ω_i and descend from the jth sphere. Then all arcs $C_{j,\lambda}^i$ together form the boundary of Ω_i. Circular arcs $C_{j,\lambda}^i$ are oriented positively (counter clockwise) with respect to Ω_i if $a_j^i < 0$ and negatively (clockwise), otherwise. Then we have

$$\oint_{B(\Omega_i)} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{(t^2+s^2+4r_i^2)^k} = \sum_{j\in\mathcal{N}_i} \sum_{\lambda=1}^{\Lambda_j^i} \int_{C_{j,\lambda}^i} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{(t^2+s^2+4r_i^2)^k}, \quad k = 1,2,3. \tag{14}$$

In order to simplify the formulas in the following we will omit the upper index i, except the cases when it may cause misunderstanding. To compute the volume $V(M)$, it is sufficient to give formulas for the following curve integrals:

$$J_k = \int_{C_{j,\lambda}} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{(t^2+s^2+4r_i^2)^k}, \quad k = 1,2,3$$

(here we omit the indices j,λ in J_k, too). Since $C_{j,\lambda}$ is a circular arc given by Eq. (13) (where $a_j \neq 0$) then $C_{j,\lambda}$ is parameterized as follows (see Fig. 3, the indices are omitted):

$$\begin{aligned} t &= t_0 + r_0\cos\varphi, \\ s &= s_0 + r_0\sin\varphi, \end{aligned} \quad \text{for } \varphi \in \langle \alpha_{j,\lambda}; \beta_{j,\lambda} \rangle, \tag{15}$$

where (t_0, s_0) and r_0 are the center and radius of the corresponding circle.

After some computations we arrive at the following relations.

$$\begin{aligned} J_1 &= \frac{\beta_{j,\lambda} - \alpha_{j,\lambda} + (r_0^2 - A)I_1}{2}, \\ J_2 &= \frac{I_1 + (r_0^2 - A)I_2}{4}, \qquad J_3 = \frac{I_2 + (r_0^2 - A)I_3}{8}, \end{aligned} \tag{16}$$

where

$$I_k = \int_{\alpha_{j,\lambda}}^{\beta_{j,\lambda}} \frac{\mathrm{d}\varphi}{(A + B\cos\varphi + C\sin\varphi)^k}, \quad k = 1, 2, 3$$

with

$$A = \frac{4r_i^2 + t_0^2 + s_0^2 + r_0^2}{2}, \qquad B = t_0 r_0, \qquad C = s_0 r_0,$$

and

$$t_0 = -\frac{b_j}{2a_j}, \qquad s_0 = -\frac{c_j}{2a_j}, \qquad r_0 = \sqrt{\frac{b_j^2 + c_j^2 - 4a_j d_j}{4a_j^2}}. \tag{17}$$

If we denote

$$D = A^2 - B^2 - C^2$$

then one can verify that for the case when $\beta_{j,\lambda} - \alpha_{j,\lambda} < 2\pi$ the following formulas hold

$$I_1 = \frac{2}{\sqrt{D}}\left[\frac{\pi}{2} - \arctan\frac{A\cos^- + B\cos^+ + C\sin^+}{\sqrt{D}\sin^-}\right],$$

where

$$\cos^- = \cos\frac{\beta_{j,\lambda} - \alpha_{j,\lambda}}{2}, \qquad \cos^+ = \cos\frac{\alpha_{j,\lambda} + \beta_{j,\lambda}}{2},$$
$$\sin^+ = \sin\frac{\alpha_{j,\lambda} + \beta_{j,\lambda}}{2}, \qquad \sin^- = \sin\frac{\beta_{j,\lambda} - \alpha_{j,\lambda}}{2},$$
$$I_2 = \frac{1}{A^2 - B^2 - C^2}\left[\frac{-B\sin x + C\cos x}{A + B\cos x + C\sin x}\Bigg|_{\alpha_{j,\lambda}}^{\beta_{j,\lambda}} + AI_1\right],$$
$$I_3 = \frac{1}{2D}\left[\frac{-B\sin x + C\cos x}{(A + B\cos x + C\sin x)^2}\Bigg|_{\alpha_{j,\lambda}}^{\beta_{j,\lambda}} + \frac{\frac{-B}{A}\sin x + \frac{C}{A}\cos x}{A + B\cos x + C\sin x}\Bigg|_{\alpha_{j,\lambda}}^{\beta_{j,\lambda}}\right] + \frac{2A^2 + B^2 + C^2}{2AD}I_2.$$

For the case $\beta_{j,\lambda} - \alpha_{j,\lambda} = 2\pi$ the integrals I_1, I_2, I_3 are given by:

$$I_1 = \frac{2\pi}{D^{1/2}}, \qquad I_2 = \frac{2\pi A}{D^{3/2}}, \qquad I_3 = \frac{\pi(2A^2 + B^2 + C^2)}{D^{5/2}}.$$

For I_k there exist explicit formulas (see, for example, [34]).

In the case when Ω_i is unbounded, $\Omega_i^c = \mathbb{R}^2 - \Omega_i$ is bounded (cf. Fig. 4) and we can use the following equality

$$16r_i^4 \iint_{\Omega_i}\left[\frac{\partial Q(t,s)}{\partial t} - \frac{\partial P(t,s)}{\partial s}\right]\mathrm{d}t\,\mathrm{d}s + 16r_i^4 \iint_{\Omega_i^c}\left[\frac{\partial Q(t,s)}{\partial t} - \frac{\partial P(t,s)}{\partial s}\right]\mathrm{d}t\,\mathrm{d}s = \frac{4}{3}\pi r_i^3,$$

for computing the surface integral in Eq. (10). Computation of integrals over Ω_i^c leads to the same curve integrals, given by Eq. (12), but with different curve orientations.

So, we come to the formula

$$V = \sum_{i=1}^{n}\left[\chi_V(\Omega_i) + \mathcal{I}_i\right], \tag{18}$$

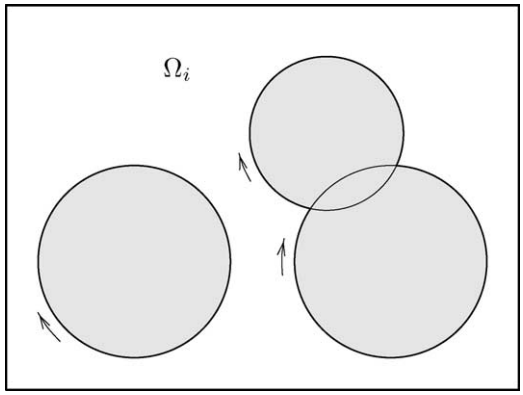

Fig. 4. Unbounded domain Ω_i.

where $\mathcal{I}_i$ is defined above by Eq. (10), and

$$\chi_V(\Omega_i) = \begin{cases} 0, & \Omega_i \text{ is bounded,} \\ \frac{4}{3}\pi r_i^3, & \Omega_i \text{ is all plane except the union of several disks.} \end{cases}$$

Remark 2. If Ω_i is unbounded, the corresponding sum of integrals $\mathcal{I}_i$ will be negative and we get the correct value for the integral over $B(\Omega_i)$ given by Eq. (10).

Remark 3. Similar method can be derived for the calculation of the partial or "free" volume of an atom (see [31]). In other words, let S_1 be an arbitrary sphere. Denote by $S_2, \ldots, S_k$, all nontrivial neighbors of S_1. Let $F = S_1^{2,\ldots,k}$ denote the part of S_1 which is *outside of all its neighbors* $S_2, \ldots, S_k$. We consider volume $V(F)$ as a "free" volume of the atom S_1.

The idea of partial volume may be useful when we add a new atom to the molecule with known volume. In this case we can compute its "free volume", which is simply the volume change of the whole molecule.

3. Computation of surface area

The surface area $A(M)$ is calculated as a surface integral of the first kind

$$A(M) = \iint\limits_{B(M)} |\mathrm{d}\boldsymbol{\sigma}| = \sum_{i=1}^{n} \iint\limits_{B_i(M)} |\mathrm{d}\boldsymbol{\sigma}| = \sum_{i=1}^{n} A_i, \tag{19}$$

where M, $B(M)$ and $B_i(M)$ are described in Section 2. All integrals at right-handed side of Eq. (19) can be calculated separately. The problem of computing $A(M)$ is reduced to computing n surface integrals of the first kind.

Remark 4. In this case one can avoid the straight line boundary by rotation only the ith sphere and its neighbors, unlike the volume computation, where the rotation of the whole molecule is necessary.

3.1. Computation of $\iint_{B_i(M)} |\mathrm{d}\boldsymbol{\sigma}|$

Now we show how the area of the part of the spherical surface can be calculated by integration along the circular arcs on the plane. First we transform the surface integral of the first kind into the double integral over the Ω_i in the (t,s)-plane, next we transform the double integral into the curve integral in a similar way as was done in Section 2.

If we use again the stereographic projection of the surface we can consider t and s as parameters and obtain the following equation for the element of the surface

$$|\mathrm{d}\boldsymbol{\sigma}| = |\mathrm{d}_t\boldsymbol{r} \times \mathrm{d}_s\boldsymbol{r}|, \tag{20}$$

where '$\times$' means vector product, $\boldsymbol{r}$ is the radius vector of the element $\mathrm{d}\boldsymbol{\sigma}$, $\mathrm{d}_t\boldsymbol{r}$ and $\mathrm{d}_s\boldsymbol{r}$ are its differentials with respect to t and s, respectively. We calculate these differentials in the following way, using the transformation formulas (5)

$$\mathrm{d}_t\boldsymbol{r} = \left(\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t}, \frac{\partial z}{\partial t}\right)\mathrm{d}t, \tag{21}$$

$$\frac{\partial x}{\partial t} = \frac{4r_i^2(s^2 - t^2 + 4r_i^2)}{(t^2 + s^2 + 4r_i^2)^2}, \tag{22}$$

$$\frac{\partial y}{\partial t} = -\frac{8r_i^2 ts}{(t^2 + s^2 + 4r_i^2)^2}, \tag{23}$$

$$\frac{\partial z}{\partial t} = \frac{16r_i^3 t}{(t^2 + s^2 + 4r_i^2)^2}. \tag{24}$$

The term $\mathrm{d}_s\boldsymbol{r}$ is calculated in the similar way and eventually we obtain

$$A_i = \iint_{B_i(M)} \mathrm{d}|\boldsymbol{\sigma}| = 16r_i^4 \iint_{\Omega_i} \frac{\mathrm{d}t\,\mathrm{d}s}{(t^2 + s^2 + 4r_i^2)^2}, \tag{25}$$

where the region Ω_i represents (as above) the image of $B_i(M)$ on the plane $(t,s) \in \mathbb{R}^2$.

Suppose Ω_i is a bounded region of nonzero measure, $\mathcal{N}_i$ is the set of order numbers of the spheres which intersect the ith sphere, and Λ_j^i is the number of the arcs which form partially the boundary of Ω_i and ascend from the jth sphere. Now we use again Green's formula to replace the integration over the plane region in Eq. (25) by the sum of the integrals over the circle arcs along the boundary $B(\Omega_i)$ of the region Ω_i.

It is easy to see that if we choose now

$$P(t,s) = 2r_i^2 \frac{-s}{t^2 + s^2 + 4r_i^2}, \qquad Q(t,s) = 2r_i^2 \frac{t}{t^2 + s^2 + 4r_i^2},$$

then we obtain from Eq. (25) in the same way as in Section 2

$$A_i = 2r_i^2 \oint_{B(\Omega_i)} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{t^2 + s^2 + 4r_i^2}, \tag{26}$$

where as above $B(\Omega_i)$ is the boundary of Ω_i.

Like in Eq. (14) we get

$$\oint_{B(\Omega_i)} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{t^2 + s^2 + 4r_i^2} = \sum_{j\in\mathcal{N}_i} \sum_{\lambda=1}^{\Lambda_j^i} \int_{C_{j,\lambda}^i} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{t^2 + s^2 + 4r_i^2} = \sum_{j\in\mathcal{N}_i} \sum_{\lambda=1}^{\Lambda_j^i} J_{1\,j,\lambda}^{\,i}. \tag{27}$$

So we come again to the same integrals which we deal with in Section 2, where we denoted the integrals as J_1 (here we omit the indices, too).

An unbounded area Ω_i forms the whole plane except some disks union. In this case integration over the boundary is taken in the negative direction (the sum of integrals $\int_{C^i_{j,\lambda}} 2r_i^2(t\,\mathrm{d}s - s\,\mathrm{d}t)/(t^2+s^2+4r_i^2)$ is negative), and the result is added to the area of the whole sphere $4\pi r_i^2$, as in the volume computation over the unbounded domain. So, the general formula for surface area is similar to the formula (18) for the volume

$$A = \sum_{i=1}^{n}\left[\chi_A(\Omega_i) + 2r_i^2 \sum_{j\in\mathcal{N}_i}\sum_{\lambda=1}^{N_j}\int_{C_{j,\lambda}} \frac{t\,\mathrm{d}s - s\,\mathrm{d}t}{t^2+s^2+4r_i^2}\right], \tag{28}$$

where

$$\chi_A(\Omega_i) = \begin{cases} 0, & \Omega_i \text{ is bounded,} \\ 4\pi r_i^2, & \Omega_i \text{ is all plane except the union of several disks.} \end{cases}$$

Remark 5. The integrals in formula (28) have been calculated in the volume computation. So we can use about the same computing time to get both the volume and the area values.

4. Program components and description of the algorithm

The main program ARVO is a simple module, which uses 17 functions and subroutines. Below we give the description of these modules.

4.1. The program structure

The logical structure of the main module ARVO is simple. After reading input data, we first study the neighborhood relations of the spheres/atoms. Here we construct some useful lists, which allow us to work further only with the local subsets of atoms. Before starting the basic calculations the algorithm looks for "bad" NP points. If there is some then the whole molecule is rotated by a random angle and checked again until there are no problematic poles. If there is no North Pole problem then a loop is started over all spheres in which the corresponding integrals are calculated (see the main module in Section 6). The following scheme in Table 1 shows how the subroutines and the functions are nested.

4.2. Input and output data

Program reads the data from the ASCII file `input.dat`, in which the ith line contains at its beginning four real numbers—three Cartesian coordinates of the center and the radius of the ith sphere: x_i, y_i, z_i, and its radius r_i, where $1 \leqslant i \leqslant n$.

The output is written to the ASCII file `output.dat`; each record of which contains the values of the surface area, the volume and the number of the atom.

4.3. Important parameters

`ks=300`— maximum number of spheres (atoms).
`kl=300`— maximum number of local circles in the (t,s)-plane for one atom.
`ka=2000`— maximum total number of arcs and angles which arise from the local circles intersections.

Table 1
Subroutines and functions embedment

```
read input data

subroutine make_neighbors
   integer function neighbors

integer function North_Pole_test
   subroutine spheres_rotation

for all spheres call:
subroutine areavolume
   subroutine local_spheres
   subroutine make_ts_circles
   integer function circles_to_arcs
      integer function new_arcs
         subroutine circles_intersection
         integer function circle_in_circle
         integer function point_in_circle
         subroutine mysort
         subroutine mydsort
         integer function delete_equal
   subroutine avintegral
      real*8 function fract
```

`ki=10000`— maximum number of indices in the `neighbors_indices` array for all atoms; this number must be at least equal to the total number of neighborhood relations multiplied by 2.
`rwater/0d0/`— radius of solvent particles.
`eps_nord_pole=1d-8`— the critical value for North Pole test; if the smallest distance from the North Poles to the surface of other atoms is smaller than `eps_nord_pole`, the molecule is rotated by a random angle.
`eps_deltat=1d-12`— the critical value for comparison of t_1 and t_2 coordinates of two circles in the (t, s)-plane, when the intersection points are calculated in the `circles_intersection` subroutine.
`eps_angle=1d-12`— the critical value for comparison of two angles in `delete_equal` function; if two points on the circle are close to each other, they are declared to be equal and only one point is left.

4.4. Important variables and data structures

`spheres(ks,4)`— contains the data of all atoms: `spheres(i,1)` $= x_i$, `spheres(i,2)` $= y_i$, `spheres(i,3)` $= z_i$, and `spheres(i,4)` $= r_i$ for $1 \leqslant i \leqslant k_s$.
`neighbors_number(ks)`— `neighbors_number(`i`)` is equal to the number of neighbors of the ith sphere for $1 \leqslant i \leqslant k_s$.
`index_start(ks)`— `index_start(i)` is the order number of the index for the first neighbor of the $i$the sphere in the `neighbor_indices` array (see also Section 4.6).
`neighbors_indices(ki)`— contains the indices of neighbors for all atoms (see also Section 4.6).
`sphere_local(kl,4)`— contains coordinates and radius of the ith sphere and its neighboring spheres: `sphere_local(j,1)` $= x_j$, `sphere_local(j,2)` $= y_j$, `sphere_local(j,3)` $= z_j$, and `sphere_local(j,4)` $= r_j$.
`circles(kl,4)`— the data structure of the circles in (t, s)-plane defined by Eq. (13), see Figs. 2 and 4. The corresponding center points and radii of the circles are calculated according to Eqs. (8) and (17). Then `cir-`

`cles(j,1)` $= t_j$ and `circles(j,2)` $= s_j$ are the jth circle's center coordinates, `circles(j,3)` $= r_j$ is its radius, and `circles(i,4)` $= \pm 1$ shows the orientation of the circle.

`arcs(ka,3)` — holds the circular arcs $C^i_{j,\lambda}$ composing the boundary $B(\Omega_i)$ of the (t,s) domain Ω_i. Along these arcs the integrals in Eqs. (14) and (27) are calculated: `arcs(k,1)` $= ic_k$ is the index of the kth arc circle, `arcs(k,2)` $= \alpha_k$ is the starting kth arc's angle, and `arcs(k,3)` $= \delta_k$ is the oriented arc's angle, so the arc's end point corresponds to the angle $\beta_k = \alpha_k + \delta_k$.

`arcsnew(ka,3)` — is the auxiliary array with the same structure as the array `arcs`.

`angles(ka)` — is the auxiliary array to remember all angles corresponding to all intersection points of some circle with all other circles in (t,s)-plane.

`av(2)` — `av(1)` is the value of surface integral which corresponds to the $\mathcal{I}_i$ in Eq. (10) of the volume calculation method; `av(2)` is the value of surface integral corresponding to A_i in Eq. (25) of the surface area calculation method.

4.5. Description of subroutines and functions

In this subsection we give only a brief description of all subroutines and functions (see Table 1). More detailed description of some of them will be given later in the text.

`subroutine make_neighbors` — determination of neighborhood relationship data for all atoms (see Section 4.6).

`integer function neighbors(i,spheres,ind,ks,kl,ns)` — returns the neighbors number for the ith sphere—`neighbors: ns`, and also the indices of the corresponding spheres in array: `ind`.

`integer function North_Pole_test` — returns 1 if all North Poles are far enough from other spheres surfaces, otherwise returns 0 (see also Section 4.7).

`subroutine spheres_rotation` — if the `North_Pole_test` function returns 0, the rotation of the whole molecule is necessary, to avoid straight lines in the boundary of $B(\Omega_i)$.

`subroutine areavolume` — this is the main subroutine for computation of the surface area and the volume of the molecule (see Section 4.8 below).

`subroutine local_spheres(spheres,ind, sphere_local,nls,ks,kl)` — transfers to `sphere_local` array the data in a correspondence to the indices list `ind`.

`subroutine make_ts_circles(sphere_local, circle,kl,nls)` — prepares the circles structure for the 1st sphere of array `sphere_local` in array `circles` according to Eqs. (8) and (17).

`integer function circles_to_arcs` — returns the number of arcs which compose the boundary $B(\Omega_i)$. This function prepares the array `arcs` too (see below Section 4.8.1).

`integer function new_arcs(k,circles,arcsnew,kl,ka,nls)` — prepares arcs, which are the parts of the kth circle in array `circles` which are parts of the boundary $B(\Omega_i)$ (see Section 4.8.1).

`subroutine circles_intersection(ic1,ic2,circles,kl,a1,a2,b1,b2)` — returns angles of two intersection points of circles with indices `ic1` and `ic2` in array `circles`. We use it *only in the case, when 2 different intersection points exist*! Then `a1` and `a2` are corresponding angles with respect to the center of 1st circle, `b1` and `b2` are corresponding angles with respect to the center of 2nd circle.

`integer function circle_in_circle(i,k,circles,kl)` — returns 1 if the ith circle is inside of the kth positive oriented circle or outside of the kth negative oriented circle, and returns 0 otherwise.

`integer function point_in_circle(t,s,k,circles,kl)` — returns 1 if the point (t,s) is inside of the kth positive oriented circle or outside of the kth negative oriented circle and returns 0 otherwise.

`subroutine mysort(angles,ka,num_angle)` — sorts the array `angles` in ascending order (`num_angle` is the angles number).

`subroutine mydsort(angles,ka,num_angle)` — sorts array `angles` in descending order (`num_angle` is the angles number).

`integer function delete_equal(angles,ka,num_angle)` — if some angles in array `angles` are equal (to the precision `eps_angle`) only one such angle is left; the function returns the number of angles after possible deletions.

`subroutine avintegral(circles,arcs,kl,ka,narcs,r1,z1,avi)` — calculates the sum of integrals along all arcs corresponding to the boundary $B(\Omega_i)$ according to Eq. (16); `avi(1)` is the value of the surface integral which corresponds to $\mathcal{I}_i$ in Eq. (10) of the volume calculation method; `avi(2)` is the value of the surface integral which corresponds to A_i in Eq. (25) of the surface area calculation method.

`real*8 function fract` — auxiliary function for some fraction calculation.

4.6. Neighborhood relations

It is useful to have information about neighborhood relations for each atom. This may make the program more clear, and the work with smaller arrays may speed up. Therefore, at the beginning of the ARVO program, we construct these relations.

For the determination of neighbors for all atoms we construct three arrays:

(1) `neighbors_number(i)` is the number of neighbors for the following ith atom.
(2) `index_start(i)` is the first neighbor index for the ith atom in the array `neighbors_indices`.
(3) `neighbors_indices` is the array of neighbors indices for all atoms. The neighbors indices of the ith sphere begin at the position `index_start(i)` and finish at the position `index_start(i)+neighbors_number(i)-1` of the array `neighbors_indices`.

For example let us consider the following configuration of spheres:

the 1st sphere has neighbors with indices 2, 4, 7
the 2nd sphere has neighbors with indices 1, 3
the 3rd sphere has neighbors with indices 2, 4
the 4th sphere has neighbors with indices 1, 3
the 5th sphere is subset of some other sphere
the 6th sphere has no neighbors
the 7th sphere has neighbors with index 1

After the call of subroutine `make_neighbors`, we will have the following arrays:

`neighbors_number=(3,2,2,2,-1,0,1)`,
`index_start=(1,4,6,8,10,10,10,11)` and
`neighbors_indices(2,4,7,1,3,2,4,1,3,1)`.

Remark 6. If a sphere is a subset of some other sphere then `neighbors_number` is set to -1. In the array `index_start`, we add at the end one extra index which represents the starting position for the "next" (not existing at the present) sphere if it is added to the system. One can easily see, that the array `neighbors_number` is in some sense redundant, because except the subset case represented by -1 it holds:

`neighbors_number(i)=index_start(i+1)-index_start(i)`.

4.7. North Pole check and molecule rotation

As explained above, if the North Pole of some ith sphere lies on the surface of some other sphere, the intersection image for such two spheres will be a straight line in the (t, s)-plane tangent to the ith sphere. To avoid this situation, the whole molecule should be rotated. We calculate the minimal distance to the surface of other spheres for each North Pole and compare this minimal value to the `eps_north_pole` value. If we take this value larger, the circles in the (t, s)-plane with very large radius will be forbidden, and the computations may be more exact. The random rotation about the y-axis is made with the random number sa (the user of **ARVO** should call a function to generate the random number), if necessary:

```
      subroutine spheres_rotation(spheres,ks,ns,sa)
c     Random rotation of molecule about the y-axis
c     after bad North Pole test.
c        Some North Pole is near other spheres surface
      implicit real*8 (a-h,o-z)
      dimension spheres(ks,4)

      ca=dsqrt(1d0-sa*sa)
      do i=1,ns
         x=spheres(i,1)
         z=spheres(i,3)
         spheres(i,1)=ca*x-sa*z
         spheres(i,3)=sa*x+ca*z
      enddo
      return
      end
```

4.8. Implementation of the basic algorithm

After the North Pole test we are ready to act according to the algorithm's description given above in Sections 2 and 3. For each sphere/atom with index i the following steps are implemented:

(1) Calling the subroutine `local_spheres`. The array `sphere_local` is prepared, collecting data of the ith sphere and its neighbors.
(2) The procedure `make_ts_circles` defines the circles in (t, s)-plane, according to the formulas (8) and (17).
(3) Intersection points for all circles are calculated using the function `circles_to_arcs`. The list of all arcs which define the boundary of the (t, s) domain Ω_i is constructed (see in more details below in Section 4.8.1).
(4) When all necessary arcs are known call subroutine `avintegral` at the end of the subroutine `areavolume`. All calculations according to Eq. (16) are done (see in more details below in Section 4.8.2).

4.8.1. Preparation of the structure of arcs

Fig. 5 shows the situation after the procedure `make_ts_circles` is finished. At this point all circles and their orientations are known.

Now it is necessary to prepare the structure of the arcs corresponding to the boundary of domain Ω_i. Left picture of Fig. 5 shows the case of bounded Ω_i when there exists at least one positive oriented circle (in this case it is only the largest circle); right picture shows an unbounded Ω_i. In this case the boundary $B(\Omega_i)$ consists only of negative oriented arcs, the domain Ω_i is outside of this boundary.

Using the function `circles_to_arcs` intersection points for all circles are calculated. The intersection points are described by their angles with respect to the center of each circle. The angles are sorted by the ascending

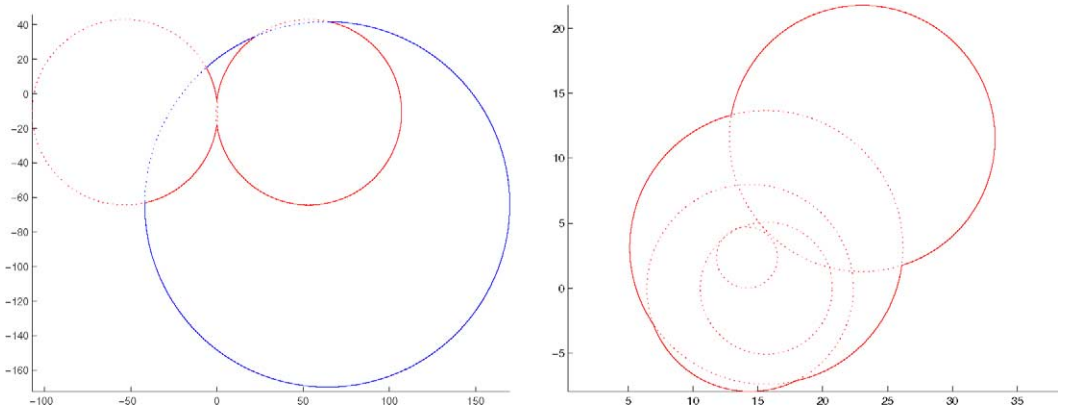

Fig. 5. Generation of the structure of arcs.

order for the positive oriented circles and by the descending order for the negative oriented circles which is done by calling the subroutine `mysort(angles,ka,num_angle)` or `mydsort(angles,ka,num_angle)`, respectively. By this the inheritance of the arcs orientation from orientation of the circles is established. If there exists an intersection point of three or more circles, it should be taken into account only once for each circle. This is done by function `delete_equal(angles,ka,num_angle)`.

The sorted angles define all possible arcs (see Fig. 5). But, certainly, not all of these arcs define the boundary $B(\Omega_i)$. One can easily see, that only in the arcs, which lie outside of all other negative oriented circles *and* inside of all other positive oriented circles are important. So we check the central point of each arc. This is done by the function `new_arcs`. At last we get all arcs of which the boundary $B(\Omega_i)$ consists of in the array `arcs` and we can proceed with evaluation of the curve integral.

4.8.2. Evaluation of the integral

First we check whether the values $\chi_V(\Omega_i)$ in Eq. (18) and $\chi_A(\Omega_i)$ in Eq. (28) are zero or not. If there exists a positive oriented circle in the array `circles` then $\chi_V(\Omega_i) = \chi_A(\Omega_i) = 0$. Otherwise, the values $\chi_V(\Omega_i)$ and $\chi_A(\Omega_i)$ will those described above in Sections 2 and 3. The check is done in the procedure `areavolume` before calling the integral evaluation subroutine `avintegral(circles,arcs,kl,ka,narcs,r1,z1,avi)`, where the values r_1 and z_1 are the corresponding values of the first sphere in the `local_spheres` array.

In the subroutine `avintegral` the formula (16) is used to get the values of corresponding integrals in Eqs. (10), (12), (14) and (26), (27).

5. Module ACCAR for computing derivatives of surface area

In molecular dynamics simulations one needs the derivatives of the potential energy against the coordinates of the particles. It follows from Eq. (1) that calculation of the derivatives (gradients) of the solvation energy requires calculation of the gradients of the accessible area. The method, described in this article allows to obtain these gradients analytically. Evaluation of the corresponding formulas is presented elsewhere [30]. Here we just mention the Fortran module and bring the testing data. Since most simulations do not require computation of the molecular volume we have created a separate module ACCAR which calculates only the accessible area and its gradients against the atomic coordinates. The input of the program is the array of the atomic coordinates, the atomic radii, and the number of atoms. The output contains the total accessible area, the partial accessible area per atom as well as the gradients against the coordinates of each atom. The user can easily organize the input data in his own way. In the presented version the program has two possibilities for this: input from user-created file, and input

from the file compatible with PDB [35] format. In this case the additional module PDBREAD must be called with the corresponding file name. A special version of the subroutine is developed to be used in the environment of SMMP package [36]. The users of SMMP can run it in the way like any other SMMP subroutine is used through CALL instruction. In this case ACCAR takes the atomic coordinates from the internal arrays xat, yat, zat of SMMP. We have tested the accuracy and performance of our module by comparing it with GETAREA module from FANTOM package [21,32,33]. The computations are done for the peptide with PDB code *1j4m*. The results are shown in Appendix A. One can see that both programs give identical results. Note that the online version of GETAREA which we have exploited prints the output numbers with the accuracy of two digits after decimal point. To save space we have omitted all records pertaining to hydrogen atoms because both programs set the radius of the hydrogen atom equal to zero and hence the area and the gradients are identically zero. For this reason the atomic numbers in the first column of the output from GETAREA are not continuously consecutive.

6. Using the program

6.1. The main ARVO module

In this section we bring the text of the main ARVO module with short comments. The module can be easily modified to be included into other programs so that the later can also be used to calculate volume and surface area of overlapping spheres. Parameters, like `ks`, `kl`, `ka`, `ki`, are defined in Section 4.3, arrays are defined in Sections 4.4 and 4.6.

```
      program ARVO
c     Computing surface area and volume of the overlapping spheres
      implicit real*8(a-h,o-z)
      parameter (pi=3.14159265358979323846264d0,ks=300,kl=300,ka=2000,
     1           ki=10000)
c       ks - maximal spheres' number
c       kl - maximal neighbors' number of one sphere (local spheres' number)
c       ka - maximal angles' or arcs' number
c       ki - maximal neighbors' relations' number = cca.
c            spheres' number * maximal neighbors' number
c
c      eps_north_pole - accuracy level in the function North_Pole_test
c      eps_deltat - accuracy level in the subroutine circles_intersection
c      eps_angle - accuracy level in the subroutine delete_equal (angles)
c
      dimension spheres(ks,4),neighbors_number(ks),index_start(ks),
     1          neighbors_indices(ki),av(2)
c
c     spheres(i,1)=xi
c     spheres(i,2)=yi     - ith sphere center coordinates
c     spheres(i,3)=zi
c     spheres(i,4)=ri     - ith sphere radius
c
c     neighbors_number, index_start, neighbors_indices description
c     is given in the   subroutine make_neighbors
c
      data rwater/0d0/
c     Solvent particle's radius
```

```
      open(10,file='output.dat')
      open(unit=11,file='input.dat')
c
c     input spheres data
c
      ns=0
 1      continue
      ns=ns+1
      read(11,*,end=2)spheres(ns,1),spheres(ns,2),spheres(ns,3),
     1              spheres(ns,4)
      spheres(ns,4)=spheres(ns,4)+rwater
        goto 1
 2      continue
      close(11)
      ns=ns-1
c
c     ns - spheres number
c

c     Study the neighborhood relations
      call make_neighbors(1,ns,spheres,neighbors_number,
     1           index_start,neighbors_indices,ks,kl,ns,ki)


c     If some North Pole is close to the other atom's surface
c     molecule's rotation is necessary
      do while (North_Pole_test(1,ns,spheres,neighbors_number,
     1          index_start,neighbors_indices,ks,ki).EQ.0)
      print *,'Rotation after bad North Pole test!'
      write(10,*)'Rotation after bad North Pole test!'
      sa=0.324d0 ! "Random" sin value
      call spheres_rotation(spheres,ks,ns,sa)! random molecule rotation
      enddo

c     Computation of area and volume as a sum of surface integrals
      V=0d0
      A=0d0
      do i=1,ns
      call areavolume(i,spheres,neighbors_number,index_start,
     1                        neighbors_indices,ks,kl,ka,ki,av)
      V=V+av(1)
      A=A+av(2)
      enddo

      print *,'Volume: ',V,' Area: ',A,'  Spheres num: ',ns
c     Writing final result in file res.dat
      write(10,*)'Volume: ',V,' Area: ',A,'  Spheres num: ',ns
      close(10)

      stop 'End'
      end
```

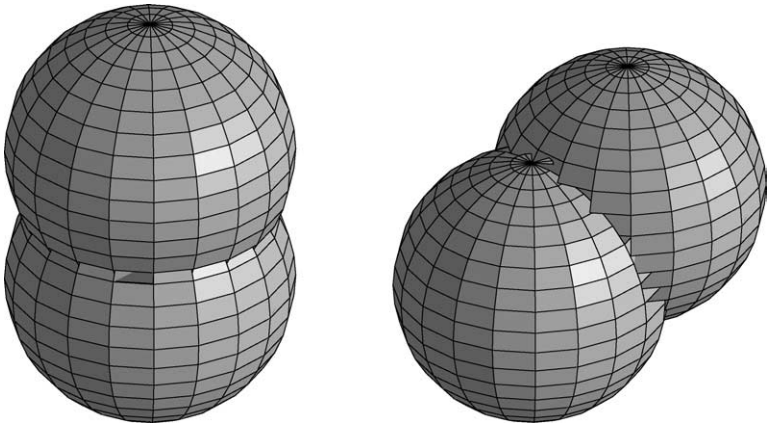

Fig. 6. Different positions of two spheres.

6.2. Running the program—input and output data

6.2.1. An example with two spheres

Here we consider the case of two spheres of equal size with radius r. Let the distance between centers of two spheres be $d = r$, so that the centers of both spheres lie on the surface of another sphere (cf. Fig. 6 left). In this case the exact formulas for the surface area A and the volume V can be easily derived. These are

$$V(r) = \frac{9}{4}\pi r^3 \quad \text{and} \quad A(r) = 6\pi r^2. \tag{29}$$

For $r = 2$ we get

$$V(2) = 18\pi \doteq 56.54866776461628 \quad \text{and} \quad A(2) = 24\pi \doteq 75.39822368615504.$$

The file `input.dat` consists only of two lines

```
0 0 0 2 - 1st sphere: center at (0,0,0) and radius=2
0 0 2 2 - 2nd sphere: center at (0,0,2) and radius=2
```

we have obtained the following results written in the `output.dat` files for PC and SPP'2000 machine, respectively:

```
Volume: 56.54866776461628 Area: 75.39822368615504 Spheres num: 2

Volume: 56.54866776461628 Area: 75.39822368615503 Spheres num: 2
```

In the following, we will give only the results received by SPP'2000 machine, except Fig. 9 whose data were calculated by a PC parallel computer.

Let us compare these results with numerical calculations. The volume was calculated by embedding the union of spheres into the cube, which was divided into n^3 cubic elements, each one of these elements (its center point) was checked for belonging to the molecule. The common volume of the cubes was taken as a numerical approximation of the molecular volume. The surface area was calculated by division of each sphere surface into $2n^2$ parts with equal area. The center of each part was checked for belonging to the molecular surface. The sum the areas of this parts is a numerical approximation to the area value.

Table 2 shows the numerical values for some n, together with their relative errors ε_V and ε_A, respectively.

Table 2
Numerical values for two spheres with equal radii

n	Volume	ε_V	Area	ε_A
10	57.6	$1.86 \cdot 10^{-2}$	72.8849495632832	$3.33 \cdot 10^{-2}$
20	56.448	$1.78 \cdot 10^{-3}$	75.39822368615503	0
50	56.567808	$3.38 \cdot 10^{-4}$	75.11673698439338	$3.73 \cdot 10^{-3}$
100	56.537088	$2.05 \cdot 10^{-4}$	75.39822368615503	0
200	56.548704	$6.41 \cdot 10^{-7}$	75.39822368615503	0

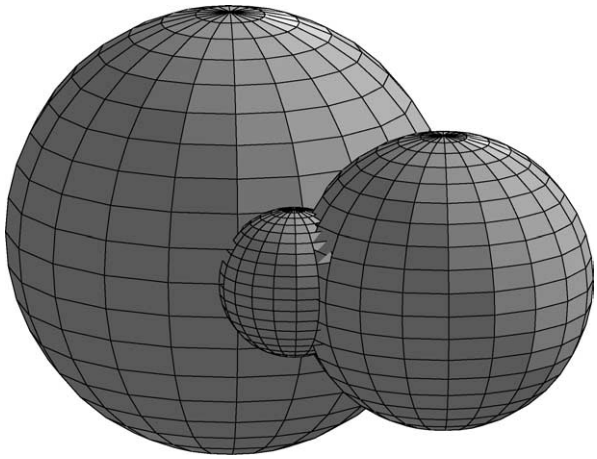

Fig. 7. Intersection of three spheres.

6.2.2. Three spheres intersection example

Another check for correctness with the (t, s) domain Ω_i having a boundary with circular arcs, can be done using the results of [17,18]. In these papers, the formulas for the volume and surface area of the *intersection of three spheres* with unequal radii are given.

Following [17], let A, B, and C be the centers of three spheres S_A, S_B, and S_C, respectively (see Fig. 7). We consider spheres of radii 1.0, 2.0, and 3.0 Å centered at A, B, and C, respectively, with distances AB, BC, and CA equal to 2.0, 4.0, and 3.0 Å, respectively. Then the volume and the surface area of the triple intersection calculated by formula, given in [17,18] are 0.5736 Å^3 and 4.214 Å^2, respectively.

The file `input.dat` containing three lines ($2.90473750965556 \doteq 3\sqrt{15}/4$)

```
0 0 0 1
2 0 0 2
-0.75 2.90473750965556 0 3
```

Due to the equalities

$$V(S_A \cap S_B \cap S_C) = V(S_A \cup S_B \cup S_C) - \big[V(S_A \cup S_B) + V(S_B \cup S_C) + V(S_A \cup S_C)\big] + \big[V(S_A) + V(S_B) + V(S_C)\big] \tag{30}$$

and

$$A(S_A \cap S_B \cap S_C) = A(S_A \cup S_B \cup S_C) - \big[A(S_A \cup S_B) + A(S_B \cup S_C) + A(S_A \cup S_C)\big] + \big[A(S_A) + A(S_B) + A(S_C)\big], \tag{31}$$

Table 3
Three spheres intersection volume and surface area results

Set	Volume [Å^3]	Area [Å^2]
$S_A \cap S_B \cap S_C$	0.5736544730318	4.2139413434876
$S_A \cup S_B \cup S_C$	144.3669682217146	148.9890027964171
$S_A \cup S_B$	35.9974158223830	54.9778714378214
$S_B \cup S_C$	143.1388152791849	148.4402528821177
$S_A \cup S_C$	115.4535300194249	117.2861257340189
	150.7964473723101	175.9291886010284

we can compute the triple intersection's volume $V(S_A \cap S_B \cap S_C)$ and surface area $A(S_A \cap S_B \cap S_C)$ using the program ARVO for calculation of the spheres' unions in the right sides of Eqs. (30) and (31), respectively. The values for the corresponding volumes and surface area are given in Table 3. In the last row the sums of volumes and surface areas of three separated spheres (S_A, S_B, and S_C) are given. The results for the three spheres' intersection volume and surface area were calculated using the formulas given in [17,18], respectively. One can easily check, that Eqs. (30) and (31) hold true. This is a successful test for the correctness of the ARVO program.

6.3. MPI parallel version of the main module

Due to the cycles over the (probably) large number of spheres in the ARVO main module, it is possible to solve the problem effectively on multiprocessor systems using Message Passing Interface (MPI) Fortran, creating a parallel program. Below in this subsection the MPI Fortran code of the main module of the program PARVO is presented. All subroutines and functions are the same as for the serial program.

This code may serve as a good example of processors communication. For instance, we said, that a random rotation of the molecule is necessary, if the North Pole test failed. But, because each processor make the rotation of only a part of the molecule, the random value should be generated only once by the 0th processor and than it should be transfered to other processors.

```
      program PARVO
c     MPI parallel computing the surface area and the volume
c     of the overlapping spheres
      implicit real*8(a-h,o-z)
      parameter (pi=3.14159265358979323846264d0,ks=300,kl=300
     1,ka=2000,ki=10000)
c          ks - maximal spheres' number
c          kl - maximal neighbors' number of one sphere (local spheres' number)
c          ka - maximal angles' or arcs' number
c          ki - maximal neighbors' relations' number = cca.
c               spheres' number * maximal neighbors' number

c
c      eps_north_pole - accuracy level in the function North_Pole_test
c      eps_deltat - accuracy level in the subroutine circles_intersection
c      eps_angle - accuracy level in the subroutine delete_equal (angles)
c
      dimension spheres(ks,4),neighbors_number(ks),index_start(ks),
     1                        neighbors_indices(ki),av(2)
c
c     spheres(i,1)=xi
```

```
c     spheres(i,2)=yi     - ith sphere center coordinates
c     spheres(i,3)=zi
c     spheres(i,4)=ri     - ith sphere radius
c

c     neighbors_number, index_start, neighbors_indices description
c     is given in the   subroutine make_neighbors

c
c     Definitions for MPI objects are in file mpif.h
c
      include 'mpif.h' ! definition of MPI-objects
c
c     We will use arrays
c
      integer status(MPI_STATUS_SIZE) ! matrix
c
c     Parameters for MPI:
c             comm - communicator
c             typ  - variables typ (for example MPI_INTEGER)
c             tag  - integer communications argument
c             ierr - error messages
c             myP  - name of "actual processor number"
c             Pr   - processors number
c
      integer comm,typDP,typI,tag,ierr,myP,Pr
      data tag/0/, typDP/MPI_DOUBLE_PRECISION/,comm/MPI_COMM_WORLD/,
     1     typI/MPI_INTEGER/

      data rwater/0d0/

c
c     Initialization of MPI
c
      call MPI_Init(ierr)
c
c     Defining the "actual" processor
c
      call MPI_Comm_rank(comm,myP,ierr) ! which processor
c
c     Defining processors number
c
      call MPI_Comm_size(comm,Pr,ierr) ! how many processors
c
c     All processors will be done next commands !!!!!
c
c
c     All processors open device
c
      open(unit=11,file='input.dat')
c
c     All processors read input data
```

```
c
c
c     input spheres data
c
      ns=0
 1     continue
         ns=ns+1
         read(11,*,end=2)spheres(ns,1),spheres(ns,2),spheres(ns,3),
     1                   spheres(ns,4)
         spheres(ns,4)=spheres(ns,4)+rwater
         goto 1
 2     continue
         close(11)
         ns=ns-1
c
c        ns - spheres number
c
c
c          Tasks' management
c

c        print *,'Processor number: ',Pr

      nc=ns/Pr        ! minimal tasks number
      ncmyP=nc*myP+1    ! order of 1st task number for myP (if all are lazy)
      nres=mod(ns,Pr) ! busy processors number (a task more)
      if(myP.Eq.0) then
c        0th processor will write output
         open(10,file='output.dat')
         if (nc.Eq.0) then
             nbPr=ns
           else
              nbPr=Pr ! busy processors number
         endif
      endif
c
c     all processors compute corresponding tasks' indices
c
      if(myP.lt.nres) then  ! myP is busy
         nc1=ncmyP+myP   ! 1st task index for myP processor
         nc2=nc1+nc      ! last task index for myP processor
      else                  ! myP is lazy
         nc1=nres+ncmyP  ! 1st task index for myP processor
         nc2=nc1+nc-1    ! last task index for myP processor
      endif

      if (nc1.le.nc2) then
c
c      All processors (in unknown order) write out task information
c

c      write(*,*) ' P: ',myP,' of',Pr,
```

```
c     1               ' started from ',nc1,' till ',nc2,
c     2               ' (nc,nres: ',nc,' ',nres,')  ENTER'

c
c     All processors solve corresponding tasks
c

c     Study the neighborhood relations
         call make_neighbors(nc1,nc2,spheres,neighbors_number,
      1           index_start,neighbors_indices,ks,kl,ns,ki)

 3       continue
c        print *, myP,': idem na NPT','  ns: ',ns
c        Here we check, that Nord Pole of no sphere lies on other sphere
c        neighbor sphere
c        npt=0 - BAD NEWS
            npt=North_Pole_test(nc1,nc2,spheres,neighbors_number,
      1                   index_start,neighbors_indices,ks,ki)
c        print *, myP,': ',npt,ns
c
c        All processors send nnp value to ZERO processor
c
      if (myP.Eq.0) then
      do i=1,nbPr-1  ! 0th processor obtains results of other ones
c
c      Receiving result from ith processor - put it in v_work variable
c      MPI_Recv(sa,1,typ,0,tag,comm,status,ierr)
c      print *, myP, ': cakam npt od ',i
c      print *, myP,': pred n_work',ns
       call MPI_Recv(n_work,1,typI,i,tag,comm,status,ierr)
c      print *, myP, ': dostal som ans od ',i,' ns: ',ns
            if (n_work.Eq.0) npt=0
               enddo
               do i=1,nbPr-1  ! 0th processor obtains results of other ones
                  call MPI_iSend(npt,1,typI,i,tag,comm,status,ierr)
               enddo
            else
c         print *, myP, ': poslem ans 0'
               call MPI_iSend(npt,1,typI,0,tag,comm,status,ierr)
                    !iSend = immediate Send
c         print *, myP, ': poslal som ans 0'
               call MPI_Recv(n_work,1,typI,0,tag,comm,status,ierr)
             npt=n_work
           endif

c      if (myP.Eq.0) continue
c
c      print *,'Minimal North Pole distance:   ',dmin
c      Change continue by print!!!!!!!!!!
c
       if (npt.Eq.0) then ! molecule rotation by random matrix
         if (myP.Eq.0) then
```

```
              print *,'Rotation after bad Nord Pole test!'
              write(10,*)'Rotation after bad Nord Pole test!'
c             iseed=3  ! iseed=3 ???
c             call srand(iseed)
c             sa=rand()/33536.0
c           CALL RANDOM_SEED()
c           CALL RANDOM_NUMBER(sa)
                    sa=0.324d0
                    do i=1,nbPr-1  ! 0th processor obtains results of other ones
                       call MPI_iSend(sa,1,typDP,i,tag,comm,status,ierr)
c         print *, myP, ': poslal som sa pre ',i, sa
                    enddo
                 else
c         print *, myP, ': som pred citanim sa'
                    call MPI_Recv(sa,1,typDP,0,tag,comm,status,ierr)
c         print *, myP, ': dostal som sa',sa
              endif
              call spheres_rotation(spheres,ks,ns,sa)
              goto 3
           endif

c
c     All processors solve corresponding tasks
c
          V=0d0
          A=0d0
          do i=nc1,nc2
             call areavolume(i,spheres,neighbors_number,index_start,
     1                    neighbors_indices,ks,kl,ka,ki,av)
             V=V+av(1)
             A=A+av(2)
          enddo
          print *,'Processor',myP,' results: V: ',V,' A: ',A
c
c     Results passing
c
          if(myP.Eq.0) then
             do i=1,nbPr-1  ! 0th processor obtains results of other ones
c
c            Receiving result from ith processor - put it in v_work variable
c
                   call MPI_Recv(v_work,1,typDP,i,tag,comm,status,ierr)
                   V=V+v_work
                   call MPI_Recv(a_work,1,typDP,i,tag,comm,status,ierr)
                   A=A+a_work
             enddo
             print *,'Volume: ',V,' Area: ',A,'  Spheres num: ',ns
c            Writing final result in file res.dat
             write(10,*)'Volume: ',V,' Area: ',A,'  Spheres num: ',ns
             close(10)
          else
c
```

Table 4
Numerical values for the second case with two spheres

n	Volume	ε_V	Area	ε_A
10	59.1547643341108	$4.61 \cdot 10^{-2}$	75.39822368615503	0
20	56.2887389303264	$4.60 \cdot 10^{-3}$	75.39822368615503	0
50	56.55012044715155	$2.57 \cdot 10^{-5}$	75.55907323001884	$2.13 \cdot 10^{-3}$
100	56.54507624244089	$6.35 \cdot 10^{-5}$	75.37811749317206	$2.67 \cdot 10^{-4}$
200	56.55075097274038	$3.68 \cdot 10^{-5}$	75.39068386378642	$1.00 \cdot 10^{-4}$

```
c      Sendig result V from myPth processor to ZERO processor
c
            call MPI_iSend(V,1,typDP,0,tag,comm,iqe,ierr)
                 ! iSend = immediate Send
            call MPI_iSend(A,1,typDP,0,tag,comm,iqe,ierr)
                 ! iSend = immediate Send
         endif
      else
        print *,myP,' pass'
      endif

      call MPI_Finalize(ierr)  ! Error messages handling
      stop 'End'
      end
```

Next we consider the same pair of spheres rotated by the angle $\pi/3$ around the x-axis (cf. Fig. 6 right). The file `input.dat` contains two lines

```
0 0 0 2
0 1.73205080756888 1 2
```

Here is the screen output for the MPI run with two processors, involved by command: `parvo -np 2` after the compilation made by the command `mpif77 -o parvo parvo.f`:

```
1.46410161513776
2.220446049250313E-15
Rotation after bad Nord Pole test!
5.323444279475265E-02
1.43281638351388
Processor 1 results: V:  30.38318449969805 A:  37.69911184307753
Processor 0 results: V:  26.16548326491951 A:  37.69911184307804
Volume: 56.54866776461756 Area: 75.39822368615557 Spheres num: 2
```

Each processor has checked one North Pole. Because the minimal distance value was too small, North Pole test failed and the molecule was rotated. The new test was successful. The values of the area and volume are changed on the last two or three positions. The corresponding numerical values are given in Table 4.

Remark 7. Shown results are not representative due to the special choice of the spheres. The analytical algorithms led in both cases to the single circle in the (t, s)-plane after the stereographic projection, so no intersections and no arc integrals were calculated. On the other hand very good results of numerical algorithm for the area computation were obtained by a lucky accident. In the second case we see that the results for $n = 50$, 100, and 200, which were expected to be better than those for $n = 10$ and 20, are in fact worse.

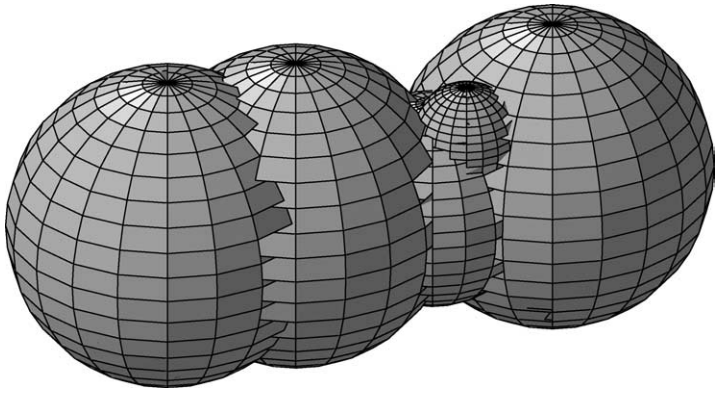

Fig. 8. Example with eight spheres.

Table 5
Numerical values for eight spheres

n	Volume	ε_V	Area	ε_A
10	2398.896	$2.96 \cdot 10^{-2}$	992.2406237098004	$1.94 \cdot 10^{-2}$
20	2342.52	$5.40 \cdot 10^{-3}$	1008.199914390036	$3.63 \cdot 10^{-3}$
50	2334.479616	$1.95 \cdot 10^{-3}$	1012.588091008571	$7.07 \cdot 10^{-4}$
100	2330.198928	$1.13 \cdot 10^{-4}$	1012.779099841909	$8.96 \cdot 10^{-4}$
200	2329.99335	$2.51 \cdot 10^{-5}$	1011.722268073241	$1.49 \cdot 10^{-4}$

6.3.1. An example with eight spheres

Consider the `input.dat` file with 8 spheres:

```
0 0 0 2
-5 0 0 6
5 0 0 6
0 1 0 4
-1 2 3 2
1 1 1 1
0 0 1 2
10 0 0 6
```

Both pictures in Fig. 8 were taken from the graphical output of the program test on this molecule in Matlab. In this case the program computes the arc integral. In Table 5 the relative errors are taken with respect to the results obtained by the program ARVO. The screen output after the command: `parvo -np 5` was

```
3.3851648071345
10000.0
.2679491924311228
1.81024967590665
.7082039324993694
Processor 3 results: V:.0 A:.0
Processor 4 results: V:  718.7702192025645 A:  320.4424506661588
Processor 1 results: V:  722.0196615613015 A:  288.5591425214843
Processor 0 results: V:  852.5202331905718 A:  389.7566522240995
Processor 2 results: V:  36.62471588135763 A:  13.1142859640694
Volume: 2329.934829835795 Area: 1011.872531375812 Spheres num: 8
```

Table 6
Numerical values for a peptide

n	Volume	ε_V	Area	ε_A
10	3375.63526	$1.18 \cdot 10^{-2}$	1530.35288	$4.16 \cdot 10^{-3}$
20	3342.21313	$1.83 \cdot 10^{-3}$	1518.36578	$3.70 \cdot 10^{-3}$
50	3333.56793	$7.62 \cdot 10^{-4}$	1523.95009	$3.80 \cdot 10^{-5}$
100	3335.36159	$2.24 \cdot 10^{-4}$	1524.09977	$6.02 \cdot 10^{-5}$
200	3336.27234	$4.86 \cdot 10^{-5}$	1524.41268	$2.65 \cdot 10^{-4}$

Remark 8. The above 5 values are from the function `North_Pole_test`, each one from some processor (we did not transfer the processor number into this function, so we do not know, which result is from which processor). Because the 3rd processor returned zero values, it calculated the values for some sphere (probably the 6th), which is the subset of some other sphere (4th sphere's). Because 10000 is the initial value for the minimal value `dmin` in the function `North_Pole_test`, we can conclude, that the result in the second line is the output made by the 3rd processor.

6.3.2. The example of the 14-residue peptide rgkwtyngityegr

In this subsection we apply the program ARVO to the calculation of the volume and surface area of the 14-residue long peptide with PDB code *1j4m*. The screen output after using the command `arvo` is as follows:

```
 0.00130930672
Volume:   3336.11018   Area:   1524.00799  Spheres num:  125
```

The corresponding numerical values are shown in Table 6.

6.3.3. Efficiency of parallelization

We further test the efficiency of parallelization of the program, parvo.f. We use the concepts of time speed-up and efficiency to measure the degree of parallelization of a computational process. The time speed-up is defined as

$$s(n) = \frac{t(1)}{t(n)}, \tag{32}$$

and the percent of parallelization (efficiency) is

$$e(n) = \frac{s(n)}{n}, \tag{33}$$

where $t(1)$ and $t(n)$ are computational times on 1 and n computers, respectively. The parallelization is considered acceptable when $e(n) \geqslant 0.5$. Fig. 9 show the dependence of the speedup and the efficiency of parallelization for the calculation of accessible surface area and volume of three real proteins with PDB codes 1e7i, 1h76 and 1cd3 which consist of 4496, 5254 and 9755 atoms, respectively. These calculations are performed in a PC-Cluster with 24 Pentium 4 Xeon 2.4 GHz Processors at the Laboratory of Statistical and Computational Physics of academia Sinica in Taipei [37]. The quality of the parallelization depends on how evenly the workloads are distributed among different computers and how much time is wasted on data transfer between them. We find that the larger is the number of atoms the better the time speed-up and efficiency.

7. Conclusion and discussion

The testings and comparison to other analytical algorithms show that the suggested method is robust and efficient.

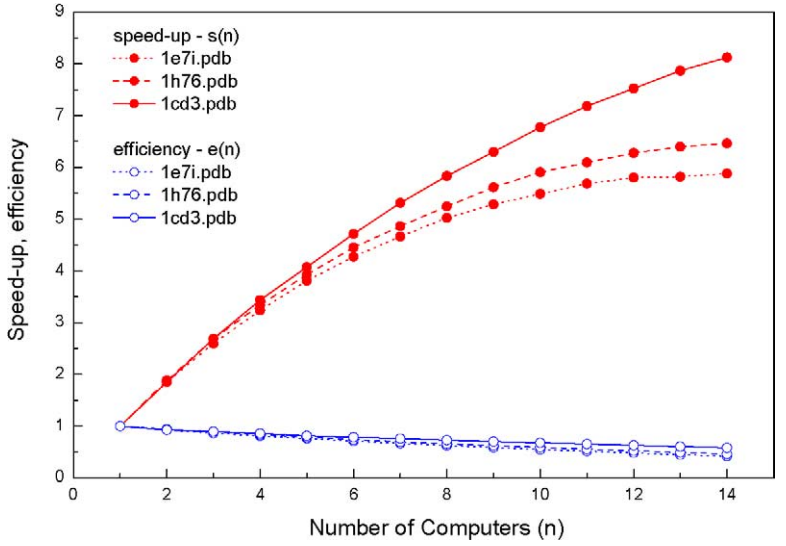

Fig. 9. The dependence of the speed-up and the efficiency of parallelization for the calculation of accessible surface area and volume of three proteins, 1e7i.pdb (4496 atoms), 1h76.pdb (5254 atoms), and 1cd3.pdb (9755 atoms).

The tables show the quality of the numerical methods. For some purposes these relative errors are maybe not large, in other cases, such as size measurement of percolation clusters one needs a better approximation of the volume. In this case it is better to use proposed analytical method. This method can be used for the comparison of the quality of different numerical algorithms, too.

The computation time of simultaneous calculation of surface area and volume is practically the same as the time for volume computation only. The code can be efficiently parallelized.

It is easy to include the program presented in this paper into various algorithms or computers packages [36,38,39] for simulations of proteins.

It has been found that many two-dimensional percolation models (including site and bond percolation on regular lattices [40–42], continuum percolation of soft disks and hard disks [43], bond percolation on random lattices [44] and continuum percolation of disks with different radius [45]) have universal finite-size scaling functions for their existence probabilities [46,47] (or call crossing probabilities). It has also been found that site and bond percolation models on three-dimensional lattices have universal finite-size scaling functions for their existence probabilities [48]. Our computing package **ARVO** can be used to compute the volume of overlapping spheres of percolating clusters for continuum percolation, which is proportional to the percolation probability of the percolation model. Thus we can use **ARVO** to check whether the percolation probability of three-dimensional lattice and continuum percolation models have universal finite-size scaling functions and critical exponent.

Acknowledgements

This work was partially supported by the National Science Council of the Republic of China (Taiwan) under Grant No. NSC 92-2112-M-001-063. The computations on SPP'2000 were carried out at the Laboratory of Information Technologies, Joint Institute for Nuclear Research (Russia). The investigations at LIT JINR, Dubna were supported in part by the Russian Found for Fundamental Research under Grant No. 02-01-00606.

Appendix A. Computation of the solvent accessible surface area of the peptide *1j4m* by GETAREA and ACCAR

The output from GETAREA

```
Surface Area and Solvation Energy of Macromolecules Output of
%\interref[locator-type=url,object-type=text/html]
GETAREA 1.1 http://www.scsb.utmb.edu/getarea/area_form.html
Tue Jan 27 03:31:46 CST 2004

 Job identifier:  get_a_25284
 Probe radius  :  1.400

 ATOM  NAME  RESIDUE       AREA            GRADIENT
     1 N     ARG     1     26.97         1.71     4.39    -9.18
     2 CA    ARG     1      5.33        -2.32    -4.39     1.37
     3 C     ARG     1      2.37        -0.15    -0.02     0.31
     4 O     ARG     1     21.41         2.14    -1.87    -5.20
     5 CB    ARG     1     30.97        -3.43     9.70     1.29
     6 CG    ARG     1      7.04        -0.43     3.43    -3.16
     7 CD    ARG     1     40.83       -10.15     6.31     2.31
     8 NE    ARG     1      0.00         0.00     0.00     0.00
     9 CZ    ARG     1     16.76        -0.40    -2.39    -0.18
    10 NH1   ARG     1     28.36        -6.02   -10.96     1.00
    11 NH2   ARG     1     31.92        -6.04     0.29   -12.09
    27 N     GLY     2      0.00         0.00     0.00     0.00
    28 CA    GLY     2      5.92         2.27    -2.57    -1.98
    29 C     GLY     2      0.23        -0.14     0.46     0.81
    30 O     GLY     2      2.98        -0.71     1.31     1.14
    34 N     LYS     3      1.28         3.04    -1.09     2.09
    35 CA    LYS     3      0.00         0.00     0.00     0.00
    36 C     LYS     3      0.00         0.00     0.00     0.00
    37 O     LYS     3     26.13        -4.63     3.16    -4.21
    38 CB    LYS     3     19.27        -1.65    -1.14    -1.97
    39 CG    LYS     3      0.00         0.00     0.00     0.00
    40 CD    LYS     3     13.61        -1.11    -2.69     0.44
    41 CE    LYS     3     27.70        -1.69    -2.69    -2.96
    42 NZ    LYS     3     41.51         4.65    -2.90   -15.30
    56 N     TRP     4      0.00         0.00     0.00     0.00
    57 CA    TRP     4      5.60         0.43    -0.38     1.23
    58 C     TRP     4      0.00         0.00     0.00     0.00
    59 O     TRP     4      0.00         0.01    -0.05    -0.04
    60 CB    TRP     4     28.95        -7.56     0.52    -1.05
    61 CG    TRP     4      1.34         0.26    -0.14    -0.13
    62 CD1   TRP     4     16.29        -2.71     1.38     1.60
    63 CD2   TRP     4      2.39         0.04    -0.06     0.01
    64 NE1   TRP     4      3.78         0.37     0.70     0.06
    65 CE2   TRP     4      4.58        -0.45     0.18     0.17
    66 CE3   TRP     4     13.41        -2.51    -3.00     2.44
    67 CZ2   TRP     4     13.70        -4.04     2.07     0.68
    68 CZ3   TRP     4     17.91        -2.81     2.89     2.67
```

```
 69 CH2  TRP     4     33.45        -2.25    -1.40    11.10
 80 N    THR     5      1.84         2.26    -1.38     0.61
 81 CA   THR     5      0.00         0.00     0.00     0.00
 82 C    THR     5      0.00         0.00     0.00     0.00
 83 O    THR     5     25.60        -5.46     4.17    -5.17
 84 CB   THR     5     16.12        -0.80    -0.55    -2.81
 85 OG1  THR     5     20.35        -1.04     4.22    -4.82
 86 CG2  THR     5     18.98        -0.24     0.18    -2.36
 94 N    TYR     6      0.00         0.00     0.00     0.00
 95 CA   TYR     6      9.52        -0.70     0.22     1.42
 96 C    TYR     6      0.00         0.00     0.00     0.00
 97 O    TYR     6      0.00         0.00     0.00     0.00
 98 CB   TYR     6     21.57         0.67     7.83    -3.24
 99 CG   TYR     6      0.94         0.39     0.04    -0.13
100 CD1  TYR     6     16.17        -3.79     1.54    -0.43
101 CD2  TYR     6      6.63        -0.23     1.78    -0.59
102 CE1  TYR     6     24.29        -5.95     5.28     0.34
103 CE2  TYR     6     19.37        -4.63     1.52     2.29
104 CZ   TYR     6     10.84        -3.51    -1.17     1.19
105 OH   TYR     6     39.17        -0.36     6.59    13.98
115 N    ASN     7      5.74         0.14     0.12    -0.19
116 CA   ASN     7     19.23        -1.48     2.43    -3.81
117 C    ASN     7      0.57        -0.14    -0.35     0.56
118 O    ASN     7     19.74         3.09     2.32    -2.99
119 CB   ASN     7      2.37         0.13    -0.91    -2.95
120 CG   ASN     7      0.62        -0.07    -0.21    -0.01
121 OD1  ASN     7     33.48         0.31    11.91    -4.42
122 ND2  ASN     7     44.82         8.18    10.75     6.73
129 N    GLY     8      2.64         0.84    -0.45     0.95
130 CA   GLY     8     45.62         5.74     5.49   -11.14
131 C    GLY     8      0.02        -0.28    -0.09    -0.11
132 O    GLY     8     24.38         6.86    -0.88    -2.07
136 N    ILE     9      0.00         0.00     0.00     0.00
137 CA   ILE     9      4.18        -1.42     0.59     0.14
138 C    ILE     9      0.00         0.00     0.00     0.00
139 O    ILE     9      0.00         0.00     0.00     0.00
140 CB   ILE     9      0.04         0.25    -0.21    -0.45
141 CG1  ILE     9      4.37         1.78    -1.85    -2.84
142 CG2  ILE     9     15.95         1.25    -1.41    -1.68
143 CD1  ILE     9     65.61        16.50    -0.24     9.17
155 N    THR    10      1.97        -1.64     0.49    -0.16
156 CA   THR    10      0.00         0.00     0.00     0.00
157 C    THR    10      0.00         0.00     0.00     0.00
158 O    THR    10     17.18         2.42    -2.81     0.59
159 CB   THR    10     20.42         5.21    -2.26    -0.84
160 OG1  THR    10     13.98         1.55    -1.48    -2.70
161 CG2  THR    10     23.76         3.33     0.71    -2.01
169 N    TYR    11      0.00         0.00     0.00     0.00
170 CA   TYR    11      6.06        -1.50     1.97    -0.79
171 C    TYR    11      0.00         0.00     0.00     0.00
172 O    TYR    11      0.00         0.00     0.00     0.00
173 CB   TYR    11      0.32        -0.03     0.78    -0.64
```

```
   174 CG   TYR    11      0.00         0.00     0.00     0.00
   175 CD1  TYR    11      0.73         2.44    -2.51    -0.27
   176 CD2  TYR    11     11.46         0.85    -0.38     0.01
   177 CE1  TYR    11      7.35         2.19    -5.58    -2.20
   178 CE2  TYR    11     27.14         3.60    -4.89     5.94
   179 CZ   TYR    11      7.57        -1.18     0.35     1.22
   180 OH   TYR    11     33.48         3.03     3.63    11.62
   190 N    GLU    12      3.97        -1.65     0.22    -0.71
   191 CA   GLU    12      0.06        -0.34     0.03    -0.54
   192 C    GLU    12      0.34        -0.45    -0.31    -0.59
   193 O    GLU    12     16.44         3.33    -0.65     2.66
   194 CB   GLU    12     13.14         0.61     2.55    -0.04
   195 CG   GLU    12      2.74        -2.31     0.40     1.87
   196 CD   GLU    12      0.36        -0.06     0.27     0.02
   197 OE1  GLU    12     39.52        13.46    -6.46     1.95
   198 OE2  GLU    12     27.46        -0.56    -8.92    -2.55
   205 N    GLY    13      0.00         0.00     0.00     0.00
   206 CA   GLY    13     25.18        -1.26    -1.31     3.47
   207 C    GLY    13      0.29         0.36     0.02    -0.39
   208 O    GLY    13     25.10         3.24    -3.21     7.19
   212 N    ARG    14      1.12         1.86    -1.43    -0.71
   213 CA   ARG    14      2.16        -1.15     0.64    -1.19
   214 C    ARG    14      0.62        -2.71     0.41     0.02
   215 O    ARG    14     35.01         8.83    -9.98    -3.29
   216 CB   ARG    14      7.82         1.63    -1.27     0.58
   217 CG   ARG    14     24.04         1.59    -3.54     2.98
   218 CD   ARG    14     26.66         0.21    -6.36     0.33
   219 NE   ARG    14      0.00         0.00     0.00     0.00
   220 CZ   ARG    14     25.40        -1.89     0.53     2.57
   221 NH1  ARG    14     23.71        -6.98    -4.07     1.51
   222 NH2  ARG    14     34.73        -5.42    -3.88    13.69
   223 OXT  ARG    14      3.98        -2.66     1.97     2.96

 -------------------------------------------------------
 POLAR  area (~Solv. Energy)           =          679.74
 APOLAR area (~Solv. Energy)           =          844.27
 UNKNOW area (~Solv. Energy)           =            0.00
 -------------------------------------------------------
 Total  area (~Solv. Energy)           =         1524.01
 -------------------------------------------------------
 Number of surface atoms    =           98
 Number of buried atoms     =           23
 Number of atoms with ASP=0 =          115
=====================================================================

          The output from  ACCAR

    No  Res   Area    gradx   grady   gradz   rad   Atom
    1  ARG   26.97    1.71    4.39   -9.18    1.50  N
    2  ARG    5.33   -2.32   -4.39    1.37    2.00  CA
    3  ARG    2.37   -0.15   -0.02    0.31    1.50  C
    4  ARG   21.41    2.14   -1.87   -5.20    1.40  O
```

```
 5  ARG   30.97   -3.43    9.70    1.29    2.00  CB
 6  ARG    7.04   -0.43    3.43   -3.16    2.00  CG
 7  ARG   40.83  -10.15    6.31    2.31    2.00  CD
 8  ARG    0.00    0.00    0.00    0.00    1.50  NE
 9  ARG   16.76   -0.40   -2.39   -0.18    1.85  CZ
10  ARG   28.36   -6.02  -10.96    1.00    1.50  NH1
11  ARG   31.92   -6.04    0.29  -12.09    1.50  NH2
12  ARG    0.00    0.00    0.00    0.00    0.00  H
13  ARG    0.00    0.00    0.00    0.00    0.00  H
14  ARG    0.00    0.00    0.00    0.00    0.00  H
15  ARG    0.00    0.00    0.00    0.00    0.00  HA
16  ARG    0.00    0.00    0.00    0.00    0.00  HB
17  ARG    0.00    0.00    0.00    0.00    0.00  HB
18  ARG    0.00    0.00    0.00    0.00    0.00  HG
19  ARG    0.00    0.00    0.00    0.00    0.00  HG
20  ARG    0.00    0.00    0.00    0.00    0.00  HD
21  ARG    0.00    0.00    0.00    0.00    0.00  HD
22  ARG    0.00    0.00    0.00    0.00    0.00  HE
23  ARG    0.00    0.00    0.00    0.00    0.00  HH1
24  ARG    0.00    0.00    0.00    0.00    0.00  HH1
25  ARG    0.00    0.00    0.00    0.00    0.00  HH2
26  ARG    0.00    0.00    0.00    0.00    0.00  HH2
27  GLY    0.00    0.00    0.00    0.00    1.50  N
28  GLY    5.92    2.27   -2.57   -1.98    2.00  CA
29  GLY    0.23   -0.14    0.46    0.81    1.50  C
30  GLY    2.98   -0.71    1.31    1.14    1.40  O
31  GLY    0.00    0.00    0.00    0.00    0.00  H
32  GLY    0.00    0.00    0.00    0.00    0.00  HA
33  GLY    0.00    0.00    0.00    0.00    0.00  HA
34  LYS    1.28    3.04   -1.09    2.09    1.50  N
35  LYS    0.00    0.00    0.00    0.00    2.00  CA
36  LYS    0.00    0.00    0.00    0.00    1.50  C
37  LYS   26.13   -4.63    3.16   -4.21    1.40  O
38  LYS   19.27   -1.65   -1.14   -1.97    2.00  CB
39  LYS    0.00    0.00    0.00    0.00    2.00  CG
40  LYS   13.61   -1.11   -2.69    0.44    2.00  CD
41  LYS   27.70   -1.69   -2.69   -2.96    2.00  CE
42  LYS   41.51    4.65   -2.90  -15.30    1.50  NZ
43  LYS    0.00    0.00    0.00    0.00    0.00  H
44  LYS    0.00    0.00    0.00    0.00    0.00  HA
45  LYS    0.00    0.00    0.00    0.00    0.00  HB
46  LYS    0.00    0.00    0.00    0.00    0.00  HB
47  LYS    0.00    0.00    0.00    0.00    0.00  HG
48  LYS    0.00    0.00    0.00    0.00    0.00  HG
49  LYS    0.00    0.00    0.00    0.00    0.00  HD
50  LYS    0.00    0.00    0.00    0.00    0.00  HD
51  LYS    0.00    0.00    0.00    0.00    0.00  HE
52  LYS    0.00    0.00    0.00    0.00    0.00  HE
53  LYS    0.00    0.00    0.00    0.00    0.00  HZ
54  LYS    0.00    0.00    0.00    0.00    0.00  HZ
55  LYS    0.00    0.00    0.00    0.00    0.00  HZ
56  TRP    0.00    0.00    0.00    0.00    1.50  N
```

```
  57  TRP    5.60    0.43   -0.38    1.23    2.00  CA
  58  TRP    0.00    0.00    0.00    0.00    1.50  C
  59  TRP    0.00    0.01   -0.05   -0.04    1.40  O
  60  TRP   28.95   -7.56    0.52   -1.05    2.00  CB
  61  TRP    1.34    0.26   -0.14   -0.13    1.85  CG
  62  TRP   16.29   -2.71    1.38    1.60    1.85  CD1
  63  TRP    2.39    0.04   -0.06    0.01    1.85  CD2
  64  TRP    3.78    0.37    0.70    0.06    1.50  NE1
  65  TRP    4.58   -0.45    0.18    0.17    1.85  CE2
  66  TRP   13.41   -2.51   -3.00    2.44    1.85  CE3
  67  TRP   13.70   -4.04    2.07    0.68    1.85  CZ2
  68  TRP   17.91   -2.81    2.89    2.67    1.85  CZ3
  69  TRP   33.45   -2.25   -1.40   11.10    1.85  CH2
  70  TRP    0.00    0.00    0.00    0.00    0.00  H
  71  TRP    0.00    0.00    0.00    0.00    0.00  HA
  72  TRP    0.00    0.00    0.00    0.00    0.00  HB
  73  TRP    0.00    0.00    0.00    0.00    0.00  HB
  74  TRP    0.00    0.00    0.00    0.00    0.00  HD1
  75  TRP    0.00    0.00    0.00    0.00    0.00  HE1
  76  TRP    0.00    0.00    0.00    0.00    0.00  HE3
  77  TRP    0.00    0.00    0.00    0.00    0.00  HZ2
  78  TRP    0.00    0.00    0.00    0.00    0.00  HZ3
  79  TRP    0.00    0.00    0.00    0.00    0.00  HH2
  80  THR    1.84    2.26   -1.38    0.61    1.50  N
  81  THR    0.00    0.00    0.00    0.00    2.00  CA
  82  THR    0.00    0.00    0.00    0.00    1.50  C
  83  THR   25.60   -5.46    4.17   -5.17    1.40  O
  84  THR   16.12   -0.80   -0.55   -2.81    2.00  CB
  85  THR   20.35   -1.04    4.22   -4.82    1.40  OG1
  86  THR   18.98   -0.24    0.18   -2.36    2.00  CG2
  87  THR    0.00    0.00    0.00    0.00    0.00  H
  88  THR    0.00    0.00    0.00    0.00    0.00  HA
  89  THR    0.00    0.00    0.00    0.00    0.00  HB
  90  THR    0.00    0.00    0.00    0.00    0.00  HG1
  91  THR    0.00    0.00    0.00    0.00    0.00  HG2
  92  THR    0.00    0.00    0.00    0.00    0.00  HG2
  93  THR    0.00    0.00    0.00    0.00    0.00  HG2
  94  TYR    0.00    0.00    0.00    0.00    1.50  N
  95  TYR    9.52   -0.70    0.22    1.42    2.00  CA
  96  TYR    0.00    0.00    0.00    0.00    1.50  C
  97  TYR    0.00    0.00    0.00    0.00    1.40  O
  98  TYR   21.57    0.67    7.83   -3.24    2.00  CB
  99  TYR    0.94    0.39    0.04   -0.13    1.85  CG
 100  TYR   16.17   -3.79    1.54   -0.43    1.85  CD1
 101  TYR    6.63   -0.23    1.78   -0.59    1.85  CD2
 102  TYR   24.29   -5.95    5.28    0.34    1.85  CE1
 103  TYR   19.37   -4.63    1.52    2.29    1.85  CE2
 104  TYR   10.84   -3.51   -1.17    1.19    1.85  CZ
 105  TYR   39.17   -0.36    6.59   13.98    1.40  OH
 106  TYR    0.00    0.00    0.00    0.00    0.00  H
 107  TYR    0.00    0.00    0.00    0.00    0.00  HA
 108  TYR    0.00    0.00    0.00    0.00    0.00  HB
```

```
109  TYR    0.00    0.00    0.00    0.00    0.00  HB
110  TYR    0.00    0.00    0.00    0.00    0.00  HD1
111  TYR    0.00    0.00    0.00    0.00    0.00  HD2
112  TYR    0.00    0.00    0.00    0.00    0.00  HE1
113  TYR    0.00    0.00    0.00    0.00    0.00  HE2
114  TYR    0.00    0.00    0.00    0.00    0.00  HH
115  ASN    5.74    0.14    0.12   -0.19    1.50  N
116  ASN   19.23   -1.48    2.43   -3.81    2.00  CA
117  ASN    0.57   -0.14   -0.35    0.56    1.50  C
118  ASN   19.74    3.09    2.32   -2.99    1.40  O
119  ASN    2.37    0.13   -0.91   -2.95    2.00  CB
120  ASN    0.62   -0.07   -0.21   -0.01    1.50  CG
121  ASN   33.48    0.31   11.91   -4.42    1.40  OD1
122  ASN   44.82    8.18   10.75    6.73    1.50  ND2
123  ASN    0.00    0.00    0.00    0.00    0.00  H
124  ASN    0.00    0.00    0.00    0.00    0.00  HA
125  ASN    0.00    0.00    0.00    0.00    0.00  HB
126  ASN    0.00    0.00    0.00    0.00    0.00  HB
127  ASN    0.00    0.00    0.00    0.00    0.00  HD2
128  ASN    0.00    0.00    0.00    0.00    0.00  HD2
129  GLY    2.64    0.84   -0.45    0.95    1.50  N
130  GLY   45.62    5.74    5.49  -11.14    2.00  CA
131  GLY    0.02   -0.28   -0.09   -0.11    1.50  C
132  GLY   24.38    6.86   -0.88   -2.07    1.40  O
133  GLY    0.00    0.00    0.00    0.00    0.00  H
134  GLY    0.00    0.00    0.00    0.00    0.00  HA
135  GLY    0.00    0.00    0.00    0.00    0.00  HA
136  ILE    0.00    0.00    0.00    0.00    1.50  N
137  ILE    4.18   -1.42    0.59    0.14    2.00  CA
138  ILE    0.00    0.00    0.00    0.00    1.50  C
139  ILE    0.00    0.00    0.00    0.00    1.40  O
140  ILE    0.04    0.25   -0.21   -0.45    2.00  CB
141  ILE    4.37    1.78   -1.85   -2.84    2.00  CG1
142  ILE   15.95    1.25   -1.41   -1.68    2.00  CG2
143  ILE   65.61   16.50   -0.24    9.17    2.00  CD1
144  ILE    0.00    0.00    0.00    0.00    0.00  H
145  ILE    0.00    0.00    0.00    0.00    0.00  HA
146  ILE    0.00    0.00    0.00    0.00    0.00  HB
147  ILE    0.00    0.00    0.00    0.00    0.00  HG1
148  ILE    0.00    0.00    0.00    0.00    0.00  HG1
149  ILE    0.00    0.00    0.00    0.00    0.00  HG2
150  ILE    0.00    0.00    0.00    0.00    0.00  HG2
151  ILE    0.00    0.00    0.00    0.00    0.00  HG2
152  ILE    0.00    0.00    0.00    0.00    0.00  HD1
153  ILE    0.00    0.00    0.00    0.00    0.00  HD1
154  ILE    0.00    0.00    0.00    0.00    0.00  HD1
155  THR    1.97   -1.64    0.49   -0.16    1.50  N
156  THR    0.00    0.00    0.00    0.00    2.00  CA
157  THR    0.00    0.00    0.00    0.00    1.50  C
158  THR   17.18    2.42   -2.81    0.59    1.40  O
159  THR   20.42    5.21   -2.26   -0.84    2.00  CB
160  THR   13.98    1.55   -1.48   -2.70    1.40  OG1
```

```
161  THR   23.76    3.33    0.71   -2.01    2.00  CG2
162  THR    0.00    0.00    0.00    0.00    0.00  H
163  THR    0.00    0.00    0.00    0.00    0.00  HA
164  THR    0.00    0.00    0.00    0.00    0.00  HB
165  THR    0.00    0.00    0.00    0.00    0.00  HG1
166  THR    0.00    0.00    0.00    0.00    0.00  HG2
167  THR    0.00    0.00    0.00    0.00    0.00  HG2
168  THR    0.00    0.00    0.00    0.00    0.00  HG2
169  TYR    0.00    0.00    0.00    0.00    1.50  N
170  TYR    6.06   -1.50    1.97   -0.79    2.00  CA
171  TYR    0.00    0.00    0.00    0.00    1.50  C
172  TYR    0.00    0.00    0.00    0.00    1.40  O
173  TYR    0.32   -0.03    0.78   -0.64    2.00  CB
174  TYR    0.00    0.00    0.00    0.00    1.85  CG
175  TYR    0.73    2.44   -2.51   -0.27    1.85  CD1
176  TYR   11.46    0.85   -0.38    0.01    1.85  CD2
177  TYR    7.35    2.19   -5.58   -2.20    1.85  CE1
178  TYR   27.14    3.60   -4.89    5.94    1.85  CE2
179  TYR    7.57   -1.18    0.35    1.22    1.85  CZ
180  TYR   33.48    3.03    3.63   11.62    1.40  OH
181  TYR    0.00    0.00    0.00    0.00    0.00  H
182  TYR    0.00    0.00    0.00    0.00    0.00  HA
183  TYR    0.00    0.00    0.00    0.00    0.00  HB
184  TYR    0.00    0.00    0.00    0.00    0.00  HB
185  TYR    0.00    0.00    0.00    0.00    0.00  HD1
186  TYR    0.00    0.00    0.00    0.00    0.00  HD2
187  TYR    0.00    0.00    0.00    0.00    0.00  HE1
188  TYR    0.00    0.00    0.00    0.00    0.00  HE2
189  TYR    0.00    0.00    0.00    0.00    0.00  HH
190  GLU    3.97   -1.65    0.22   -0.71    1.50  N
191  GLU    0.06   -0.34    0.03   -0.54    2.00  CA
192  GLU    0.34   -0.45   -0.31   -0.59    1.50  C
193  GLU   16.44    3.33   -0.65    2.66    1.40  O
194  GLU   13.14    0.61    2.55   -0.04    2.00  CB
195  GLU    2.74   -2.31    0.40    1.87    2.00  CG
196  GLU    0.36   -0.06    0.27    0.02    1.50  CD
197  GLU   39.52   13.46   -6.46    1.95    1.40  OE1
198  GLU   27.46   -0.56   -8.92   -2.55    1.40  OE2
199  GLU    0.00    0.00    0.00    0.00    0.00  H
200  GLU    0.00    0.00    0.00    0.00    0.00  HA
201  GLU    0.00    0.00    0.00    0.00    0.00  HB
202  GLU    0.00    0.00    0.00    0.00    0.00  HB
203  GLU    0.00    0.00    0.00    0.00    0.00  HG
204  GLU    0.00    0.00    0.00    0.00    0.00  HG
205  GLY    0.00    0.00    0.00    0.00    1.50  N
206  GLY   25.18   -1.26   -1.31    3.47    2.00  CA
207  GLY    0.29    0.36    0.02   -0.39    1.50  C
208  GLY   25.10    3.24   -3.21    7.19    1.40  O
209  GLY    0.00    0.00    0.00    0.00    0.00  H
210  GLY    0.00    0.00    0.00    0.00    0.00  HA
211  GLY    0.00    0.00    0.00    0.00    0.00  HA
212  ARG    1.12    1.86   -1.43   -0.71    1.50  N
```

```
 213  ARG    2.16   -1.15    0.64   -1.19    2.00  CA
 214  ARG    0.62   -2.71    0.41    0.02    1.50  C
 215  ARG   35.01    8.83   -9.98   -3.29    1.40  O
 216  ARG    7.82    1.63   -1.27    0.58    2.00  CB
 217  ARG   24.04    1.59   -3.54    2.98    2.00  CG
 218  ARG   26.66    0.21   -6.36    0.33    2.00  CD
 219  ARG    0.00    0.00    0.00    0.00    1.50  NE
 220  ARG   25.40   -1.89    0.53    2.57    1.85  CZ
 221  ARG   23.71   -6.98   -4.07    1.51    1.50  NH1
 222  ARG   34.73   -5.42   -3.88   13.69    1.50  NH2
 223  ARG    3.98   -2.66    1.97    2.96    1.40  OXT
 224  ARG    0.00    0.00    0.00    0.00    0.00  H
 225  ARG    0.00    0.00    0.00    0.00    0.00  HA
 226  ARG    0.00    0.00    0.00    0.00    0.00  HB
 227  ARG    0.00    0.00    0.00    0.00    0.00  HB
 228  ARG    0.00    0.00    0.00    0.00    0.00  HG
 229  ARG    0.00    0.00    0.00    0.00    0.00  HG
 230  ARG    0.00    0.00    0.00    0.00    0.00  HD
 231  ARG    0.00    0.00    0.00    0.00    0.00  HD
 232  ARG    0.00    0.00    0.00    0.00    0.00  HE
 233  ARG    0.00    0.00    0.00    0.00    0.00  HH1
 234  ARG    0.00    0.00    0.00    0.00    0.00  HH1
 235  ARG    0.00    0.00    0.00    0.00    0.00  HH2
 236  ARG    0.00    0.00    0.00    0.00    0.00  HH2

  Total Area:   1524.00799
```

References

[1] D. Eisenberg, A.D. McLachlan, Nature 316 (1986) 199.
[2] B. Lee, F.M. Richards, J. Mol. Biol. 55 (1971) 379–400.
[3] C. Chothia, Nature 248 (1974) 338–339.
[4] T.J. Richmond, J. Mol. Biol. 178 (1984) 63–89.
[5] T.L. Hill, in: Statistical Mechanics, McGraw-Hill, New York, pp. 122–285.
[6] A. Shrake, J.A. Rupley, J. Mol. Biol. 79 (1973) 351.
[7] T.J. Richmond, F.M. Richards, J. Mol. Biol. 129 (1978) 527.
[8] E. Sila, I. Tunom, J.L. Pascual-Ahuir, J. Comp. Chem. 12 (1991) 1077.
[9] W.C. Still, A. Tempczyk, R.C. Hawley, T. Hendricson, J. Am. Chem. Soc. 112 (1990) 6127.
[10] F. Eisenhaber, Ph. Lijnzaad, P. Argos, C. Sander, M. Scharf, J. Comp. Chem. 16 (1995) 273.
[11] M. Masuya, J. Doi, J. Molec. Graphics 13 (1995) 331.
[12] J.S. Rowlinson, Molec. Phys. 6 (1963) 517–524.
[13] R. Pavani, G. Ranghino, Comput. and Chem. 6 (1982) 133–135.
[14] A. Gavezzotti, J. Am. Chem. Soc. 105 (1983) 5220–5225.
[15] Y.K. Kang, G. Nemethy, H.A. Scheraga, J. Phys. Chem. 91 (1987) 4105–4109.
[16] K.D. Gibson, H.A. Scheraga, Molec. Phys. 62 (1987) 1247–1265.
[17] K.D. Gibson, H.A. Scheraga, J. Phys. Chem. 91 (1987) 4121–4122.
[18] K.D. Gibson, H.A. Scheraga, Molec. Phys. 64 (1988) 641–644.
[19] G. Guerrero-Ruiz, A. Ocadiz-Ramirez, R. Garduno-Juarez, Comput. and Chem. 15 (1991) 351–352.
[20] M. Petitjean, J. Comp. Chem. 15 (1994) 507–523.
[21] R. Fraczkiewicz, W. Braun, J. Comp. Chem. 19 (1998) 319.
[22] J. Greer, B. Bush, Proc. Natl. Acad. Sci. USA 75 (1978) 303–307.
[23] M.L. Connolly, Molecular Surfaces: A Review, which can be found at http://www.netsci.org/Science/Compchem/feature14g.html.

[24] M.L. Connolly, J. Appl. Cryst. 16 (1976) 548–558.
[25] M.L. Connolly, J. Am. Chem. Soc. 107 (1985) 1118–1124.
[26] E. Silla, I. Tuon, J.L. Pascual-Ahuir, J. Comp. Chem. 12 (1991) 1077–1088.
[27] E. Silla, F. Villar, O. Nilsson, J.L. Pascual-Ahuir, O. Tapia, J. Mol. Graphics 8 (1990) 168–172.
[28] http://www.biohedron.com.
[29] M.P. do Carmo, Differential Geometry of Curves and Surfaces, Prentice-Hall, Englewood Cliffs, NJ, 1976.
[30] Sh. Hayryan, C.-K. Hu, J. Skřivánek, E. Hayryan, I. Pokorný, cond-mat/0309170, and J. Comp. Chem., in press.
[31] E.A. Ayrjan, J. Buša, J. Džurina, Sh. Hayryan, J. Plavka, Communication of the Joint Institute for Nuclear Research, Dubna, E11-2002-292, 2002.
[32] B. von Freyberg, W. Braun, J. Comp. Chem. 14 (1993) 510.
[33] http://www.scsb.utmb.edu/cgi-bin/get_a_form.tcl.
[34] A.P. Prudnikov, Yu.A. Brychkov, O.I. Marichev, Integrals and Series (transl. from Russian by N.M. Queen), Gordon and Breach Science Publishers, New York, London, 1988–1992.
[35] We use the mirror site http://pdb.life.nthu.edu.tw.
[36] F. Eisenmenger, U.H.E. Hansmann, Sh. Hayryan, C.-K. Hu, Comput. Phys. Commun. 138 (2001) 192–212.
[37] The website of the Laboratory of Statistical and Computational Physics is http://www.sinica.edu.tw/~statphys/.
[38] Sh. Hayryan, C.-K. Hu, S.-Y. Hu, R.-J. Shang, J. Comput. Chem. 138 (2001) 1287–1296.
[39] C.-Y. Lin, C.-K. Hu, U.H.E. Hansmann, Proteins—Structure Function and Genetics 52 (2003) 436–445.
[40] C.-K. Hu, C.-Y. Lin, J.-A. Chen, Phys. Rev. Lett. 75 (1995) 193–196 and 2786(E).
[41] C.-K. Hu, C.-Y. Lin, J.-A. Chen, Physica A 221 (1995) 80.
[42] C.-K. Hu, C.-Y. Lin, Phys. Rev. Lett. 77 (1996) 8.
[43] C.-K. Hu, F.-G. Wang, J. Korean Phys. Soc. 31 (1997) S271.
[44] H.P. Hsu, S.C. Lin, C.-K. Hu, Phys. Rev. E 64 (2001) 016127.
[45] H. Watanabe, S. Yukawa, N. Ito, C.-K. Hu, C.-Y. Lin, W.J. Ma, J. Phys. Soc. Japan 70 (2001) 1537.
[46] C.-K. Hu, Phys. Rev. B 46 (1992) 6592.
[47] C.-K. Hu, Phys. Rev. Lett. 69 (1992) 2739.
[48] C.-Y. Lin, C.-K. Hu, Phys. Rev. E 58 (1998) 1521.