Solutions to Exercise 4

Algebraic Curve, Surface Splines – IV: Molecular Models

CS384R, CAM 395T, BME 385J: Fall 2007

Question 1. Describe the **LEG** (Labelled Embedded Graph) atomic representations as per class notes, of the twenty protein amino acids (or protein residues), and the two common protein secondary structures (i.e., α-helices and β-sheets).

Solution.

The **LEG** representation of a molecule is simply an annotated graph representation of the chemical structure of the molecule, in which each node represents an atom and each edge a chemical bond. Each atom may be annotated by its symbol and the **vdW** radius, each edge may be annotated by the length of the corresponding chemical bond and possibly a dihedral angle, and each pair of consecutive edges by a bond angle.

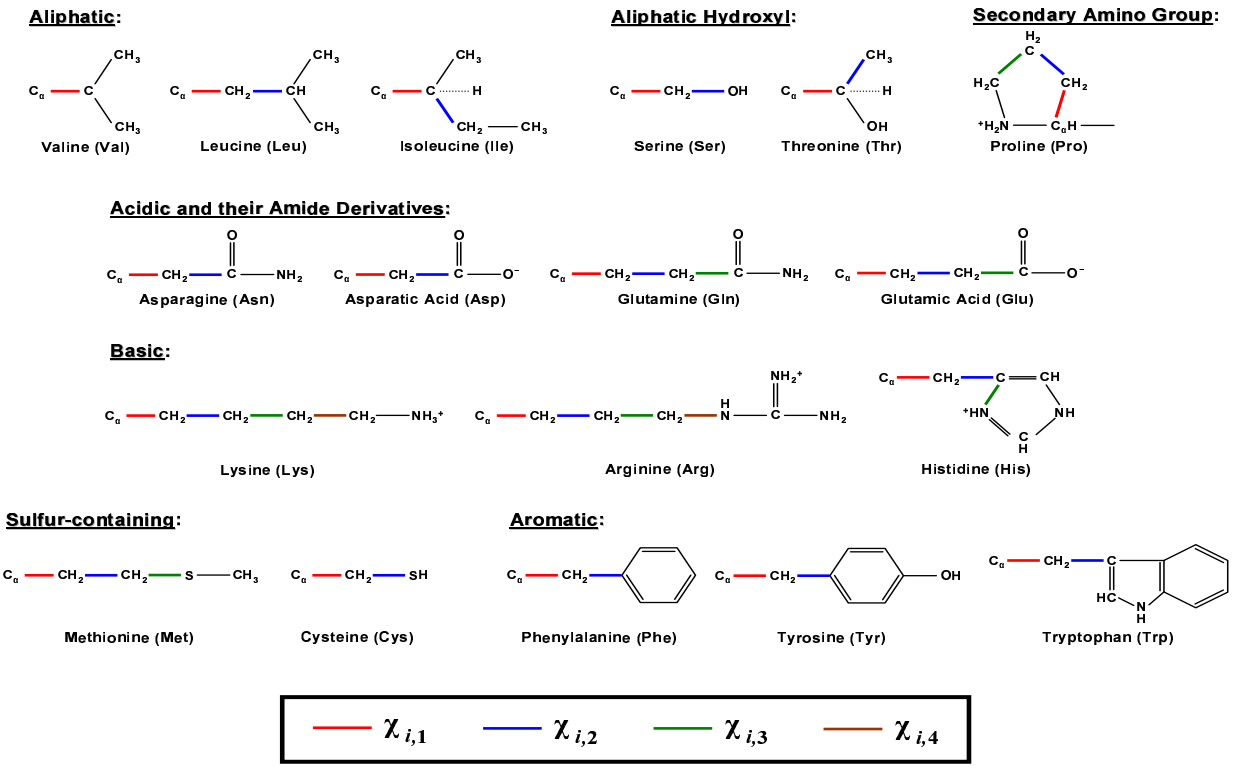


Figure 1: Chemical structures of 18 of the 20 amino acids with their side-chain dihedrals ($\chi_{i,1}$, $\chi_{i,2}$, $\chi_{i,3}$, $\chi_{i,4}$) identified. The remaining two, i.e., Glycine (Gly) and Alanine (Ala), do not have any side-chain dihedrals. Adapted from [8].

In Figure 1 we show the chemical structures of the 20 amino acids, and in Tables 1, 2 and 3 we list all possible **vdW** radii, bond lengths and bond angles, respectively, that appear in these

Atom or Group	Symbol	R_{vdW} (Å)	Notes
>CHR	CA	1.90	Main-chain α-carbon (excluding α-carbon of Gly)
>C=O	C	1.75	Main-chain carbonyl carbon
>CH—	CH	2.01	Side-chain aliphatic carbon with one hydrogen (C^{β} of Ile, C^{γ} of Leu, C^{β} of Thr, C^{β} of Val)
$>CH_2$	CH2	1.92	Side-chain aliphatic carbon with two hydrogens, except those at β-position and those next to a charged group (C^{γ} of Arg, $C^{\gamma 1}$ of Ile, C^{γ} and C^{δ} of Lys, C^{γ} of Met, C^{γ} and C^{δ} of Pro)
$>CH_2^{\beta}$	CH2b	1.91	Side-chain aliphatic carbon with two hydrogens at β-position (C^{β} of Arg, Asn, Asp, Cys, Gln, Glu, His, Leu, Lys, Met, Phe, Pro, Ser, Trp, Tyr)
$>CH_2^{ch}$	CH2ch	1.88	Side-chain aliphatic carbon next to a charged group (C^{δ} of Arg, C^{γ} of Glu, C^{ϵ} of Lys)
$-CH_3$	CH3	1.92	Side-chain aliphatic carbon with three hydrogens (C^{β} of Ala, $C^{\gamma 2}$ and $C^{\delta 1}$ of Ile, $C^{\delta 1}$ and $C^{\delta 2}$ of Leu, $C^{\gamma 2}$ of Thr, $C^{\gamma 1}$ and $C^{\gamma 2}$ of Val)
−CH=	CHar	1.82	Aromatic carbon with one hydrogen (carbon atoms on the rings of Phe, Trp and Tyr)
>C=	Car	1.74	Aromatic carbon with no hydrogen (C^{γ} of Phe, C^{γ} and $C^{\epsilon 2}$ of Trp, C^{γ} of Tyr)
−CH=	CHim	1.74	C^{δ} and C^{ϵ} on the imidazole side-chain of His
>C=O	Cco	1.81	Side-chain carbonyl carbon (C^{γ} of Asn, C^{δ} of Gln)
$-COO^-$	Ccoo	1.76	Side-chain carboxyl carbon (C^{γ} of Asp, C^{δ} of Glu)
−SH	SH	1.88	S on Cys
−S—	S	1.94	S on Met
>NH	N	1.70	Main-chain amide nitrogen
>NH	NH	1.66	Side-chain nitrogen with one hydrogen ($N^{\epsilon 1}$ of Trp)
$>NHn^+$	NH+	1.65	$N^{\delta 2}$ and $N^{\epsilon 1}$ of His (n = 0 or 1; may be partially charged)
$-NH_2$	NH2	1.62	Side-chain neutral nitrogen with two hydrogen ($N^{\delta 2}$ of Asn, $N^{\epsilon 2}$ of Gln)
$-NH_2^+$	NH2+	1.67	Side-chain partially charged nitrogen on Arg
$-NH_3^+$	NH3+	1.67	Side-chain nitrogen on Lys
>C=O	O	1.49	Main-chain carbonyl oxygen
>C=O	Oco	1.52	Side-chain carbonyl oxygen ($O^{\delta 1}$ of Asn, $O^{\epsilon 1}$ of Gln)
$-COO^-$	Ocoo	1.49	Side-chain carboxyl oxygen ($O^{\delta 1}$ and $O^{\delta 2}$ of Asp, $O^{\epsilon 1}$ and $O^{\epsilon 2}$ of Glu)
−OH	OH	1.54	Side-chain hydroxyl oxygen (O^{γ} of Ser, $O^{\gamma 2}$ of Thr, O^{η} of Tyr)
H_2O	H2O	1.68	Water oxygen

Table 1: List of van der Waals radii for 25 protein atoms [7].

Atom type	Description
C	Carbonyl C atom of the peptide backbone
C5W	Tryptophan C^{γ}
CW	Tryptophan $C^{\delta 2}$, $C^{\epsilon 2}$
CF	Phenylalanine C^{γ}
CY	Tyrosine C^{γ}
CY2	Tyrosine C^{ζ}
C5	Histidine C^{γ}
CN	Neutral carboxylic acid group C atom
CH1E	Tetrahedral C atom with one H atom
CH2E	Tetrahedral C atom with two H atoms (except CH2P, CH2G)
CH2P	Proline C^{γ}, C^{δ}
CH2G	Glycine C^{α}
CH3E	Tetrahedral C atom with three H atoms
CR1E	Aromatic ring C atom with one H atom (except CR1W, CRH, CRHH, CR1H)
CR1W	Tryptophan $C^{\zeta 2}$, $C^{\eta 2}$
CRH	Neutral histidine $C^{\epsilon 1}$
CRHH	Charged histidine $C^{\epsilon 1}$
CR1H	Charged histidine $C^{\delta 2}$
N	Peptide N atom of proline
NR	Unprotonated N atom in histidine
NP	Pyrrole N atom
NH1	Singly protonated N atom (His, Trp, peptide)
NH2	Doubly protonated N atom
NH3	Triply protonated N atom
NC2	Arginine $N^{\eta 1}$, $N^{\eta 2}$
O	Carbonyl O atom
OC	Carboxyl O atom
OH1	Hydroxyl O atom
S	S atom
SM	Methionine S atom
SH1E	Singly protonated S atom

Bond type	Bond length (Å)	Bond type	Bond length (Å)
C5W–CW	1·433	CH1E–CH1E	1·540
CW–CW	1·409	CH1E–CH2E	1·530
C–CH1E	1·525	CH1E–CH3E	1·521
C5–CH2E	1·497	CH1E–N	1·466
C5W–CH2E	1·498	CH1E–NH1	1·458
CF–CH2E	1·502	CH1E–NH3	1·491
CY–CH2E	1·512	CH1E–OH1	1·433
C–CH2E	1·516	CH2E–CH2E	1·520
CN–CH2E	1·503	CH2P–CH2E	1·492
C–CH2G	1·516	CH2P–CH2P	1·503
C5W–CR1E	1·365	CH2E–CH3E	1·513
CW–CR1E	1·398	CH2P–N	1·473
CW–CR1W	1·394	CH2G–NH1	1·451
CF–CR1E	1·384	CH2E–NH1	1·460
CY–CR1E	1·389	CH3E–NH1	1·460
CY2–CR1E	1·378	CH2E–NH3	1·489
C5–CR1H	1·354	CH2E–OH1	1·417
C5–CR1E	1·356	CH2E–S	1·822
C–N	1·341	CH2E–SM	1·803
C–NC2	1·326	CH2E–SH1E	1·808
C5–NH1	1·378	CH3E–SM	1·791
CW–NH1	1·370	CR1E–CR1E	1·382
C–NH1	1·329	CR1E–CR1W	1·400
C–NH2	1·328	CR1W–CR1W	1·368
C5–NR	1·371	CR1E–NH1	1·374
C–O	1·231	CRH–NH1	1·345
CN–O	1·208	CRHH–NH1	1·321
C–OC	1·249	CR1H–NH1	1·374
CY2–OH1	1·376	CRH–NR	1·319
C–OH1	1·304		

Table 2: Bond lengths in proteins [3].

Angle type	Angle (°)	Angle type	Angle (°)
C5W–CW–CW	107·2	CH3E–CH1E–CH3E	110·8
CW–C5W–CH2E	126·8	CH3E–CH1E–NH1	110·4
C5W–CW–CR1E	133·9	CH3E–CH1E–OH1	109·3
CW–CW–CR1E	118·8	C–CH2E–CH1E	112·6
CW–CW–CR1W	122·4	C5–CH2E–CH1E	113·8
CW–C5W–CR1E	106·3	CF–CH2E–CH1E	113·8
CW–CW–NH1	107·4	C5W–CH2E–CH1E	113·6
CH1E–C–N	116·9	CY–CH2E–CH1E	113·9
CH1E–C–NH1	116·2	C–CH2E–CH2E	112·6
CH1E–C–O	120·8	C–CH2G–NH1	112·5
CH1E–C–OC	117·0	C–CH2G–NH3	112·5
CH2E–C5–CR1E	129·1	CH1E–CH2E–CH1E	116·3
CH2E–C5–CR1H	131·2	CH1E–CH2E–CH2P	104·5
CH2E–CF–CR1E	120·7	CH1E–CH2E–CH2E	114·1
CH2E–C5W–CR1E	126·9	CH1E–CH2E–CH3E	113·8
CH2E–CY–CR1E	120·8	CH1E–CH2E–OH1	111·1
CH2E–C–N	118·2	CH1E–CH2E–S	114·4
CH2G–C–N	118·2	CH1E–CH2E–SH1E	114·4
CH2E–C5–NH1	122·7	CH2E–CH2E–CH2E	111·3
CH2E–C–NH1	116·5	CH2E–CH2P–CH2P	106·1
CH2G–C–NH1	116·4	CH2P–CH2P–N	103·2
CH2E–C–NH2	116·4	CH2E–CH2E–NH1	112·0
CH2E–C5–NR	121·6	CH2E–CH2E–NH3	111·9
CH2E–C–O	120·8	CH2E–CH2E–SM	112·7
CH2G–C–O	120·8	CY2–CR1E–CR1E	119·6
CH2E–C–OC	118·4	CW–CR1E–CR1E	118·6
CH2G–C–OC	118·4	CW–CR1W–CR1W	117·5
CR1E–CY2–CR1E	120·3	CF–CR1E–CR1E	120·7
CR1E–CY–CR1E	118·1	CY–CR1E–CR1E	121·2
CR1E–CF–CR1E	118·6	C5–CR1E–NH1	106·5
CR1W–CW–NH1	130·1	C5–CR1H–NH1	107·2
CR1E–C5–NH1	105·2	C5W–CR1E–NH1	110·2
CR1H–C5–NH1	106·1	C5–CR1E–NR	109·5
CR1E–CY2–OH1	119·9	CR1E–CR1E–CR1W	121·1
N–C–O	122·0	CR1W–CR1W–CR1E	121·5
NC2–C–NC2	119·7	CR1E–CR1E–CR1E	120·0
NC2–C–NH1	120·0	NH1–CRHH–NH1	108·4
NH1–C–O	123·0	NH1–CR1E–NR	111·7
NH2–C–O	122·6	C–N–CH1E	122·6
OC–C–OC	122·9	C–N–CH2P	125·0
C–CH1E–CH1E	109·1	CH1E–N–CH2P	112·0
C–CH1E–CH2E	110·1	C–NH1–CH1E	121·7
C–CH1E–CH3E	110·5	C–NH1–CH2G	120·6
C–CH1E–N	111·8	C–NH1–CH2E	124·2
C–CH1E–NH1	111·2	C–NH1–CH3E	120·6
C–CH1E–NH3	111·2	C5–NH1–CRHH	109·3
CH1E–CH1E–CH2E	110·4	C5–NH1–CRH	109·0
CH1E–CH1E–CH3E	110·5	CW–NH1–CR1E	108·9
CH1E–CH1E–NH1	111·5	CRHH–NH1–CR1H	109·0
CH1E–CH1E–OH1	109·6	CRH–NH1–CR1E	106·9
CH2E–CH1E–CH3E	110·7	C5–NR–CR1E	105·6
CH2E–CH1E–N	103·0	CR1E–NR–CR1E	107·0
CH2E–CH1E–NH1	110·5	CH2E–SM–CH3E	100·9
CH2E–CH1E–NH3	110·5	CH2E–S–S	103·8

Table 3: Bond angles in proteins [3].

chemical structures. Using these information, it is straight-forward to construct the required **LEG** representations of the amino acids.

Since secondary structures (e.g., α-helices and β-sheets) are composed of primary structures (i.e., amino acids), the **LEG** representation of secondary structures can also be constructed from the information in Figure 1 and Tables 1, 2 and 3. However, the (ϕ, ψ) dihedral angles of the residues in α-helices and β-sheets lie in fairly restricted ranges: $(-45°, -60°)$ for α-helices, $(-120°, 115°)$ for parallel β-sheets, and about $(-140°, 135°)$ for anti-parallel β-sheets. The bond lengths and bond angles may also change slightly.

Question 2. Given a **LEG** representation of a protein (created from the PDB), describe an algorithm to detect and output the **LEG** representations of all α-helices and β-sheets in that protein. Your algorithm should be able to distinguish between *parallel* and *anti-parallel* β-sheets.

Solution.

We will use geometric properties of α-helices and β-sheets in order to extract them from the **LEG** representation L of the given protein P.

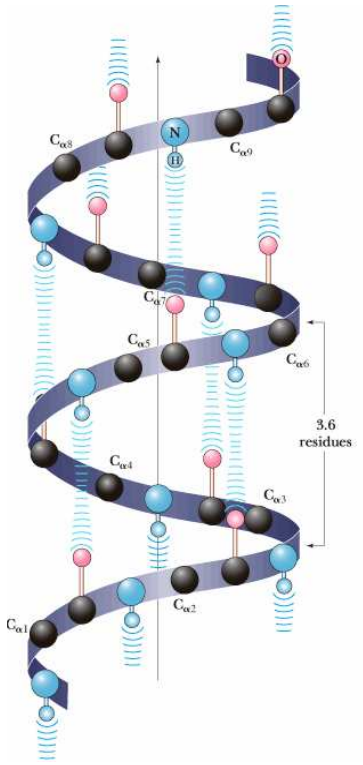


Figure 2: Geometric structure of an α-helix [4].

Extracting α-helices from L. We traverse L along the peptide backbone of P, and using the internal coordinates (i.e., bond lengths, bond angles, dihedral angles, etc.), bond types and atom types specified in L, we detect and output all maximal contiguous segments of this backbone (along with side chains) that satisfy the following properties of α-helices.

- The amino acids in an α-helix are arranged in a right-handed helical structure with each amino acid corresponding to a 100° turn in the helix and a 1.5 Å translation along the helical axis. Thus there are 13 atoms and 3.6 amino acid residues per turn, and each turn is 5.4 Å wide (see Figure 2).
- The $C{=}O$ group of residue i forms a hydrogen bond with the N-H group of residue $i+4$.
- Amino acid residues in an α-helix typically have dihedral angles $\phi \approx -45°$ and $\psi \approx -60°$.

Extracting β-sheets from L. We scan the peptide backbone of P given in L, and detect and output all maximal contiguous segments of this backbone (along with side chains) that satisfy the following properties of β-sheets.

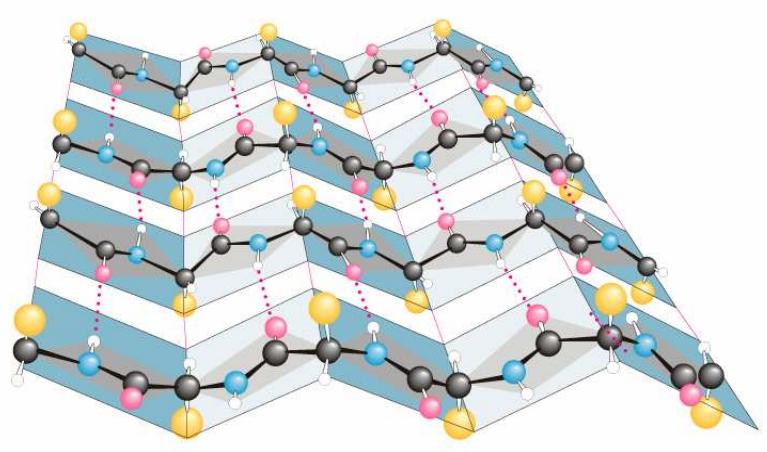


Figure 3: Geometric structure of a β-sheet [4].

- Each β-strand can be viewed as a helical structure with two residues per turn. The distance between two such consecutive residues is 3.47 Å in anti-parallel β-sheets and 3.25 Å in parallel β-sheets.
- Unlike α-helices the $C{=}O$ groups in the backbone of a β-strand form hydrogen bonds with the N-H groups in the backbone of adjacent strands.
  - In parallel β-sheets all N-termini of adjacent strands are oriented in the same direction (see Figure 4(b)). If the C_α atoms of residues i and j of two different strands are adjacent, they do not hydrogen bond to each other, rather rasidue i may form hydrogen bonds to residues $j-1$ or $j+1$ of the other strand.
  - In anti-parallel β-sheets the N-terminus of one strand is adjacent to the C-terminus of the next strand (see Figure 4(a)). If a pair of C_α atoms from two successive β-strands are adjacent, then unlike in parallel β-sheets they form hydrogen bonds to each other's flanking peptide groups.
- The (ϕ, ψ) dihedrals are about $(-120°, 115°)$ in parallel β-sheets, and about $(-140°, 135°)$ in anti-parallel β-sheets.
- Unlike in α-helices, peptide carbonyl groups in successive residues point in alternating directions.

Question 3. Given a **LEG** representation of a protein P,

(a) Describe an algorithm to compute the **vdW** (union-of-spheres) surface of P.

(b) Describe an algorithm to detect all solvent exposed atoms of P.

(c) Augment the algorithm of part (b) to detect where two or three of these exposed atoms intersect.

(d) Describe how to construct the **L-R** molecular surface (also called a sphere solvent contact surface) of the protein P, using the information generated in parts (b) and (c).

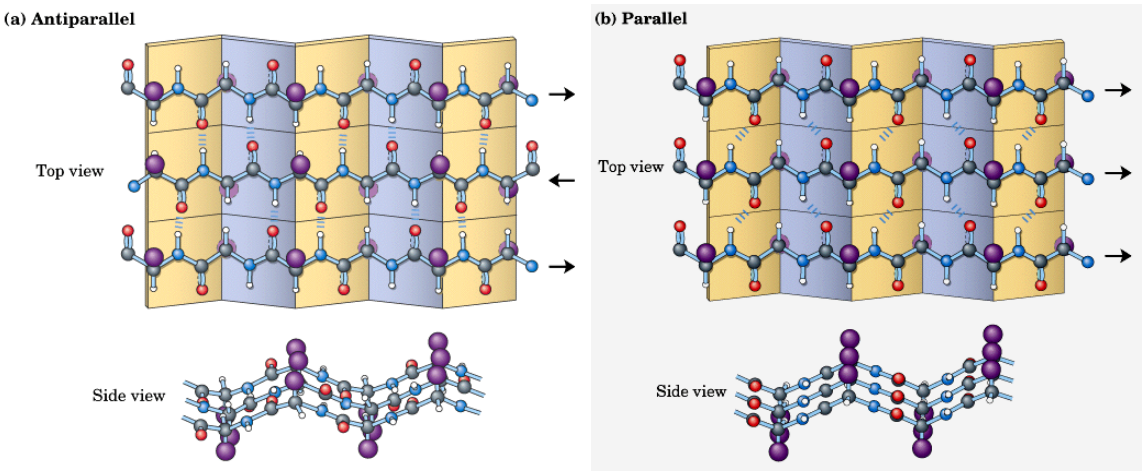


Figure 4: Two types of β-sheets: (a) anti-parallel, and (b) parallel [6].

(e) Describe a method to detect where if at all, the **L-R** surface of part (d), self intersects.

(f) Can you solve parts (b) and (c) in $\mathcal{O}(n \log n)$ time, where n is the number of atoms in the protein? You can assume for simplicity that all atoms have the same radius.

Solution.

Let us first explore a couple of properties of a moleclue (described in [5]) that can be exploited to design efficient algorithms for manipulating the "union-of-sphere" model of the molecule (e.g., for computing the molecular surface). In the worst case, the arrangement defined by n balls in 3-space (i.e., the subdivision of 3-space into cells of dimensions 0, 1, 2, and 3, defined by the balls) may have $\mathcal{O}(n^3)$ combinatorial complexity, the boundary defined by their union may have complexity $\mathcal{O}(n^2)$. However, the balls defining the atoms in the "union-of-sphere" model of a molecule have the following two proporties which allow for more efficient and simpler algorithms for manipulating them:

- The centers of two balls cannot get too close to each other.
- The range of radii of the balls is fairly restricted (e.g., see Table 4 below).

C	Cal	H	N	O	P	S
1.52 Å	3.48 Å	0.70 Å	1.36 Å	1.28 Å	2.18 Å	2.10 Å

Table 4: Radii of balls used to represent different types of atoms [5].

The following theorem, proved in [5], gives a couple of useful consequences of the two properties listed above.

Theorem 1 (Theorem 2.1 in [5]). *Let $M = \{B_1, \ldots, B_n\}$ be a collection of n balls in 3-space with radii $r_1, \ldots, r_n$ and centers at $c_1, \ldots, c_n$. Let $r_{min} = \min_i \{r_i\}$ and let $r_{max} = \max_i \{r_i\}$. Also let $S = \{S_1, \ldots, S_n\}$ be the collection of spheres such that S_i is the boundary surface of B_i. If there are*

positive constants k, ρ such that $\frac{r_{max}}{r_{min}} < k$ and for each B_i the ball with radius $\rho \cdot r_i$ and concentric with B_i does not contain the center of any other ball in M (besides c_i), then:

(*i*) *For each $B_i \in M$, the maximum number of balls in M that intersect it is bounded by a constant.*

(*ii*) *The maximum combinatorial complexity of the boundary of the union of the balls in M is $\mathcal{O}(n)$.*

Table 5 lists the values of k, ρ and the maximum and average number of balls intersecting any given ball in various molecules [5]. As the table shows, k is quite small and ρ is quite close to 1, resulting in a small number of insersections per ball.

molecule	k	ρ	maximum number of balls intersecting a given ball	avgerage number of balls intersecting a given ball
caffine	2.17	0.71	10	4.5
acetyl	3.11	0.67	16	5.4
crambin	1.64	0.78	10	5.5
felix	1.64	0.81	9	4.9
SuperOxide Dismutase	1.95	0.76	16	5.5

Table 5: Values of k, ρ and the maximum and average number of balls intersecting a single ball in various molecules [5].

Given a "union-of-ball" representation of a molecule, Theorem 1 can now be used to design an efficient data structure that can answer insersection queries with either a point or with a ball whose radius is bounded by a r_{max}. We will use this data structure in parts $3(a)$ – $3(f)$.

An Efficient Intersection Query Data Structure (from [5]). Let M be the set of n balls as defined in Theorem 1. We subdivide the entire 3-space into axis-parallel cubes of size $2r_{max} \times 2r_{max} \times 2r_{max}$ each. For each $B \in M$, we compute the grid cubes that B intersects. Let C be the set of non-empty grid cubes. Since each ball can intersect at most 8 grid cubes, the size of C is bounded by $\mathcal{O}(n)$. Also observe that according to Theorem 1, each cube can be intersected by at most a constant number of balls. We store the cubes in C in a balanced binary search tree ordered lexicographically by the bottom-left-front vertices of the cubes. With each cube we store the list of $\mathcal{O}(1)$ balls of M that intersects it.

Now given a query ball Q, we compute all (at most 8) grid cubes it intersects, and search for each of these cubes in the binary search tree. For each such cube that exists in the search tree, we check the balls stored in it for intersection with Q. Each search will take $\mathcal{O}(\log n)$ time, and the total number of balls tested will be $\mathcal{O}(1)$. Hence, we have the following theorem.

Theorem 2 (Theorem 3.1 in [5])**.** *Given a collection M of n balls as defined in Theorem 1, one can construct a data structure using $\mathcal{O}(n)$ space and $\mathcal{O}(n \log n)$ preprocessing time, to answer intersection queries for balls whose radii are not greater than r_{max}, in time $\mathcal{O}(\log n)$.*

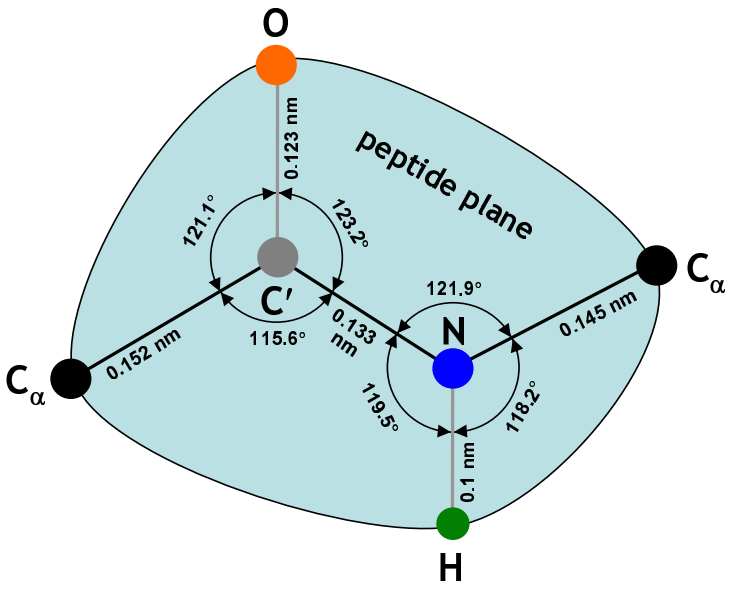


Figure 5: A peptide plane with all bond lengths and bond angles shown [4].

Part 3(a): We first convert the **LEG** representation of P to the "union-of-spheres" representation, and then compute its boundary surface.

LEG to "Union-of-Spheres" Conversion. For each ball B_i in the union we need to compute its center c_i and radius r_i. The r_i value is simply the van der Waals (**vdW**) radius of the atom, and can be obtained from various sources (e.g., [2], see also Table 4). The **LEG** representation itself might be annotated with the **vdW** radius of each atom, However, since **vdW** radii are not standardized, values obtained from different sources might differ slightly. The c_i values can be computed easily using the internal coordinates (i.e., bond lengths, bond angles and dihedral angles) specified in the **LEG** representation. For example, we can choose the N atom on an arbitrary peptide plane (see Figure 5) of the protein, and put the atom (i.e., its center) at the origin. The C_α atom connected to the N atom is placed at distance 1.45 Å from the origin along the positive x-axis. The H atom connected to the N atom is then placed on the xy plane using the bond length N-$H = 1$ Å and the bond angle C_α-N-$H = 118.2°$. After the peptide plane containing these three atoms are fixed, it is straight-forward to compute the coordinates of the remaining atom centers using the given internal coordinates.

Computing the vdW Surface [5]. Given a collection M of balls as defined in Theorem 1, the algorithm proceeds in the following three steps:

1. For each $B \in M$, compute the balls in $M \setminus \{B\}$ intersecting it.
2. Using the information generated in step 1, compute the (potentially null) contribution of each ball $B \in M$ to the union boundary.
3. Transform the local information generated in step 2 into global structures describing the required connected component of the union boundary.

Each step is described in more details below.

Step 1: We use the intersection query data structure described earlier (see Theorem 2). Each intersection query takes $\mathcal{O}(\log n)$ time, and hence the total cost of this step is $\mathcal{O}(n \log n)$.

Step 2: Let B_i be a ball, and we want to compute its contribution to the union boundary. We know from Theorem 1 that at most a constant number of other balls intersect B_i. Let B_j be such a ball, if any. We consider the following three cases:

(i) If B_j fully contains B_i, we stop processing B_i as it cannot contribute to the union boundary.

(ii) If B_i fully contains B_j, we simply ignore B_j.

(iii) If neither of the two cases above holds, we compute the intersection between S_i and S_j, which is a circle C_{ij} on S_i (and S_j). This circle splits S_i into two parts: one part is completely contained within S_j and hence cannot contribute to the union boundary, and the other part which is called the *free* part of S_i, may actually appear on the union boundary.

After the process above is repeated for every ball intersecting B_i, we get a collection of circles on S_i. These circles form a 2D arrangement A_i on S_i. A face of A_i belongs to the union boundary iff it is on the free part defined by each such circle. Since the number of such circles is $\mathcal{O}(1)$, A_i can be computed in $\mathcal{O}(1)$ time using brute force. Within the same bound we can mark each face on A_i as *free* or *not free*. A free face is guaranteed to appear on the union boundary. Since the above procedure for B_i takes $\mathcal{O}(1)$ time, the total time complexity of this step is $\mathcal{O}(n)$.

Step 3: In this step, the outer connected component of the union bounadry of M is represented using a graph data structure. In order to do so, each arrangement A_i is augmented slightly as follows. If A_i is the whole sphere S_i, it is split into two parts using some circle. Next, if a boundary component of A_i is a simple circle C, then C is split into two arcs by adding two new vertices. If C belongs to two arangements A_i and A_j, the same two vertices are added to both arrangements. Finally, if a free face of A_i contains holes, the face is split into (sub)faces by adding extra arcs so that none of (sub)faces contains any holes in it. All these additions are made canonical by fixing a direction d, and adding all extra edges along great circles that are intersections of S_i with planes parallel to the direction d. This step can easily be performed in $\mathcal{O}(n)$ time, and after this step the union boundary will consist of only simple faces that are bounded by at least two edges, and each edge will be shared by exactly two faces (assuming general position).

Now we compute the **vdW** surface (i.e., the outer union boundary) of M and store it as a graph $G = (V, E)$ (which is initially empty) as follows. First, we find the ball $B_{max} \in M$ with the point having the largest z-coordinate. Let f_{max} be the face of B_{max} that contains this point. Then clearly f_{max} belongs to the outer union boundary of M. Now starting from this face f_{max} we traverse the entire connected component containing f_{max} using depth-first search. Each free face f we encounter during this traversal will be made a vertex $v_f \in V$, and each arc shared by two such faces f_1 and f_2 will be made an edge $(v_{f_1}, v_{f_2}) \in E$. Everytime we encounter a free face f which has not been visited before, we determine its boundary (i.e., the edges bounding this face), which can be done in $\mathcal{O}(1)$ time. For each such edge e of f, we can find the face f' that shares e in $\mathcal{O}(1)$ time. If f' is a free face and has not been visited before, we recursively visit f'. After we have visited each free face reachable from f_{max} once, G contains the **vdW** surface of M. Clearly, this traversal takes $\mathcal{O}(1)$ time.

Hence, the following theorem follows.

Theorem 3 (Theorem 4.1 in [5])**.** *The* **vdW** *surface of the union of a connected collection of balls as defined in Theorem 1 can be computed in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space.*

Part 3(b): We increase the radius of each atom in P by r_s, where r_s is the radius of a solvent atom. Clearly, the collection of these enlarged atoms still satisfies the requirements of Theorem 1, and the theorem holds. Hence, we can find all atoms in this collection that contribute to the outer union boundary in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space using the same algorithm as in part 3(a).

Part 3(c): The algorithm in part 3(b) already constructs the intersection query data structure described earlier for the set of enlarged atoms in P. If P contains n atoms, this construction takes $\mathcal{O}(n \log n)$ time and uses $\mathcal{O}(n)$ space. The algorithm also identifies all solvent exposed atoms. Now, for any ball B_i representing a solvent exposed atom in P, we can find the set T_i of $\mathcal{O}(1)$ other solvent exposed atoms that intersects it in $\mathcal{O}(\log n)$ time. Since the size of the set $T_i \cup \{B_i\}$ is bounded by a constant, we can detect in $\mathcal{O}(1)$ time where two or three of the atoms in this set intersect. We can identify all such intersections in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space by using the same process as above for each solvent exposed atoms in P.

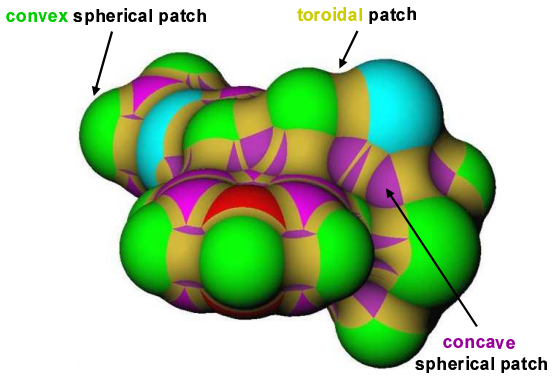


Figure 6: 3D image showing the decomposition of the **L-R** surface into three different kinds of patches: convex spherical, toroidal and concave spherical.

Part 3(d): The **L-R** surface of a molecule M with respect to a solvent atom B of radius r is the inner envelope of the region described by B rolling on the **vdW** surface $\mathcal{B}$ of M in all possible directions [1]. This surface can be decomposed into a collection of three kinds of patches: convex spherical, toroidal and concave spherical (see Figure 6).

We can detect the locations of these patches using the information generated in parts 3(b) and 3(c), and thus construct the entire **L-R** surface.

Convex Spherical Patches. A convex spherical patch is formed when the rolling solvent atom B is in contact with only one atom $B_i \in M$, and it is the maximal connected set of points on the spherical surface S_i of B_i that B touches in this manner. In order to find the extent of the spherical patch on S_i that belongs to the **L-R** surface, we simply increase the radius of all balls within distance $2r_{max} + r$ of B_i by r (by Theorem 1 there are only a constant number of them), and use the method in step 2 of part 3(a).

Toroidal Patches. A toroidal patch is formed when the rolling solvent atom B (of radius r) is in touch with the spherical surfaces S_1 and S_2 of two solute atoms. We increase the radius of S_1 by r and compute its intersection circle l_2 with S_2, and similarly increase the radius of S_2 by r and compute its intersection l_1 with S_1. Now if we move B along the intersection of S_1 and S_2, it will

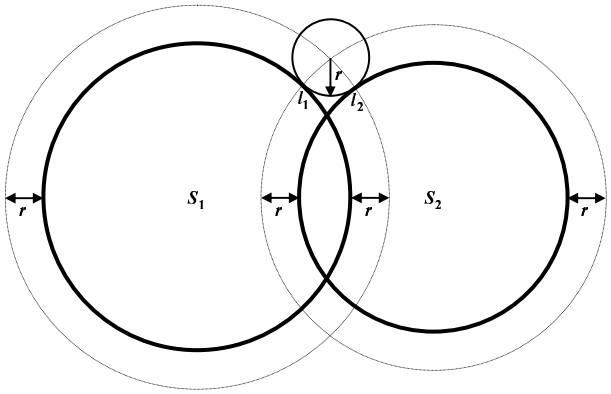


Figure 7: The solvent atom of radius r sweeps a torus if it is moved in such a way that it is always in touch with the spheres S_1 and S_2. The line l_1 (l_1) along which it keeps in touch with S_1 (resp. S_2) can be found by increasing the radius of S_2 (resp. S_1) by r and computing its intersection with S_1 (resp. S_2).

keep in touch with the two spheres along l_1 and l_2, respectively, and the inward facing arc of B will sweep a torus (see Figure 7). Other atoms intersecting with S_1 and S_2 may split this torus into several toroidal patches.

Concave Spherical Patches. A concave spherical (triangular) patch is formed when the rolling solvent atom B simultaneously touches three atoms. The three contact points define a spherical triangle on the surface S of B whose edges are arcs of great circles on S. This triangle is a concave spherical patch on the **L-R** surface.

Part 3(e): We describe below how the three different types of patches (i.e., convex spherical, toroidal and concave spherical) on an **L-R** surface can intersect one another, and how to detect them.

Convex Spherical Patches. It has been shown in [1] (see Lemma 3 in [1]) that the convex spherical patches cannot intersect any other part of the **L-R** surface.

Toroidal Patches. It has been shown in [1] that two different toroidal patches cannot intersect each other (see Lemma 4 in [1]), and also that a toroidal patch cannot intersect another convex/concave spherical patch (see Lemma 5 in [1]).

A toroidal can only intersect itself. As shown in Figure 8, a toroidal patch intersects itself when it can be constructed as a rotational surface of an arc of a circle around an axis that intersects the arc.

Concave Spherical Patches. It has been shown in [1] that a concave spherical patch cannot intersect itself (see Lemma 6 in [1]), or another concave patch (see Lemma 7 in [1]), or a toroidal patch (see Lemma 5 in [1]).

As shown in Figure 9 two distinct concave patches can intersect each other. Since each concave patch is a part of a sphere, we can detect this type of intersections easily by solving sphere-sphere intersection problems.

Part 3(f): The algorithms in parts 3(b) and 3(c) already run in $\mathcal{O}(n \log n)$ time.

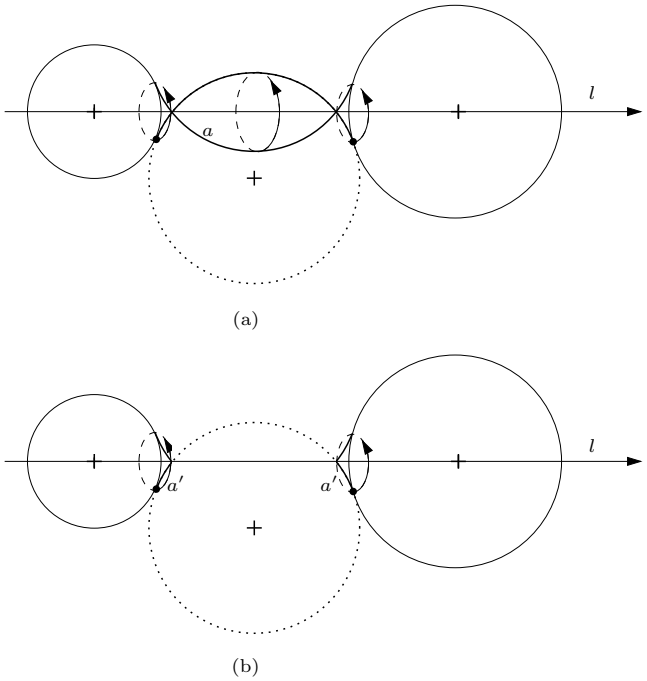


Figure 8: (a) The arc a rotating around the axes l describes a self intersection portion of torus. (b) The arc a' rotating around the axes l describes portion of torus with no self intersection.

References

[1] Bajaj, C., Lee, H. Y., Merkert, R., and Pascucci, V. Nurbs based b-rep models for macromolecules and their properties. In *SMA '97: Proceedings of the fourth ACM symposium on Solid modeling and applications* (New York, NY, USA, 1997), ACM, pp. 217–228.

[2] Batsanov, S. Van der waals radii of elements. *Inorganic Materials 37* (September 2001), 871–885(15).

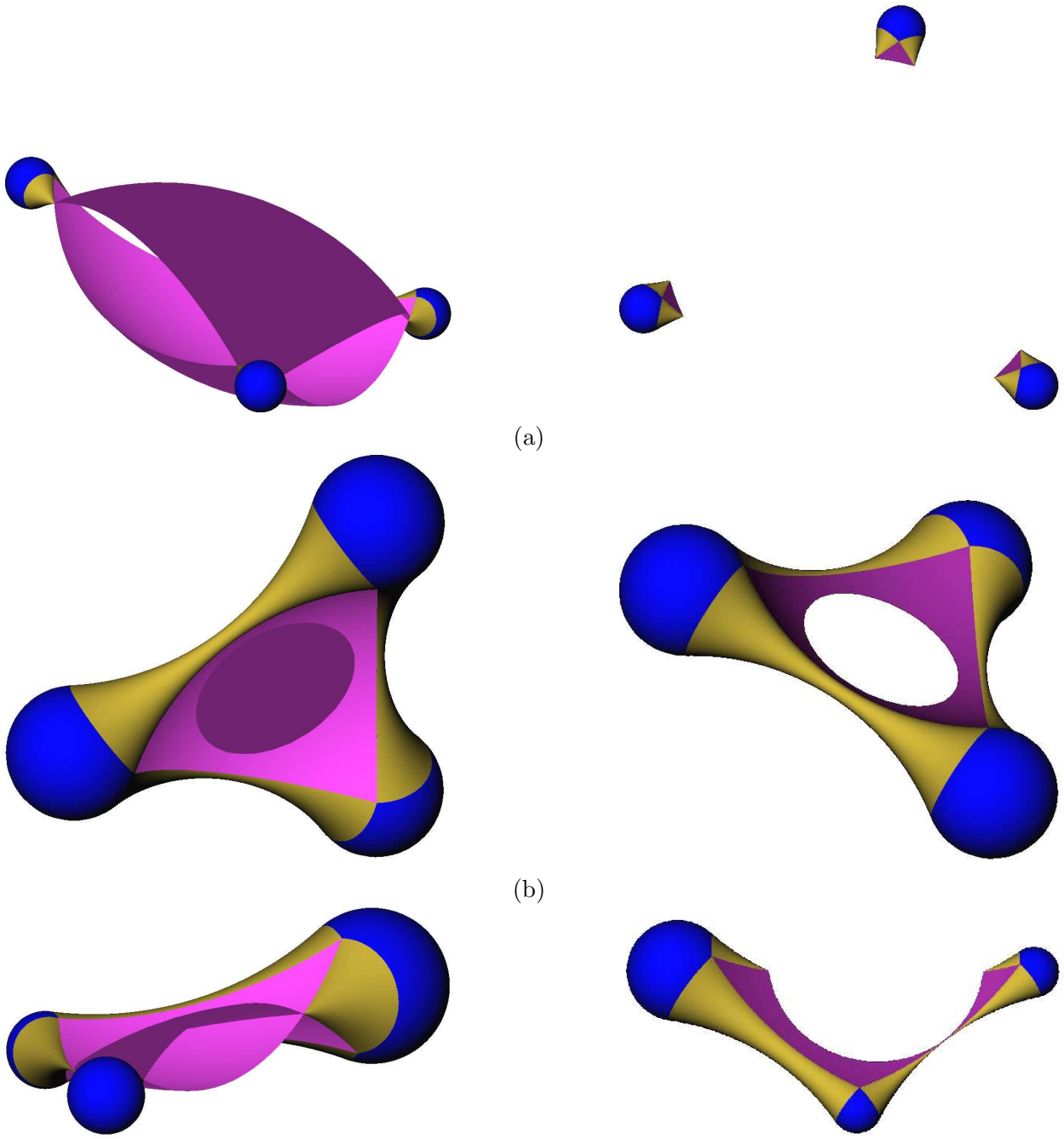
[3] Engh, R. A., and Huber, R. Accurate bond and angle parameters for x-ray protein structure refinement. *Acta Crystallographica Section A 47*, 4 (July 1991), 392–400.

[4] Garrett, R., and Grisham, C. *Biochemistry*, 2nd ed. Saunders Collge Publishing, New York, 1999.

[5] Halperin, D., and Overmars, M. H. Spheres, molecules, and hidden surface removal. In *SCG '94: Proceedings of the tenth annual symposium on Computational geometry* (New York, NY, USA, 1994), ACM, pp. 113–122.

[6] Horton, H. R., Moran, L. A., Ochs, R. S., Rawn, D. J., and Scrimgeour, K. G. *Principles of Biochemistry*, 3rd ed. Prentice Hall, July 2002.

[7] Li, A.-J., and Nussinov, R. A set of van der waals and coulombic radii of protein atoms for molecular and solvent-accessible surface calculation, packing evaluation, and docking. *Proteins: Structure, Function, and Genetics 32*, 1 (1998), 111–127.



(a)

(b)

(c)

Figure 9: Three possible self-intersecting **L-R** surfaces for different radii of the solvent and molecule atoms. On the left the self-intersecting **L-R** surfaces are shown. On the right the corresponding solvent contact surfaces are shown (without self-intersections).

[8] Schlick, T. *Molecular Modeling and Simulation: An Interdisciplinary Guide*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.