Tessellation of Quadratic Elements

Scott E. Dillard[1,3], Vijay Natarajan[1,3], Gunther H. Weber[1,3],
Valerio Pascucci[2,3], Bernd Hamann[1,3]

[1] Department of Computer Science, University of California, Davis
[2] Lawrence Livermore National Laboratory
[3] Institute for Data Analysis and Visualization

Abstract. Topology-based methods have been successfully used for the analysis and visualization of piecewise-linear functions defined on triangle meshes. This paper describes a mechanism for extending these methods to piecewise-quadratic functions defined on triangulations of surfaces. Each triangular patch is tessellated into monotone regions, so that existing algorithms for computing topological representations of piecewise-linear functions may be applied directly to piecewise-quadratic functions. In particular, the tessellation is used for computing the Reeb graph, which provides a succinct representation of level sets of the function.

1 Introduction

Scalar functions often represent physical quantities like temperature, pressure, etc. Scientists interested in understanding the local and global behavior of these functions study their level sets. A *level set* of a function f consists of all points $f^{-1}(c)$ where the function value is equal to a constant c. Various methods have been developed for the purpose of analyzing the topology of level sets of a scalar function. These methods primarily apply to piecewise linear functions. We discuss an extension of these methods to bivariate piecewise quadratic functions defined over a triangulated surface.

A *contour* is a single connected component of a level set. Level sets of a smooth bivariate function are simple curves. The *Reeb graph* of f is obtained by contracting each contour to a single point [1], see Fig. 1. The connectivity of level sets changes at *critical points* of a function. For smooth functions, the critical points occur where the gradient becomes zero. Critical points of f are the *nodes* of the Reeb graph, connected by *arcs* that represent a family of topologically equivalent contours.

1.1 Related Work

Methods to extract contours from bivariate quadratic functions have been explored in the context of geometric modeling applications [2]. *A priori* determination of the topology of contours has been studied also in the computer graphics and visualization [3, 4]. Much research has focused on functions defined by bilinear and trilinear interpolation of discrete data given on a rectilinear

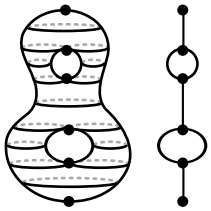

Fig. 1. Reeb graph of a height function defined over a double torus. Critical points of the surface become nodes of the Reeb graph.

grid. Work in this area has led to efficient algorithms for computing the contour tree, a special Reeb graph that has no cycles [5, 6]. More general algorithms have been developed for computing Reeb graphs and contour trees for piecewise-linear functions [7–9].

Topological methods were first used in computer graphics and scientific visualization as a user interface element, to describe high-level topological properties of a dataset [10]. They are also used to selectively explore large scientific datasets by identifying important function values related to topological changes in a function [11, 12], and for selective visualization [13]. Reeb graphs have also been used as the basis for searching large databases of shapes [14], and for computing surface parametrizations of three-dimensional models [15].

1.2 Results

Given a triangulated surface and a piecewise-quadratic function defined on it, we tessellate the surface into monotone regions. A graph representing these monotone regions is a valid input for existing algorithms that compute Reeb graphs and contour trees for piecewise-linear functions. The essential property we capture in this tessellation is that the Reeb graph of the function restricted to a single tile of the tessellation is a straight line. In other words, every contour contained in that tile intersects the triangle boundary at least once and at most twice. We tessellate each triangular patch by identifying critical points of the function within the triangle and connecting them by arcs to guarantee the required property.

2 Background

We consider bivariate, piecewise-quadratic functions defined over triangular meshes. Bivariate quadratics are functions $f : \mathbb{R}^2 \to \mathbb{R}$ of the form

$$f(x, y) = Ax^2 + Bxy + Cy^2 + Dx + Ey + F.$$

A critical point of f is a point $\mathbf{x}$ where $\nabla f(\mathbf{x}) = \mathbf{0}$. The partial derivatives are given by the following linear expressions

$$\frac{\partial f}{\partial x} = 2Ax + By + D \quad \text{and} \quad \frac{\partial f}{\partial y} = 2Cy + Bx + E.$$

The location of a critical point $(\hat{x}, \hat{y})$ is given by

$$\hat{x} = \frac{-2CD + BE}{4AC - B^2} \quad \text{and} \quad \hat{y} = \frac{-2AE + BD}{4AC - B^2}.$$

Bivariate quadratics and their critical points can be classified based on their contours. The contours are conic sections. Let $H = 4AC - B^2$ be the determinant of the Hessian matrix of f. We partition the set of bivariate quadratic functions into three classes:

1. $H > 0$: Contours are ellipses; The critical point is maximum or minimum.
2. $H < 0$: Contours are hyperbolas; The critical point is a saddle.
3. $H = 0$: Contours are parabolas, pairs of parallel lines, or single lines; No critical point exists.

We refer to members of these classes as *elliptic*, *hyperbolic*, and *parabolic*, respectively. We further classify the critical points of elliptic functions using the second-derivative test. The critical point is a minimum when $A > 0$, and a maximum when $A < 0$. When $A = 0$, the sign of C discriminates between maxima and minima.

2.1 Line Restrictions

Let $\ell(t) = \begin{pmatrix} x_0 + tx_d \\ y_0 + ty_d \end{pmatrix}$ be a parametrically defined line passing through $(x_0, y_0)^T$ in direction $(x_d, y_d)^T$. Now restrict the domain of the bivariate quadratic $f(x, y)$ to only those points on $\ell(t)$. We then have a univariate quadratic $r(t) = \alpha t^2 + \beta t + \gamma$, where

$$\begin{aligned} \alpha &= Ax_d^2 + Bx_d y_d + Cy_d^2, \\ \beta &= 2Ax_0 x_d + B(y_0 x_d + x_0 y_d) + 2Cy_0 y_d + Dx_d + Ey_d, \text{ and} \\ \gamma &= Ax_0^2 + Bx_0 y_0 + Cy_0^2 + Dx_0 + Ey_0 + F. \end{aligned}$$

We call $r(t)$ a *line restriction* of f. If $\alpha \neq 0$, r is a parabola with one critical point at $\hat{t} = -\beta/2\alpha$; if $\alpha = 0$, r is linear. We refer to a critical point of this univariate quadratic function as a *line-critical point*, while we refer to critical points of the bivariate function as *face-critical points.* The line restrictions have several useful properties:

1. There is at most one line-critical point on a line restriction, since the function along the line is either quadratic or linear.

2. If a line intersects any contour twice, it must contain a line-critical point between these intersections: Assume that the function value on the contour is zero. The sign of the line-restriction changes each time it crosses the contour. Applying the mean value theorem to the derivative of the line-restriction, there must be a critical point between two zero crossings.
3. A line-critical point is located at the point where the line is tangent to a contour, following from the previous property.
4. Any line restriction passing through a face-critical point has a line-critical point which is coincident with the face-critical point: The gradient at the face critical point is zero. Thus, all directional derivatives are zero, and, in particular, the derivative of the line restriction is zero.

2.2 Contours and Critical Points

A critical point is a point where the number of contours or the connectivity of existing contours changes. When the gradient is not defined, we may classify critical points based on the behavior of the function in a local neighborhood [16]. Figure 2 shows this classification. Consider a plane of constant height passing through the graph surface $(x, y, f(x, y))$ of f. The intersection of the surface and the plane is a set of contours, each homeomorphic to either a closed loop or a line segment. When this plane passes a minimum, a new contour is created. When the surface passes a maximum, an existing contour is destroyed. When the surface passes a saddle, two types of events occur: (a) Two segments may merge into a new segment or a segment may split into two. (b) The endpoints of a segment may connect with each other to form a loop or a loop may split into a segment.

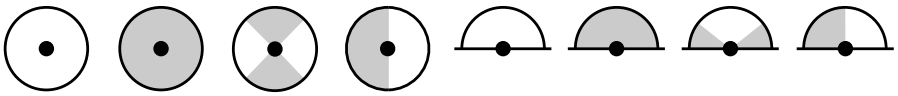

Fig. 2. Interior minimum, maximum, saddle and regular point, and boundary minimum, maximum, saddle and regular point. Shaded areas are regions with function value less than the indicated point.

We consider critical points of a function restricted to a triangular patch. A face criticality of an elliptic function creates or destroys a loop when the sweep plane passes it. A face criticality of a hyperbolic function interchanges the connectivity of two segments. A line criticality can create or destroy a segment, merge two segments into a new segment or split a segment in two, transform a segment into a loop or a loop into a segment. However, a line-critical point cannot create or destroy loops. We determine whether a line criticality is an extremum or a saddle by examining the directional derivative perpendicular to the edge, at the critical point. Vertices, when not located exactly at a hyperbolic saddle point, can only create or destroy segments within a triangular patch. (See Sect. 4 for a proof.)

3 Tessellation

We are given a scalar-valued bivariate function defined by quadratic triangular patches. We aim to tessellate the patches into monotone regions, for the purpose of analyzing the topology of level sets of one patch, and, by extension, of a piecewise-defined function composed of many patches. The tessellation will decompose each triangular patch into subpatches, so that each subpatch has a Reeb graph which is a straight line. More specifically, every level set within each subpatch is a single connected component and is homeomorphic to a line segment. We achieve this by ensuring that each subpatch contains exactly one maximum and one minimum, and no saddles. Since we are interested in the topology of the subpatches but not their geometry, we only compute a combinatorial graph structure which captures the connectivity of contours. Some of the arcs of this graph originate from the patch, such as boundary edges, and thus their geometry can be inferred. The embeddings of remaining arcs are not computed since they are not required to construct the Reeb graph.

The construction of the tessellation proceeds by using a case analysis. For each patch, we count the number of line-critical points ($L = 0, 1, 2, 3$) and face-critical points ($F = 0, 1$). Each pair $\langle F, L \rangle$ is handled as an individual case to determine the appropriate tessellation. We first describe the composition of the tessellation. The nodes of a tessellation graph include: (1) all three vertices of the triangle, (2) all line-critical points that exist on the triangle boundary and are not coincident with a triangle vertex, and (3) the face-critical point, assuming that it lies within the triangle and is not coincident with the boundary. We are given (1) as input, but (2) and (3) must be computed in a pre-processing step. Numerical problems can arise. No root finding is needed to compute (2) and (3), so exact arithmetic may be used. However, if speed and consistency are to be favored over accuracy then we only need to ensure that the tessellation graphs of every patch agree on their boundary edges. The existence and location of (3) does not effect the tessellation boundary, so no consistency checks are needed for computing these points. The computation of (2) must agree between two triangles sharing an edge. To ensure this, we do not treat edges as line restrictions of bivariate quadratics, but rather as univariate quadratics defined by data prescribed for the edge. The arcs are included into the tessellation based on the following rules:

- If a line-critical point exists on an edge, we connect that node to both triangle vertices on that edge. Otherwise, if an edge has no line-critical point, then we connect its vertices by an arc.
- If a face criticality does not exist then
  - If there is only one line criticality, we connect it to the opposite vertex, as in Fig. 3 $\langle 0, 1 \rangle$.
  - If there are two line-critical points, we connect each one to the other. The tessellation is not yet a triangulation. There are two possible arcs that can be added to triangulate the quadrilateral shown in Fig. 3 $\langle 0, 2 \rangle$. One, but not both, may be incident on a boundary saddle. We include this arc into the tessellation.

- If there are three line criticalities, we connect each one to the other two, as shown in Fig. 3 $\langle 0, 3\rangle$.

– Otherwise, if a face-critical point exists in the triangle, connect it by an arc to every other node, as shown in Fig. 3 $\langle 1, 1\rangle$, $\langle 1, 2\rangle$ and $\langle 1, 3\rangle$.

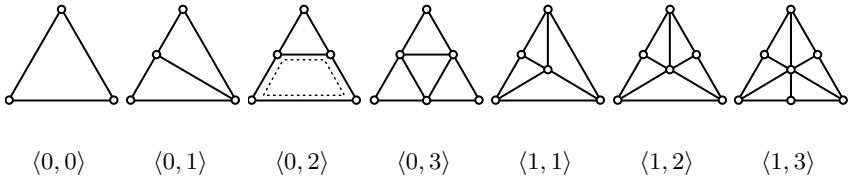

Fig. 3. Tessellation cases $\langle F, L\rangle$, where F is the number of face-critical points and L is the number of line-critical points.

4 Case Analysis

The tessellation constructed using these rules satisfies the monotonicity property. We first state and prove some useful results about configurations of triangle vertices, line-critical points and face-critical points. We assume that all triangles in the triangulation are non-degenerate, i.e., every angle is between 0 and π and every triangle has non-zero, finite area. In the proofs below, we make significant use of two properties of the Reeb graph of the function defined over a triangular patch. The first property is that the Reeb graph does not contain any cycles because the domain is a topological disk [8]. The second property is that the number of extrema in the graph is twice the number of saddles.

Lemma 1. *If a triangle vertex is a boundary saddle of the function restricted to the triangle, then it lies at the intersection of the two hyperbolic asymptotes.*

Proof. We prove this by contradiction. Assume that there exists a vertex v that is a boundary saddle, but not the hyperbolic saddle point. As we sweep the level sets "downward in function value" and pass v, two contours merge or a single contour splits into two. Assume, without loss of generality, that a contour splits into two, as shown in Fig. 4. Above the value of v, the contour is contained entirely in the triangle. Below the value of v, the contour passes outside the triangle and then back in; the triangle cuts the contour into two segments. (These segments may be joined elsewhere, but locally they are distinct.) Consider the segment of the contour approaching v from the right. If this segment is to remain strictly inside the triangle until the sweep arrives at v, its tangent direction is constrained by the triangle edge to the right of v. Similarly, the tangent direction of the contour segment approaching from the left is constrained by the triangle edge to the left of v. All contours except for hyperbolic asymptotes are smooth, and so the tangent on the right of v must agree with the tangent on the left. The edges must be parallel, and therefore the triangle must be degenerate, which violates our assumption.

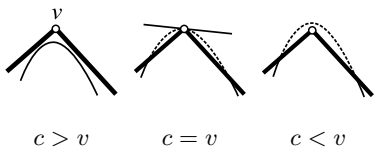

Fig. 4. A smooth contour c cannot both touch the triangle boundary at v and lie completely in the interior

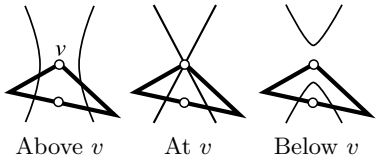

Fig. 5. A triangle containing a boundary saddle at a vertex v must contain a line-critical point on the opposite edge.

Lemma 2. *If a vertex of the triangular patch is a boundary saddle, then the triangle has zero face-critical points and one line-critical point.*

Proof. Considering Lemma 1, if a vertex v is a boundary saddle then v is a face criticality of a hyperbolic function. Therefore, the face criticality does not lie in the interior. The edges incident on v cannot have line criticalities because v is necessarily the line-critical point of all lines that pass through it. The edge opposite v intersects the asymptotes twice and therefore must have a line criticality, see Fig. 5.

Lemma 3. *If a triangular patch contains exactly one line criticality and no face criticality, then that line criticality is reachable by a monotone path from any other point in the triangle.*

Proof. Assume that none of the vertices is a boundary saddle. The line criticality may be an extremum or a saddle. If it is an extremum, the triangle does not contain any boundary saddle. The Reeb graph of the triangular patch is a straight line, and every contour in the patch is homeomorphic to a line segment. We can reach any point from the extremum by walking along contours that monotonically sweep the patch. If the line criticality is a saddle, then it is the only point at which the connectivity of contours in the patch change, because no other saddles exist. The Reeb graph has one internal node and three leaves. Starting from the saddle, we can walk to any other point in the patch by choosing the appropriate contour component to follow as it is swept away from the saddle. Assume that one vertex is a boundary saddle, as shown in Fig. 5. If the line-critical point is also a boundary saddle, the Reeb graph has two saddles, which implies at least four extrema. This is impossible because there are only two vertices left. Therefore, the line criticality must be an extremum. Assume without loss of generality that it is a maximum. The two vertices on that edge must be minima, so the Reeb graph has one maximum, one saddle and two minima. A monotone path exists from the maximum to all points in the triangle.

Lemma 4. *If a triangular patch contains zero face-critical points and two line-critical points, the two line-criticalities are connected by a monotone path.*

Proof. Lemma 2 implies that there are no vertex saddles. Let e_0 and e_1 be the line criticalities. If both e_0 and e_1 are saddles, then they are connected by a

monotone path because they are the only two interior nodes in the Reeb graph of the triangular patch. If both are extrema, then one must be a maximum and the other a minimum and the Reeb graph is a straight line. If e_0 is a boundary saddle and e_1 is an extremum, then there is a monotone path from that boundary saddle to every point in the patch.

We now use the above lemmas to prove that in all cases our tessellations consist of monotone subpatches.

Case $\langle \mathbf{0}, \mathbf{0} \rangle$: No line-critical point exists on any edge, and no face-critical point exists in the triangle. The function is monotone along the edges. There is exactly one maximum and one minimum, which occur at vertices. All contours in the triangle are homeomorphic to a line segment.

Case $\langle \mathbf{0}, \mathbf{1} \rangle$: One line-critical point e exists on an edge, and no face-critical point exists in the triangle. Let v_0, v_1 and v_2 be the triangle vertices, where v_0 is the vertex opposite e. We split the triangle in two, creating patches e, v_1, v_0 and e, v_2, v_0, each containing zero face- and line criticalities. Considering Lemma 3, the arc e, v_0 is guaranteed to have a monotone embedding.

Case $\langle \mathbf{0}, \mathbf{2} \rangle$: Two line-critical points exist, e_0 and e_1, and no face-critical point exists. We first subdivide the triangle along the monotone arc between e_0 and e_1. Considering Lemma 4, we know this arc exists. This arc splits the patch into a triangular subpatch and a quadrilateral subpatch. The triangular subpatch belongs to Case $\langle 0, 0 \rangle$. Both e_0 and e_1 may not be boundary saddles because this implies the Reeb graph of the triangular patch has two saddles and two extrema, an impossible configuration. Let e_0 be a saddle and e_1 an extremum. If we triangulate the quadrilateral by adding an arc which does not terminate at e_0, then e_0 will still be a saddle of its triangular subpatch, which violates our desired monotonicity property. To prevent this, we always triangulate the quadrilateral by adding an arc which has e_0 as an endpoint. If e_0 and e_1 are both extrema, the Reeb graph is a straight line because no other saddles exist.

Case $\langle \mathbf{0}, \mathbf{3} \rangle$: Monotone arcs exist between all three pairs of line criticalities. Please refer to the extended version of this paper for proof [17].

Case $\langle \mathbf{1}, \mathbf{0} \rangle$: This case is impossible. Face-critical points occur in elliptic and hyperbolic functions only. Consider a triangular patch containing an elliptic minimum. Tracking the topology of level sets during a sweep in the increasing function direction, we note that a loop, contained entirely inside the triangle, is created at the minimum. This loop cannot be destroyed without first being converted into a segment by a line criticality. Therefore, the triangle boundary should contain at least one line criticality. A similar argument holds when the triangular patch contains an elliptic maximum. Let us consider a triangular patch containing a hyperbolic face-critical point. The level set at this critical point consists of a pair of intersecting asymptotes. These asymptotes intersect the triangle boundary at four unique points. Since there are only three edges, at least one edge intersects the asymptotes twice. This triangle edge contains a line-critical point between the two intersection points.

Cases $\langle \mathbf{1}, \mathbf{1} \rangle$, $\langle \mathbf{1}, \mathbf{2} \rangle$, $\langle \mathbf{1}, \mathbf{3} \rangle$: We tessellate the patch by connecting the face criticality to all vertices and line criticalities on the boundary. These new arcs

are not only monotone, but they are straight lines as well, because any line-restriction containing the face criticality as an end-point is necessarily monotone. All possible critical points appear as nodes in the tessellation, all new subpatches are triangular and contain zero face and line criticalities.

5 Application to Reeb Graphs

We show how these tessellations may be used to compute a Reeb graph. Reeb graph algorithms, such as the algorithm of Cole-McClaughin *et al.* [8], operate by tracking contours of a level set during a plane sweep of the range of function values. Since all arcs of our tessellation are monotone, any contour intersects an arc at most once. Contours of function values that are not equal to any node intersect the boundary of every triangular patch twice, or not at all. We can follow a contour by starting at one arc that it intersects, and moving to another arc of the same triangle that intersects the contour.

When the domain of a function is planar, such as 2D grey-valued images or terrains/height fields, the Reeb graph contains no cycles and is called a contour tree. Efficient contour tree algorithms proceed in two distinct steps [5, 7]. First, two separate trees, called the *join tree* and *split tree*, are constructed using a variety of methods. The join tree tracks topological changes in the subdomain lying above a level set, and the split tree tracks topological changes in the subdomain lying below a level set. In the second step, the join and split trees are merged to yield the contour tree. Our tessellation graph is a valid input for the join and split tree construction algorithms. Applying any of these algorithms to the tessellation graph of a piecewise-quadratic function yields the correct contour tree for that function. Carr *et al.* [7] and Chiang *et al.* [9] utilize the following properties to ensure the correctness of their algorithms [6]:

1. All critical points of the piecewise function appear as nodes in the graph.
2. For any value h, a path above (below) h exists in the graph if and only if a path above (below) h exists in the domain.

We explicitly include every potential critical point of f in the tessellation, so that the first property is satisfied. We assert that the monotonicity property of our tessellation ensures that properties 2 and 3 are also satisfied.

6 Conclusions and Future Work

We have described a tessellation scheme for piecewise-quadratic functions on surfaces. Our tessellation allows existing Reeb graph and contour tree construction algorithms to be applied to a new class of inputs, thereby extending the applications of these topological structures. We intend to develop a similar tessellation scheme for trivariate quadratic functions defined over tetrahedral meshes. We hope that this work will contribute to the development of robust and consistent topological methods for analysis of functions specified as higher-order interpolants or approximation functions over 2D and 3D domains.

References

1. Reeb, G.: Sur les points singuliers d'une forme de pfaff completement integrable ou d'une fonction numrique. Comptes Rendus Acad. Sciences **222** (1946) 847–849
2. Worsey, A., Farin, G.: Contouring a bivariate quadratic polynomial over a triangle. Comput. Aided Geom. Des. **7**(1-4) (1990) 337–351
3. Nielson, G., Hamann, B.: The asymptotic decider: resolving the ambiguity in marching cubes. In: Proc. Visualization '91, IEEE Computer Society Press (1991) 83–91
4. Nielson, G.: On marching cubes. IEEE Trans. Visualization and Comp. Graph. **9**(3) (2003) 283–297
5. Pascucci, V., Cole-McLaughlin, K.: Efficient computation of the topology of level sets. In: Proc. Visualization '02, IEEE Computer Society Press (2002) 187–194
6. Carr, H.: Topological Manipulation of Isosurfaces. PhD thesis, University of British Columbia (2004)
7. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Comput. Geom. Theory Appl. **24**(2) (2003) 75–94
8. Cole-McLaughlin, K., Edelsbrunner, H., Harer, J., Natarajan, V., Pascucci, V.: Loops in reeb graphs of 2-manifolds. In: Proc. of the 19th annual symposium on computational geometry, ACM Press (2003) 344–350
9. Chiang, Y.J., Lenz, T., Lu, X., Rote, G.: Simple and optimal output-sensitive construction of contour trees using monotone paths. Comput. Geom. Theory Appl. **30**(2) (2005) 165–195
10. Bajaj, C.L., Pascucci, V., Schikore, D.R.: The contour spectrum. In: Proc. Visualization '97, IEEE Computer Society Press (1997) 167–174.
11. Takahashi, S., Nielson, G.M., Takeshima, Y., Fujishiro, I.: Topological volume skeletonization using adaptive tetrahedralization. In: Proc. Geometric Modeling and Processing 2004, IEEE Computer Society Press (2004) 227–236
12. Weber, G.H., Scheuermann, G., Hagen, H., Hamann, B.: Exploring scalar fields using critical isovalues. In: Proc. Visualization '02, IEEE Computer Society Press (2002) 171–178
13. Carr, H., Snoeyink, J., van de Panne, M.: Simplifying flexible isosurfaces using local geometric measures. In: Proc. Visualization '04, IEEE Computer Society Press (2004) 497–504
14. Hilaga, M., Shinagawa, Y., Kohmura, T., Kunii, T.L.: Topology matching for fully automatic similarity estimation of 3d shapes. In: Proce. 28th annual conference on computer graphics and interactive techniques, ACM Press (2001) 203–212
15. Zhang, E., Mischaikow, K., Turk, G.: Feature-based surface parameterization and texture mapping. ACM Trans. Graph. **24**(1) (2005) 1–27
16. Milnor, J.W.: Morse Theory. Princeton University Press, Princeton, New Jersey (1963)
17. Dillard, S.E.: Tessellation of quadratic elements. Technical report, University of California, Davis, Department of Computer Science (2006)