Computing the weighted flow complex*

Joachim Giesen Matthias John

Institut für Theoretische Informatik
ETH Zürich, CH-8092 Zürich, Switzerland
{giesen,john}@inf.ethz.ch

Abstract. The flow complex was introduced recently as a data structure on a finite number of points in $\mathbb{R}^3$. The flow complex is a two dimensional simplicial complex that turned out to be useful for modeling applications like surface reconstruction. In order to apply the flow complex to tasks in bio-geometric modeling an extension to weighted points is needed. Here we report on an algorithm to compute the weighted flow complex, its implementation and two applications in bio-geometric modeling.
Keywords. flow complex, bio-geometric modeling

1 Introduction

In [9, 10] we introduced the flow complex as a data structure on finite point sets in $\mathbb{R}^3$ and applied it successfully to the problem of surface reconstruction. The flow complex is a cell complex where each cell can be triangulated. It is closely related to the Delaunay triangulation of the same point set. In fact, our efficient algorithm to compute the flow complex is based on the Delaunay triangulation. But neither complex is a subcomplex of the other.

Inspired by the work of Edelsbrunner et al. [6] we want to apply the flow complex also for tasks in bio-geometric modeling. In bio-geometry molecules are often modeled as a union of balls or positively weighted points. That is, the input that we have to structure is a finite set of weighted points in contrast to unweighted points that we structure in the flow complex. Hence to apply the flow complex to bio-geometric modeling an extension of its definition to weighted points is necessary. It turns out that the definition can be extended quite easily to capture also the weighted case. But this is not true for the algorithm to compute the flow complex. The reason is that the structure of the 1- and 2-cells of the weighted flow complex can be much more complicated than their counterparts in the unweighted flow complex. This is due to the fact that not every weighted point corresponds to a vertex in the weighted flow complex, but still can have an influence on the structure. That makes it more complicated to derive the weighted flow complex from the weighted Delaunay diagram than it is to derive the unweighted flow diagram from the Delaunay triangulation.

The purpose of this paper is to show how the complications caused by introducing weights can be resolved and to demonstrate that the weighted flow complex can be a useful data structure for bio-geometric modeling.

The paper is organized as follows: In the second section we introduce the weighted flow complex. Starting point of our study is a distance function associated with a finite set P of weighted points in $\mathbb{R}^3$. This function assigns to every point in $\mathbb{R}^3$ its least power distance to any of the points in P. It is intimately related to the power diagram of P. The distance function has a unique direction of steepest ascent at almost every point in $\mathbb{R}^3$. The points where such a direction does not exist are the critical points of the distance function, i.e. its local extrema and saddle points. We study where a point in $\mathbb{R}^3$ flows if it always follows the direction of steepest ascent of the distance function. It turns out that all points either flow into a critical point or to infinity. The set of all points that flow into a certain critical point is called the stable manifold of this critical point. We call the collection of all stable manifolds the weighted flow complex induced by P.

In the third section we give an algorithmic characterization of the 0-, 1- and 2-cells of the weighted flow complex and implicitly characterize its 3-cells. The algorithm to compute the 2-cells is especially important since it reveals the nature of the flow complex in general, i.e. the recursive structure of the algorithm can in principle be generalized to compute higher order cells in weighted or unweighted flow complexes in higher dimensions.

In the fourth section we present two applications of the weighted flow complex in bio-geometric modeling. First, we use it to decompose macromolecules into their constituents. Second, we give an alternative definition of pockets [6], i.e. essential cavities, in macromolecules based on the weighted flow complex. Our definition is not equivalent to the definition in [6]. We think it is easier to grasp since it avoids some technicalities and directly builds on the intuitive notion of a pocket.

We conclude the paper with the fifth section where we discuss our implementations of the presented algorithms and the results we got in the aforementioned applications.

* Partly supported by the IST Programme of the EU and the Swiss Federal Office for Education and Science as a Shared-cost RTD (FET Open) Project under Contract No IST-2000-26473 (ECG - Effective Computational Geometry for Curves and Surfaces).

2 Weighted flow complex

A *weighted point* p in three dimensional Euclidean space is a tuple (z, r) where $z \in \mathbb{R}^3$ denotes the point itself and $r \in \mathbb{R}$ its weight. Every weighted point gives rise to a distance function $\pi_p : \mathbb{R}^3 \to \mathbb{R}$, namely the *power distance function*. The power distance of a point $x \in \mathbb{R}^3$ from a weighted point p is defined as

$$\pi_p(x) = \|x - z\|^2 - r^2.$$

Let P be a finite set of weighted points. From all the power distance functions $\pi_p, p \in P$, together, we derive a distance function $h : \mathbb{R}^3 \to \mathbb{R}$ which assigns to every point in $\mathbb{R}^3$ its least power distance to any weighted point in P, i.e.

$$h(x) = \min_{p \in P} \pi_p(x).$$

We are interested in the gradient vector field of h and its critical points, i.e. its local minima, local maxima and saddle points. Extra care has to be taken since h is not smooth everywhere. That is, the ordinary theory of gradients and critical points does not apply here. Instead we are going to apply the critical point theory of distance functions that was developed by Grove [11]. To do so we associate with every point $x \in \mathbb{R}^3$ the subset $A(x) \subset P$ which contains the nearest neighbors of x in P, namely

$$A(x) = \{p \in P : h(x) = \pi_p(x)\}.$$

Note that $|A(x)| \geq 1$. Let $H(x)$ be the convex hull of the points in $A(x)$. We call the point x *critical* if it is contained in $H(x)$ otherwise we call it *regular*. The index of a critical point x is the dimension of $H(x)$. It can be shown that a critical point x is

a local minimum , if $\dim H(x) = 0$.
a saddle point , if $\dim H(x) = 1$ or 2.
a local maximum , if $\dim H(x) = 3$.

The distance function h and its critical points are closely related to the *power-* and the *weighted Delaunay diagram* of P. The power diagram of P is a decomposition of $\mathbb{R}^3$ into the *power cells* of the points in P. The power cell of $p \in P$ is given as

$$V_p = \{x \in \mathbb{R}^3 : \forall q \in P, \pi_p(x) \leq \pi_q(x)\},$$

i.e. it contains all points of $\mathbb{R}^3$ that do not have a larger power distance to p than to any other point in P. That is, V_p contains exactly the points where value of h is determined by p. The points that have the same power distance from two weighted points in P form a hyperplane. Thus V_p is either a convex polyhedron or empty. Closed facets shared by two power cells are called *power facets*, closed edges shared by three or more power cells are called *power edges* and the points shared by four or more power cells are called *power vertices*. The term *power object* can denote either a power cell, facet, edge or vertex. The *power diagram* of P is the collection of all power objects. Note that the distance function h is not smooth at $x \in \mathbb{R}^3$ if and only if x is contained in a power object of dimension less than three.

The dual of the power diagram is called weighted Delaunay diagram. For convenience we want to refer to weighted Delaunay diagrams just as Delaunay diagrams in the following. The Delaunay diagram of P is a cell complex that decomposes the convex hull of the points in P. The convex hull of four or more points in P defines a *Delaunay cell* if the intersection of the corresponding power cells is not empty and there exists no superset of points with the same property. Analogously, the convex hull of three or two points in P defines a *Delaunay face* or *Delaunay edge*, respectively, if the intersection of their corresponding power cells is not empty. Up to degenerate situations every point in P is a *Delaunay vertex*. The term *Delaunay object* can denote either a Delaunay cell, face, edge or vertex.

The definition of Delaunay diagrams provides us with a duality between power- and Delaunay objects. That is, for every d-dimensional power object, $0 \leq d \leq 3$, there is a dual $(3 - d)$-dimensional Delaunay object and vice versa. For more information see Aurenhammer [1]. Using this duality we can characterize the critical points of the distance function h. It is easy to see that a local minimum of h is a point in P that is contained in its own power cell. We should note here that not necessarily ever point in P is contained in its own power cell. If x is either a saddle point or a local maximum then $|A(x)| > 1$, i.e. x is contained in a power object of dimension less than three, and $H(x)$ is a Delaunay object. In fact, we have the following theorem.

Theorem 1. *The critical points of h are the intersection points of power objects and their dual Delaunay objects.* □

So far the critical point theory of distance functions allowed us to characterize the critical points of h. Now we want to find an analog of the gradient vector field for the distance function h. Starting point is the observation that at any point the gradient vector of a smooth function points in the direction of steepest ascent of the function. As a replacement for the gradient vector field we want to assign to every regular point of h the unit vector that points in the direction of steepest ascent of h. To the critical points we assign the zero vector.

The direction of steepest ascent of h at every point $x \in \mathbb{R}^3$ can be characterized in terms of the *driver* $d(x)$ of x.

Driver. For any point $x \in \mathbb{R}^3$ let $d(x)$ be the point in $H(x)$ closest to x. We call $d(x)$ the driver of x.

The driver of a point x will be also very important for our algorithms where we are going to make use of the following more explicit characterization that allows to determine drivers efficiently.

Lemma 1. *For a point $x \in \mathbb{R}^3$ let V be the lowest dimensional power object that contains x and let D be the dual Delaunay object of V. The driver of x is the point in D closest to x. Furthermore, all points in the interior of a power object have the same driver.* □

But most importantly knowing the driver of a point x allows to compute the direction of steepest ascent of h at x.

Lemma 2. *For any regular point $x \in \mathbb{R}^3$ let $d(x)$ be the driver of x. The steepest ascent of the distance function h at x is in the direction of $x - d(x)$.* □

The unit vector field $v : \mathbb{R}^3 \to \mathbb{R}^3$ assigns to every point in $x \in \mathbb{R}^3$ the direction of the steepest ascent of h at x, i.e.

$$v(x) = \frac{x - d(x)}{\|x - d(x)\|} \text{ if } x \neq d(x) \text{ and } 0 \text{ otherwise.}$$

This vector field leads us to the main topic of our study, the *flow* associated with the vector field v. The flow associated with v is a function

$$\phi : [0, \infty) \times \mathbb{R}^3 \to \mathbb{R}^3$$

such that its right derivative at every point $x \in \mathbb{R}^3$ satisfies the following equation

$$\lim_{t \downarrow t_0} \frac{\phi(t, x) - \phi(t_0, x)}{t - t_0} = v(\phi(t_0, x)).$$

We often refer to the first argument of the flow as time.

An important notion associated with a flow is that of a *fixpoint*. A point $x \in \mathbb{R}^3$ is called a fixpoint of ϕ if $\phi_x(t) = x$ for all $t \geq 0$. It is not difficult to see that the fixpoints of ϕ are exactly the critical points of the distance function h. Because of this observation we want to refer to a fixpoint of ϕ as a minimum, saddle or maximum if the corresponding critical point of the height function is a minimum, saddle or maximum, respectively.

With the notion of a fixpoint at hand we can also give an explicit description of the flow ϕ. For all fixpoints x of ϕ we have of course

$$\phi(t, x) = x \ , \ t \in [0, \infty).$$

Otherwise let $d(x)$ be the driver of x and R be the ray originating at x and shooting in the direction $v(x)$. Let z be the first point on R whose driver is different from $d(x)$. Note that such a z need not exist in $\mathbb{R}^3$ if x is contained in an unbounded power object. In this case let z be the point at infinity in the direction of R. We set:

$$\phi(t, x) = x + t \cdot v(x) \ , \ t \in [0, \|z - x\|)$$

For $t \geq \|z - x\|$ the flow is given as follows:

$$\begin{aligned} \phi(t, x) &= \phi\left(t - \|z - x\| + \|z - x\|, \, x\right) \\ &= \phi\left(t - \|z - x\|, \, \phi\left(\|z - x\|, \, x\right)\right) \end{aligned}$$

If we fix the second argument of the flow ϕ to be some point $x \in \mathbb{R}^3$ then we get the *orbit* or *flow line* of x, i.e. a function

$$\phi_x : [0, \infty) \to \mathbb{R}^3 \ , \ t \mapsto \phi(t, x).$$

The orbit describes the motion of x under the flow ϕ. One can show that the orbits of regular points are piecewise linear curves.

The flow line of a regular point either connects it with some fixpoint of ϕ or it leaves any compact subset of $\mathbb{R}^3$ in finite time. For every fixpoint x of ϕ we collect all points in $\mathbb{R}^3$ that flow into x and call the resulting set $S(x)$ the *stable manifold* of x, i.e.

$$S(x) = \{y \in \mathbb{R}^3 \ : \ \lim_{t \to \infty} \phi_y(t) = x\}.$$

Up to degeneracies the stable manifolds of the fixpoints of ϕ build a three dimensional cell complex. We call this cell complex the *weighted flow complex*.

3 The cells of the weighted flow complex

In this section we have a closer look on the cells of the weighted flow complex. These cells are the stable manifolds of fixpoints of the flow derived from the unit vector field induced by a set of weighted points. It turns out that the index of a fixpoint, i.e. the index of the corresponding critical point of the distance function, gives the dimension of its stable manifold.

0-cells

From Theorem 1 we know that the local minima of the flow are weighted points that are contained in their dual power cell. In contrast to unweighted case not every weighted point is contained in its dual power cell. It can even happen that the dual power cell of a weighted point is empty. But it is algorithmically easy to check if a weighted point is contained in its dual power cell provided one has already computed the power- and the Delaunay diagram.

As one expects from a local minimum x no point besides x itself flows into it. That is, the stable manifold of x contains just x itself. The stable manifolds of the local minima are the 0-cells of the weighted flow complex. Thus the 0-cells of the weighted flow complex are a subset of the set of weighted points.

1-cells

Up to degeneracies the stable manifolds of the index 1 saddle points are the 1-cells of the flow complex. In fact, we can prove the following lemma.

Lemma 3. *Let x be an index 1 saddle of ϕ. If the stable manifold $S(x)$ of x does not contain a point on a power edge then the closure of $S(x)$ is a simple piecewise linear curve whose endpoints are local minima of ϕ.* □

Note that the situations where $S(x)$ does contain a point on a power edge are really degenerate in the sense that such a situation is not stable under small perturbations of the weighted points that determine ϕ.

Probably the easiest way to describe the stable manifold of an index 1 saddle point is to give an algorithm to compute it. For the description of the algorithm we assume that the power- and the Delaunay diagram of P have already been computed such that we can query them.

ONECELL(Index-1-saddle x)
1 $V, E := \emptyset$
2 $vw :=$ Delaunay edge that contains x.
3 $F := \{xv, xw\}$
4 **while** $F \neq \emptyset$ **do**
5 choose $yd \in F$; $F := F - \{yd\}$.
6 $y' :=$ first intersection point of the segment from y to d with a power facet f that intersects yd in only one point; or d if such a point does not exist.
7 $V := V \cup \{y'\}$; $E := E \cup \{yy'\}$
8 **if** $y' \neq d$ **do**
9 $dd' :=$ Delaunay edge dual to f.
10 $F := F \cup \{y'd'\}$
11 **end if**
12 **end while**

The algorithm ONECELL stores the vertices of the piecewise linear curve which is the stable manifold of the index 1 saddle point x in a set V. The edges of this curve are stored in a set E. In line 1 both sets sets are initialized with the empty set. From Theorem 1 we know that the index 1 saddle point x is the intersection point of a Delaunay edge vw with its dual power facet. We compute this Delaunay edge in line 2 and initialize in line 3 a set F with the two line segments xv and xw that connect x with the endpoints of the Delaunay edge. We will always maintain the invariant that the second endpoint of a line segment stored in F is a Delaunay vertex. Lines 4 and 12 enclose the main loop of the algorithm. While the set F is not empty we choose in line 5 a line segment $yd \in F$ and remove it from F. We know from Lemma 1 that the driver along a line segment can only change at the boundary of a power object. In line 6 we determine the point y' where the driver along the line segment yd might change. In line 7 we include this point in our vertex set V and the edge yy' in the edge set E. If y' equals d then we have reached a local minimum of the flow. That is, the stable manifold of x ends here. Otherwise we have to determine the driver of the point y'. This driver is the endpoint d' of the Delaunay edge dd' dual to the power facet f. In line 10 we include the line segment $y'd'$ into F for further processing.

Note that the algorithm ONECELL does not treat the degenerate case. We have shown in [8] for flow diagrams in two dimensions how to deal explicitly with degenerate situations. In our implementation we use explicit perturbations.

The description of the algorithm ONECELL shows that the stable manifold of an index 1 saddle point in general is a polygonal arc with many vertices. That is in contrast to the unweighted case where the stable manifolds of index 1 saddles are exactly the *Gabriel* edges, i.e. straight line segments. In fact, one can show that the stable manifold of an index 1 saddle point can have up to $O(|P|)$ vertices.

2-cells

For the 2-cells of the flow complex we have a lemma very similar to Lemma 3. This lemma states that up to degeneracies the stable manifolds of the index 2 saddle points are the 2-cells of the flow complex.

Lemma 4. *Let x be an index 2 saddle of ϕ. If the stable manifold $S(x)$ of x does not contain a power*

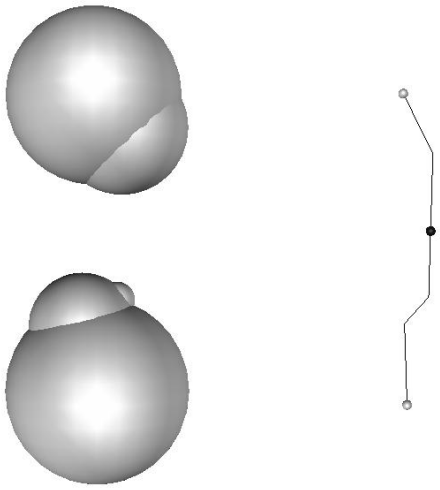


Fig. 1. On the left a set of positively weighted points in $\mathbb{R}^3$ shown as balls. On the right: The two local minima and the saddle point of the point set from the left.

vertex then the closure of $S(x)$ is a piecewise linear surface with boundary. The boundary of this surface is made up from 1-cells. □

As for the 1-cells the degenerate situation that $S(x)$ does contain a power vertex is not stable under small perturbations of the point set P.

Again we want to turn to an algorithm to describe the 2-cells explicitly. We assume that the power- and the Delaunay diagram of P have already been computed.

```
TwoCell( Index-2-saddle x )
1  F := ∅
2  InFlowEdge(x)
3  return F
```

The algorithm TwoCell initializes in line 1 a set F that is going to store all the triangles that will make up the stable manifold $S(x)$ of the index 2 saddle x. The set F is a global variable that can also be accessed by the subroutines InFlowEdge and InFlowFacet. In line 2 the algorithm just calls a subroutine InFlowEdge which we are going to describe next.

```
InFlowEdge( Point-on-a-power-edge v )
1  f := Delaunay facet dual to the power edge
        that contains v.
2  for each Delaunay edge dd' incident to f
        whose endpoints lie on different
        sides of its dual power facet g do
3     s := intersection of the triangle vdd' with
           the power facet g.
4     w := endpoint of s different from v.
5     InFlowFacet(v, w, d)
6     InFlowFacet(v, w, d')
7     if w is not a saddle of index 1 do
8        InFlowEdge(w)
9     end if
10 end for
```

The subroutine InFlowEdge recursively calls (and is called by) another subroutine InFlowFacet which reads in pseudocode as follows.

```
InFlowFacet( Point v, Point v', Driver d )
1  c := power cell dual to d.
2  p := intersection of c with triangle vv'd
3  F := F ∪ {p}
4  if d is not a local minimum do
5     for each power facet f incident to c that
           is intersected by the triangle
           vv'd in a line segment
           ww' ≠ vv' do
6        dd' := Delaunay edge dual to f.
7        InFlowFacet( (w, w', d') )
8     end for
9     for each power edge e incident to c that
           is intersected by the triangle
           vv'd in a point w ≠ v, v' do
10       if w is not a saddle of index 1 do
11          InFlowEdge(w)
12       end if
13    end for
14 end if
```

We are now going to describe the subroutines InFlowEdge and InFlowFacet in detail.

The subroutine InFlowEdge computes the set of points that flow into a point v which has to be contained in a power edge. See Figure 2 for an illustration.

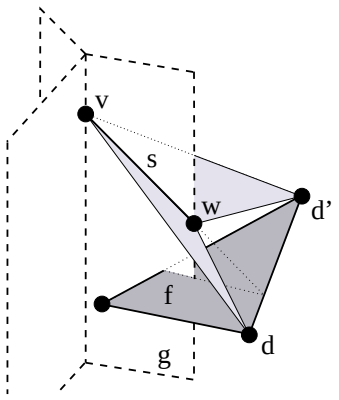


Fig. 2. Geometric objects that occur in the definition of the subroutine InFlowEdge

All the drivers of the points that flow into v are contained in the Delaunay facet f dual to the power edge that contains v. In line 1 we compute this Delaunay facet. Only a Delaunay edge dd' incident to f whose endpoints d and d' lie on different sides of its dual power facet g can yield a two dimensional flow into v. That is, the edge dd' contains a driver that drives the points on a certain line segment s into v, but all the points that flow into this line segment s eventually also flow into v. The line segment s is the intersection of the triangle vdd' with the power facet g. We compute the line segment s in line 3 inside the loop enclosed by the lines 2 and 10 which iterates over all such Delaunay edges dd'. Let w be the second endpoint of s besides v. That is, w is the point on s farthest away from v. We compute w in line 4. The flow into the line segment s is either driven by d or d' or it flows into s via w. The first case is handled by the recursive calls of the subroutine InFlowFacet in lines 5 and 6. The second case is handled by the recursive call of InFlowEdge in line 8 provided w is not an index 1 saddle point. In the latter case we have already reached the boundary of the stable manifold $S(x)$ of the index 2 saddle point x.

The subroutine InFlowEdge computes the set of points that are driven by a Delaunay vertex s into a line segment vv'. The line segment vv' has to be contained in a power facet. See Figure 3 for an illustration.

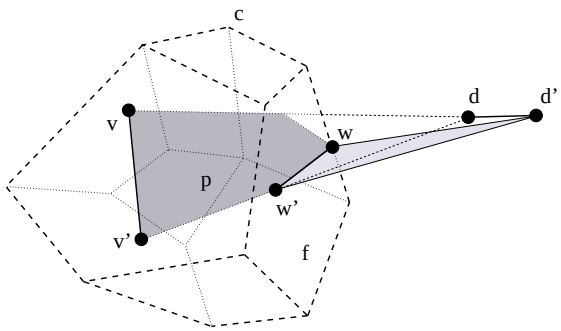


Fig. 3. Geometric objects that occur in the definition of the subroutine InFlowFacet

Let c be the power cell dual to d. All the points in the intersection of c with the triangle $vv'd$ flow into the segment vv'. In line 1 we determine the power cell c and in line 2 its intersection p with the triangle $vv'd$. We add p or a triangulation of it to F in line 3. If d is a local minimum of the flow then p is exactly the triangle $vv'd$ and no other points flow into vv'. Otherwise more points flow into vv'. These points can only flow into vv' through the intersection of the boundary of c with the triangle $vv'd$. This intersection can be decomposed into a collection of line segments on power facets and points on power edges. Since we assume non-degeneracy the triangle $vv'd$ cannot intersect the boundary of c in a power vertex. As for the computation of the 1-cells in our implementation we deal with degeneracies by explicitly perturbing the point set P. The line segments are the intersections of the triangle $vv'd$ with power facets f in the boundary of the power cell c. We take care of these line segments ww' in the loop enclosed by the lines 5 and 8 by calling InFlowFacet recursively with input (w, w', d') in line 7. To do so we have determined d' in line 6 as the second endpoint of the Delaunay edge dd' of a power facet f incident to the power cell c. In the loop enclosed by the lines 9 and 11 we take care of the points w in the intersection of the triangle $vv'd$ with power edges in the boundary of the power cell c. If w is not an index 1 saddle point we call the InFlowEdge recursively with w as its argument. Otherwise, i.e. w is an index 1 saddle point, we have already reached the boundary of the stable manifold $S(x)$ of the index 2 saddle point x.

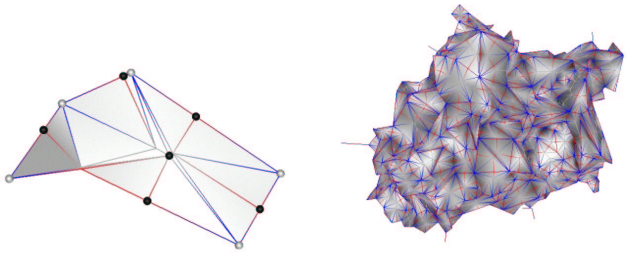


Fig. 4. On the left: A 2-cell of the weighted flow complex. On the right: The 2-skeleton of a weighted flow complex

3-cells

We are not going to compute the 3-cells of the flow complex explicitly but only indirectly through the simplicial complex made up from all 0-, 1- and 2-cells. Lets call this complex the *2-skeleton* of the flow complex. With similar techniques as in the unweighted case one can prove the following theorem.

Theorem 2. *In case that there are no degeneracies the stable manifolds of the local maxima of a*

flow ϕ are exactly the bounded regions of the 2-skeleton of the flow complex. □

That is, the 3-cells of the flow complex are exactly the stable manifolds of the local maxima of the flow ϕ.

4 Applications in bio-geometric modeling

In this section we are going to demonstrate how the weighted flow complex can be applied in modeling certain properties of macromolecules. To do so we need at first a geometric model of the molecule at hand. A natural model is a union of balls in $\mathbb{R}^3$ where each ball in the union represents an atom of the molecule. Such models are called space filling models [5, 12]. A ball is characterized by a pair $(z, r) \in \mathbb{R}^3 \times [0, \infty)$, where $z \in \mathbb{R}^3$ denotes the center of the ball and r denotes its radius. That is, a ball can be seen as a weighted point with positive weight. In space filling models the balls are centered at the locations of the corresponding atoms. Usually one gets these locations from X-ray diffraction of the crystallized protein. The two most popular space filling models differ in the radii they assign to the balls.

(1) In the *van der Waals model* the radius of a ball is the van der Waals radius of the corresponding atom.
(2) In the *solvent accessible model* the radius of a ball is the van der Waals radius of the corresponding atom plus the radius of some solvent molecule also modeled as a ball. That is, the balls in the solvent accessible model are always larger than the corresponding balls in the van der Waals model. The solvent is frequently taken to be a water molecule, modeled as a ball with radius radius 1.4 Å (Angstrom).

We want to discuss two applications. The first application aims for decomposing a molecule. Macromolecules are often a collection of proteins that are only loosely coupled. We want to use the *1-skeleton* of the weighted flow complex to decompose a macromolecule in these constituents. In the second application we are going to rephrase the concept of pockets in macromolecules as developed by Edelsbrunner et al. [6] in terms of the *2-skeleton* of the weighted flow complex.

In the following we assume that we are given a space filling model of macromolecule as input as a set P of positively weighted points. Our approach can handle both types of space filling models as input, but the solvent accessible

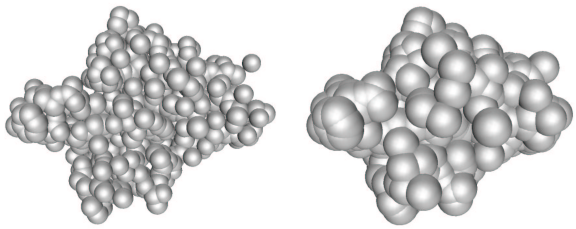


Fig. 5. Space filling models of some Neurotoxin protein. On the the left: Van der Waals model. On the right: Solvent accessible model.

seems to be more reasonable from a chemical perspective. The points in P model the atoms of the macromolecule and their weights the radii of these atoms. We say that a point x is contained in a space filling model if x is contained in the union of balls that correspond to the points in P. We derive both the distance function h and the flow ϕ from the point set P.

Decomposing macromolecules

When we try to decompose a macromolecule in its constituents we essentially try to predict the bonds between the atoms of the molecule. There should not exist bonds between the different constituents of the macromolecule. Bader [2] has developed a theory based on the gradient vector field of the charge density to predict bonds. He writes in his book,

> The topology of ρ, the charge density, as displayed in the global properties of its gradient vector field, yields a faithful mapping of the chemical concepts of atoms, bonds, and structure.

The global properties mentioned in this citation are for example the critical points of the charge density. Bonds in this theory are represented as index 2 saddle points. If we assume that the distance function h approximates $(-\rho)$, i.e. the negative of the charge density, to some extent then we should be able to derive an approximation of the bond structure from the index 1 saddle points of h.

The collection of all 1-cells of the weighted flow complex, i.e. the collection of the stable manifolds of the index 1 saddle points, forms the 1-skeleton of the flow complex. One can show that the 1-skeleton is connected. That is, we cannot decompose a macromolecule by directly using the 1-skeleton of the flow complex induced by P.

Instead we are going to use a *filtration* of the 1-skeleton.

The distance function h of course also takes values at the index 1 saddle points. We assign these values to the corresponding 1-cells of the weighted flow complex. Let S_1^α be the subset of the 1-skeleton that contains all 1-cells whose assigned value is smaller than or equal to α. Since we have

$$S_1^\alpha \subseteq S_1^\beta \text{ if } \alpha \leq \beta,$$

these sets gives us a filtration of the 1-skeleton.

We found that setting α to -8 Å (Angstrom) decomposes many macromolecules in its constituents if the input was a solvent accessible model with solvent radius 1.4 Å. See Figure 6 for an example. If the input was a van der Waals model a value of -1.2 Å works well.

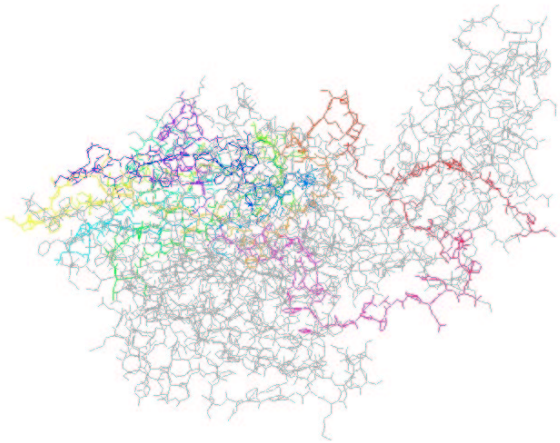


Fig. 6. Decomposition of a Rhinovirus into constituents. One constituent is highlighted.

Pockets in macromolecules

As we mentioned earlier pockets were introduced in [6] to model essential cavities in macromolecules. Their definition is based on a space filling model of the molecule. Again let P be the set of positively weighted points that model the atoms in the space filling model. We define a *void* as a compact connected region in the complement of the space filling model inside the convex hull of P, i.e. compact regions in the convex hull of P not occupied by the balls that model the atoms. If we let the weights of the points in P grow new voids are created and existing ones get destroyed. Finally all voids get destroyed. In [6] the following growth model for the weight r of a point (z, r) was chosen:

$$r(t) = \sqrt{r^2 + t} \text{ for all } t \geq 0$$

The weights are now a function of time. See Figure 7 for two snapshots of such a growth process for a set of weighted points in the plane. Note that the growth process does not change the power- and the Delaunay diagram.

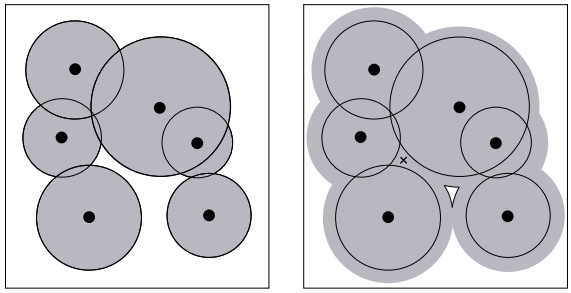


Fig. 7. On the left: A space filling model of a molecule made up from six atoms in two dimensions. The atoms of this molecule do not define a void. On the right: If we grow the disks a void emerges which gets destroyed if we grow the disks further. The point at x is a maximum, i.e. at x another void which was created earlier has already been destroyed.

While the weights grow a void shrinks until it finally consists of only a single point before it vanishes. For reasons that will become obvious later we call these points *positive maxima*. We group positive maxima together if they can be connected by a path in the complement of the space filling model inside the convex hull of P, i.e. in the intersection of the convex hull of P with the complement of the space occupied by the balls that model the atoms. Essentially, these groups of positive maxima define a pocket. Edelsbrunner et al. assign a shape to a pocket. Our approach to assign a shape to a pocket is based on the weighted flow complex induced by P. It is closely related to the definition of voids and positive maxima as we have presented them here, but puts these definitions in the context of a more general theory.

We partition the set of critical points of the flow induced by the balls in P into two sets.

We partition the set of critical points of the flow induced by the balls in P into two sets. The first set contains all critical points that are contained in the space filling model of the molecule. The critical points in the second set are all critical points not contained in the first set, i.e. they re-

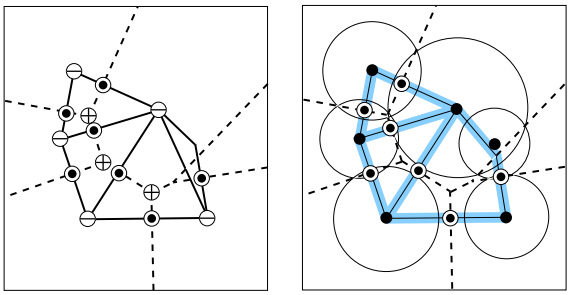


Fig. 8. On the left: The critical points of the flow induced by the points from Figure 7. The power diagram of the points is shown with dashed lines and the Delaunay diagram with solid lines. The local maxima are denoted as $\oplus$, the saddles as $\odot$ and the local minima as $\ominus$. Note that two of the maxima are at positions where voids get destroyed if we grow the disks. On the right: The 1-skeleton of the corresponding weighted flow complex.

side outside the space filling model. From the definition of the distance function h it follows that h takes non-positive values on the first set of critical points and positive values on the second set. That is why we call the critical points from the first set *negative* and the points from the second set *positive*. The positive maxima that we used when we introduced pockets are exactly positive maxima as defined here. For a formal definition of pockets we introduce the *restricted flow complex* as the subcomplex of the flow complex that contains all cells that correspond to positive critical points. See Figure 9 for an example.

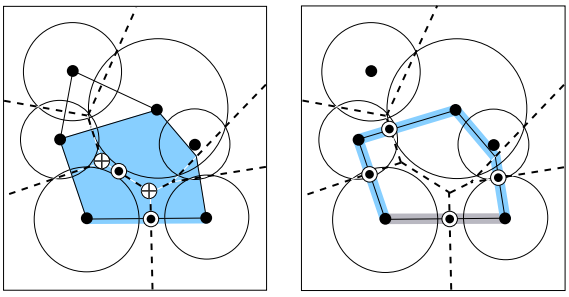


Fig. 9. On the left: An example of a restricted flow complex. This complex contains the stable manifolds of two saddle points $\odot$ and two local maxima $\oplus$. It defines one pocket. At both maxima voids get destroyed if we grow the disks. On the right: The boundary (light blue) of the pocket on the left contains the stable manifolds of three saddles $\odot$. The mouth (gray) of the same pocket is the stable manifold of one saddle.

From the restricted flow complex we directly derive our definition of a pocket.

Pocket. A pocket K is a maximal connected component of the restricted flow complex, i.e. the component is connected and there is no larger connected component in the restricted flow complex that contains it. Thus a pocket is a collection of cells corresponding to some subset of positive critical points.

To visualize pockets we make use of the cell structure of the weighted flow complex. We call a negative critical point a *boundary critical point* of a pocket K if its corresponding cell in the weighted flow complex is contained in the boundary of a cell that corresponds to a positive critical point in K. Note that the cells that correspond to boundary critical points all have to be contained in the *2-skeleton* of the weighted flow complex. Instead of visualizing a pocket directly we will visualize its boundary, i.e. the complex made up from the cells corresponding to its boundary critical points. See Figure 9 for an example.

One can distinguish three types of pockets according to their number of openings to the outside of the protein. The outside is defined as the stable manifold of a virtual maximum at infinity, i.e. the closure of the set of all points that flow to infinity under the flow ϕ induced by P. We call a positive critical point an *opening critical point* if its stable manifold is contained in the boundary of the stable manifold of the maximum at infinity. We call a maximal connected component in the subcomplex of the restricted flow complex that is build by the cells corresponding to the opening critical points a *mouth*. We use the number of mouths of a pocket to classify pockets. A pocket is called,

(1) a *void*, if it is not incident to any mouth. A void is an unaccessible cavity, i.e. solvent molecules from outside the protein cannot enter a void. Voids sometimes have a stabilizing functionality for a protein.
(2) a *normal pocket*, if it is incident to exactly one mouth.
(3) a *tunnel*, if it is incident to more than one mouth. Proteins who function as an ion pump typically contain a large tunnel.

5 Implementation and results

We implemented the algorithms OneCell and TwoCell using C++ and the Computational Geometry Algorithms Library *CGAL* [4] which provides fast and robust weighted Delaunay triangulations in three dimensions through its regular triangulation package.

We tested our implementations on protein datasets that we retrieved from the Protein Data bank [3]. All tests were performed on a 480 Mhz Sun Ultra Sparc II. Screenshots of some of the computed results can be found in Figures 6 and 10. We used *Geomview* [7] for the rendering of the models. All data were computed from solvent accessible models of the molecules.

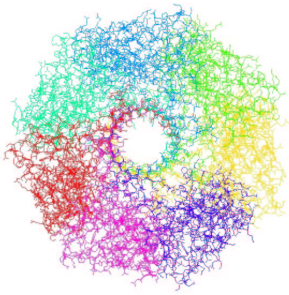
In Table 1 we summarize additional data of the shown molecules. It is interesting to note that in our experiments the runtime per atom is almost constant, i.e. here it seems to be independent of the total number of atoms in the model.

Protein	pdb key	Atoms	Time	ms/Atoms
NEUROTOXIN	1nxb	543	0.8	1.47
GRAMICIDIN	1alz	850	1.7	2.00
OXYGEN	1mbd	1666	2.8	1.68
1XTHROMBIN	1ppb	2819	4.9	1.74
RHINOVIRUS	4rhv	6542	11.6	1.77
2XTHROMBIN	1aho	6901	12.0	1.74
RECEPTOR	a7gg	8580	15.2	1.77
ALPHA-TOXIN	7ahl	22778	41.5	1.82
NITROGENASE	1n2c	24432	43.5	1.78

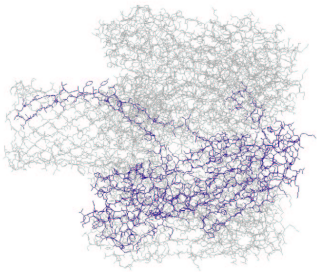
Table 1. Basic data for several molecules. With the pdb key molecule data can be retrieved from the Protein Data bank [3]. The timings are given in seconds and the processing rate is given in milliseconds per atom.

References

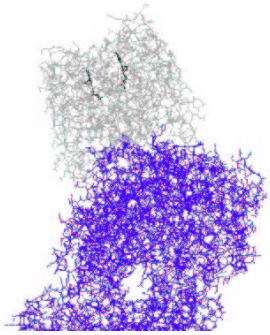
1. F. Aurenhammer and R. Klein. Voronoi Diagrams. In *Handbook of Computational Geometry*, J.-R. Sack and J. Urrutia (eds.), pp. 201–290, Elsevier (2000)
2. R.F.W. Bader. Atoms in molecules: a quantum theory. Clarendon Press (1990)
3. H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov and P.E. Bourne. The Protein Data Bank. *Nucleic Acids Res.*, **28**, pp. 235–242 (2000) `http://www.rcsb.org/pdb/`
4. Computational Geometry Algorithms Library `http://www.cgal.org`
5. T.E. Creighton. *Proteins: Structures and Molecular Properties.* Second Edition, Freeman, New York, (1993)
6. H. Edelsbrunner, M. A. Facello and J. Liang. On the definition and the construction of pockets in macromolecules. *Discrete Apl. Math.* **88**, pp. 83–102, (1998)
7. `http://www.geomview.org`
8. J. Giesen and M. John. A new diagram from disks in the plane. In *Proc. 19th Symp. Theoretical Aspects of Computer Science*, pp. 238–249, (2002)
9. J. Giesen and M. John. Surface reconstruction based on a dynamical system. *Computer Graphics Forum* **21**(3), pp. 363–371, (2002)
10. J. Giesen and M. John. The flow complex: A data Structure for Geometric Modeling. In *Proc. 14th ACM-SIAM Symp. Discrete Algorithms*, pp. 285–294, (2003)
11. K. Grove. Critical Point Theory for Distance Functions. In *Proceedings of Symposia in Pure Mathematics* **54**(3), pp. 357–385, (1993)
12. F.M. Richards. Areas, volumes, packing and protein structures. *Ann. Rev. Biophys. Bioeng.*, **6**, pp. 151–176, (1977)



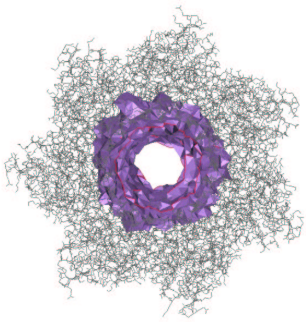
(a) A decomposition of Alpha-toxin. Different protein chains are colored differently.



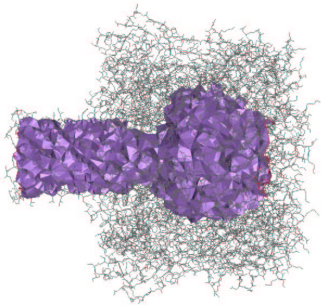
(b) A sideview of (a). Only one protein chain is emphasized.



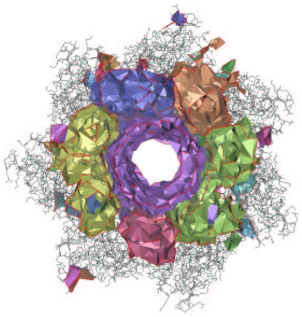
(c) A zoom into the decomposition of Nitrogenase. On the top, two nicely separated ATP molecules.



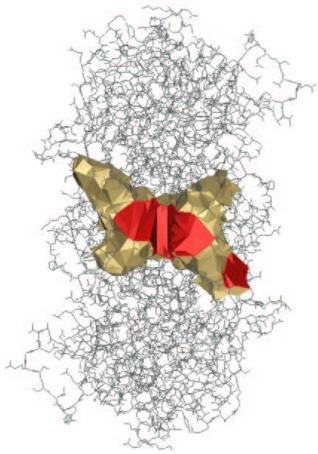
(d) A tunnel in Alpha-toxin. No mouths are shown.



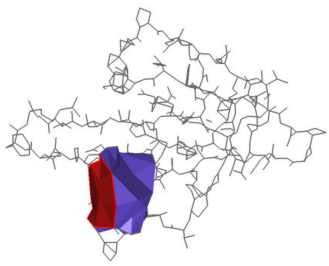
(e) A sideview of the tunnel shown in (d).



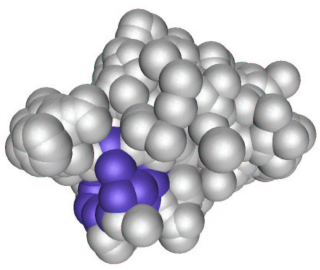
(f) The tunnel shown in (d) and supporting normal pockets. No mouths are shown.



(g) Two Thrombin proteins bind at their active sites which gives rise to a tunnel. The three mouths of the tunnel are emphasized.



(h) A normal pocket in Neurotoxin. The mouth is emphasized.



(i) A space filling model of Neurotoxin. Atoms at the normal pocket shown in (h) are highlighted.

Fig. 10. Decompositions and pockets of some macromolecules