I/O Efficient Path Traversal in Well-Shaped Tetrahedral Meshes

Craig Dillabaugh *

Abstract

We present a data structure which represents a well-shaped convex tetrahedral mesh, $\mathcal{M}$, in linear space such that path traversals visiting a sequence of K tetrahedra require $O(K/\lg B)$ I/O operations in the external memory model. As applications of our structure we show how to efficiently perform axis parallel box queries and how to report the intersection of $\mathcal{M}$ with an arbitrarily oriented plane.

1 Introduction

Traversal of a path visiting the elements of a data structure is a situation that commonly arises in computing. For example, reporting an elevation contour on a digital terrain model, finding a path between nodes in a network, and even searching in a tree structure, can all be viewed as traversing a path. In representing very large graphs in the external memory setting, the graph is often partitioned into *blocks* which correspond to the disk blocks in memory. An effective partitioning of the graph permits the traversal of a path without incurring a large cost in terms of I/Os (input-output operations that occur each time a new block is visited). Nodine *et al.* [8] first studied the problem of graph blocking, and derived bounds for path traversal in several classes of graphs. Agarwal *et al.* [2] describe a $O(N)$ space data structure for planar graphs of bounded degree d, that permits traversal of a path visiting a sequence of K vertices in $O(K/\log_d B)$ I/Os. Our own research, [4], examined succinct representions of such graphs while maintaining the $O(K/\log_d B)$ traversal bound.

In this paper we explore representing a tetrahedral mesh, $\mathcal{M}$, in a manner which permits I/O efficient path traversal. We assume that $\mathcal{M}$ is *well-shaped*, by which we mean that the aspect ratio of each tetrahedron is bounded by a constant. This assumption is valid for instance for mesh generation algorithms that enforce the well-shaped property on their output meshes [6].

1.1 Results

We present a representation for well-shaped convex tetrahedral meshes in $\mathbb{R}^3$, permitting efficient path traversal in the external memory setting. We partition the dual graph of $\mathcal{M}$ into regions and store these regions in disk blocks. We also construct fixed-size neighbourhoods around the vertices forming the boundaries between the regions, and store a collection of these neighbourhoods in blocks. I/O efficiency of path traversal is guaranteed by the fact that the number of vertices we can traverse every time we encounter a boundary vertex (tetrahedron in the primal) is bounded from below. Our mesh representation has two key requirements. First we must be able to partion $\mathcal{M}$'s dual into block sized regions, and secondly the number of *boundary* vertices occurring at the intersection of the regions must be small. We will demonstrate that these requirements can be met for well-shaped meshes.

We give a representation of a well-shaped convex tetrahedral mesh that can be stored in $O(N)$ space and that permits traversal of a path visiting a sequence of K tetrahedra in $O(K/\lg B)$ [1] I/Os. We have not yet completed detailed analysis of the pre-processing time, but the geometric separator algorithm we employ [7] can partition a mesh in $scan(N)$ I/Os. Since the recursive partitioning will require no more than $O(\log N)$ steps the preprocessing requires at worst $O(\log N \cdot scan(N))$ I/Os. As applications of our technique, we demonstrate how our structure can be used to report the intersection of $\mathcal{M}$ with an axis-parallel box, or an arbitrarily oriented plane $\mathcal{H}$, in $O(K/\lg B)$ I/Os. For axis-parallel box queries, the Priority R-Tree [1] permits reporting in $O((N/B)^{2/3} + K/B)$ I/Os, but cannot efficiently handle arbitrarily oriented plane queries. In each case K is the total number of tetrahedra intersecting the query box, or plane, respectively. Further, we note that the $(N/B)^{2/3}$ term for the Priority R-Tree is the search time in $\mathbb{R}^3$, which we do account for in our analysis.

2 Mesh Partitioning

The tetrahedral mesh $\mathcal{M}$ is composed of vertices, edges, faces, and tetrahedra. The vertices, edges and faces form the *skeleton* of the mesh, and we say a tetrahedron is *adjacent* to elements of the skeleton that compose its boundary. Two tetrahedra are adjacent if and only if, they share a face. Let $\mathcal{M}^*$ be the dual graph of $\mathcal{M}$. For each tetrahedron $t \in \mathcal{M}$, $\mathcal{M}^*$ contains a vertex $v(t)$ corresponding to t. Two vertices, $v(t)$ and $v(t')$, are connected by an edge in $\mathcal{M}^*$ if the corresponding tetra-

*Computational Geometry Lab, School of Computer Science, Carleton University, cdillaba@connect.carleton.ca

[1] Unless otherwise stated we use lg to represent $\log_4$.

hedra share a face. Exclusive of the outer face (which we do not represent in $\mathcal{M}^*$) the degree of any vertex in $\mathcal{M}^*$ is at most 4.

For tetrahedron t we let $R(t)$ be the radius of the smallest enclosing sphere of t, and $r(t)$ be the radius of the largest sphere that can be inscribed in t. The *aspect ratio* of t, denoted c_t, is defined as the ratio of these two radii, $c_t = R(t)/r(t)$. If the largest aspect ratio of each tetrahedron $t \in \mathcal{M}$ is bounded by a constant, c, then we say $\mathcal{M}$ is *well-shaped*.

In a *geometric graph* a d dimensional coordinate is associated with each vertex. These coordinates yield an embedding of the graph in $\mathbb{R}^d$. Miller *et al.* [7] studied the problem of finding separators for geometric graphs. More specifically they developed a technique for finding separators for k-ply neighbourhood systems defined as follows.

Definition 1 ([7]) *A* k-ply neighbourhood system *in* d *dimensions is a set* $B_1, \ldots, B_n$ *of* n *closed balls in* $\mathbb{R}^d$ *such that no point in* $\mathbb{R}^d$ *is strictly interior to more than* k *balls.*

The authors were able show that separators could be found on several concrete classes of graphs that can be represented as k-ply *neighbourhood systems*. The authors extend this work to the class of α*-overlap graphs*; they are defined as follows:

Definition 2 ([6]) *Let* $\alpha \geq 1$ *be given, and let* $(B_1, \ldots, B_n)$ *be a 1-ply neighbourhood system. The* α-overlap graph *for this neighbourhood system is the undirected graph with vertices* $V = 1, \ldots, n$ *and edges:*

$$E = \{(i,j) : B_i \cap (\alpha \cdot B_j) \neq \emptyset \wedge (\alpha \cdot B_i) \cap B_j \neq \emptyset\}$$

For the class of α-overlap graphs the paper's main result is summarized in Lemma 1.

Lemma 1 ([6]) *Let* G *be an* α*-overlap graph in a fixed dimension* d*. Then* G *has an* $O\left(\alpha \cdot n^{(d-1)/d} + q(\alpha, d))\right)$ *separator that* $(d+1)/(d+2)$*-splits.*

In our setting $d = 3$, and α is fixed, thus the $q(\alpha, d)$ term, which is a function of two constants (representing the maximum degree of a vertex), is constant, and we are left with a separator of size $O\left(n^{2/3}\right)$. The geometric separator can also be used to find a vertex and edge separator on a well-shaped tetrahedral mesh [6]. Next we show that the dual graph is well-shaped and the separator can likewise be applied to partition $\mathcal{M}^*$.

Lemma 2 *Let* $\mathcal{M}$ *be a tetrahedral mesh where each tetrahedron* $t \in \mathcal{M}$ *has aspect ratio bounded by* c*. Let* $\mathcal{M}^*$ *be the dual graph of* $\mathcal{M}$*, then* $\mathcal{M}^*$ *is a subgraph of an* α*-overlap graph for* $\alpha = 2c$*.*

Proof. Consider the following neighbourhood system. Let $v(t) \in \mathcal{M}^*$ be a vertex corresponding to tetrahedron $t \in \mathcal{M}$. Let $b(t)$ the the largest inscribed ball in t, with radius $r(t)$ and centred at point $p(t)$. The collection of balls $\mathcal{B} = \{b(1), \ldots, b(n)\}$ correspond to the n tetrahedra of $\mathcal{M}$ and the vertices of $\mathcal{M}^*$, and form a 1-ply system. No two balls can overlap, since each is inscribed within a single tetrahedron. We show that $\mathcal{B}$ forms an α-overlap graph, for $\alpha = 2c$.

Since each $b(t)$ is maximal, it touches each of the four faces of t at a point. Let $B(t)$ be the smallest ball that wholly contains t, recall that this ball has radius $R(t)$. Furthermore let the point $P(t)$ be the centre of $B(t)$. Since $B(t)$ wholly contains t, it also contains $b(t)$ and its centre $p(t)$. Therefore, the distance between the centres of these two balls, $P(t)$ and $p(t)$, is less than the radius, $R(t)$, of the larger ball. Consider the ball of radius $2R(t)$ centered at $p(t)$. Since its distance to $P(t)$ is less than $R(t)$, this ball contains both $P(t)$ and the entire ball $B(t)$, which wholly contains t.

For $i = 1, \ldots, 4$, let $b(i) \in \mathcal{B}$ represent the neighbours of $v(t)$ in $\mathcal{M}^*$. Each $b(i)$ touches the face of t in $\mathcal{M}$ across which its corresponding tetrahedron is adjacent to t. The ball centered at $p(t)$ of radius $2R(t)$ fully contains t (and all its faces) and thus intersects each of these balls.

The aspect ratio of all $t \in \mathcal{M}$ is bounded by c, so for the aspect ratio, $c(t)$, of any tetrahedron, we have that $c(t) \leq c$. Increasing the radius of a ball $b(t)$ to $2R(t)$ is equivalent to $2\frac{R(t)}{r(t)}r(t) = 2c(t) \cdot r(t) \leq 2c \cdot r(t)$. Thus, the overlap graph is an α-overlap graph with $\alpha = 2c$. □

Lemma 3 *Given the dual graph of a tetrahedral mesh with bounded aspect ratio* c*, there is a partitioning algorithm which produces a* $O\left(n^{\frac{2}{3}}\right)$ *separator which* $\frac{4}{5}$*-splits* $\mathcal{M}^*$*.*

Proof. By Lemma 2, we demonstrated that $\mathcal{M}^*$ is a subgraph of an α-overlap graph for $\alpha = 2c$. Applying the separator theorem of Lemma 1 yields a $O\left(n^{\frac{2}{3}}\right)$ separator that $\frac{4}{5}$-splits $\mathcal{M}^*$. □

This separator is sufficient to partition $\mathcal{M}^*$, but we must still show that it can be recursively applied to split $\mathcal{M}^*$ into block sized regions while bounding the total number of boundary vertices. In Lemma 4 we extend the result of [5] on planar graphs to well-shaped tetrahedral meshes.

Lemma 4 *An* n*-vertex dual graph,* $\mathcal{M}^*$*, of a well-shaped tetrahedral mesh can be divided into* $O(n/r)$ *regions with no more than* r *vertices each, and* $O\left(\frac{n}{r^{1/3}}\right)$ *boundary vertices in total.*

Proof. By Lemma 3 we can subdivide the dual graph $\mathcal{M}^*$, on n vertices, into two regions of size δn and

$(1-\delta)n$ for $\frac{1}{5} \leq \delta \leq \frac{4}{5}$ with a separator of size $\beta = O\left(n^{\frac{2}{3}}\right)$. Each region retains the vertices in the separator (boundary vertices), and the separator is recursively applied to the new regions until we have regions of maximum size r.

Let v be a boundary vertex in the resulting subdivided dual graph. Let $d(v)$ be one less than the number of regions that contain v, and let $D(n,r)$ be the sum of the $d(v)$'s over all boundary vertices for a graph of size n with regions of maximum size r. $D(n,r)$ can be calculated by the recurrence: $D(n,r) \leq cn^{2/3} + D(\delta n + O\left(n^{2/3}\right), r) + D((1-\delta)n + O\left(n^{2/3}\right), r)$ for $n > r$, and $D(n,r) = 0$ for $n \leq r$, where $c > 1$ is a constant, and $\frac{1}{5} \leq \delta \leq \frac{4}{5}$. It can be shown by induction that $D(n,r) \in O\left(\frac{n}{r^{1/3}}\right)$. Since all boundary vertices have $d(v) \geq 1$ this value also bounds the total number of boundary vertices. □

3 Data Structure and Navigation

We apply the recursive partitioning algorithm on $\mathcal{M}^*$ to produce regions of at most B vertices, where B is the disk block size. Each region is stored in a single block in external memory. The block stores both the dual vertices and the corresponding geometry from $\mathcal{M}$. Next, around each boundary vertex we identify a neighbourhood which we call its α-neighbourhood. The α-neighbourhood is selected by performing a breadth first search, starting at the boundary vertex, and retaining the subgraph of $\mathcal{M}^*$ induced on the first $\alpha = \sqrt{B}$ vertices encountered. We can store the regions in $O(N/B)$ blocks, and the $O\left(N/\sqrt{B}\right)$ α-neighbourhoods in $O(N/B)$ blocks. Thus the total space is linear.

To traverse the data structure, assume that we start with some tetrahedron t interior to a region for which the corresponding block is loaded. We follow the path to a neighbour of t. If the neighbour is a boundary vertex we load the α-neighbourhood. Since $\mathcal{M}^*$ is of bounded degree loading an α-neighbourhood guarantees at least $\log_4 \sqrt{B} = O(\lg B)$ progress along the path before another I/O is incurred. A path of length K can be traversed with $O(K/\lg B)$ I/Os. We summarize our results by the following theorem.

Theorem 5 *Given a convex tetrahedral mesh, $\mathcal{M}$, of bounded aspect ratio, there is an $O(N/B)$ block representation of $\mathcal{M}$ that permits traversal of a path which visits a sequence of K tetrahedra of $\mathcal{M}$ using $O\left(\frac{K}{\lg B}\right)$ I/Os.*

4 Applications

We now demonstrate the application of our data structures for reporting the results of box and plane intersection queries.

Depth First Traversal: To begin we demonstrate how we can perform a depth first traversal on a mesh represented using our structures, as intersection queries can be answered as special cases of depth first traversal. A challenge in performing depth first traversals in $\mathcal{M}^*$ is that depth first traversal algorithms generally mark vertices as visited to avoid revisiting them from another execution branch. In our data structure, vertices may appear in multiple blocks, and loading all copies of a vertex to mark them as visited destroys the I/O efficiency of the structure. Thus we need a means of determining if a vertex in $\mathcal{M}^*$ (tetrahedron in $\mathcal{M}$) has already been visited. This can be achieved using the following lemma based on De Berg *et al.* [3].

Lemma 6 ([3]) *Given a convex tetrahedral mesh $\mathcal{M}$ and a tetrahedron $t_s \in \mathcal{M}$, then for every tetrahedron $(t \neq t_s) \in \mathcal{M}$, a unique* entry face *can be selected such that the edges in the dual $\mathcal{M}^*$ corresponding to the entry faces implicitly form a tree rooted at $t_s^* \in \mathcal{M}^*$.*

The entry faces are selected by picking a special tetrahedron t_s and a reference point p_s interior to t_s. We perform a depth first traversal of $\mathcal{M}^*$ on the implicit tree rooted at t_s^*. The selection of entry faces is based entirely on p_s and the geometry of the current vertex of $\mathcal{M}^*$ (recall that each dual vertex stores the corresponding tetrahedron geometry). Depth first traversal in a tree does not require the use of mark bits.

We now address the problem of reporting the features of the mesh skeleton during traversal without double reporting. Consider the point p_s, and assume that the endpoints of no line segment are collinear with p_s and that no three points defining a face are co-planar with p_s (these assumptions can be removed, but we omit details here). With respect to a tetrahedron, t, we say that any vertex, line, or face adjacent to t is *invisible* if the line connecting p_s with any point on that feature intersects the interior of t, otherwise we say the feature is *visible*. For any element k on the skeleton of $\mathcal{M}$ the neighbourhood of k is the collection of tetrahedra adjacent to k. We give without proof the following lemma and theorem. The lemma leads to a technique for reporting the skeleton without duplicates, while the theorem summarizes our results for depth first traversal.

Lemma 7 *For any element k on the skeleton of $\mathcal{M}$, there is one, and only one, tetrahedron t in the neighbourhood of k, for which k is invisible with respect to p_s.*

Theorem 8 *Given a convex well-shaped tetrahedral mesh $\mathcal{M}$ of N tetrahedra, $\mathcal{M}$ can be represented in linear space such that from an arbitrary tetrahedra, $t \in \mathcal{M}$, a depth first traversal can be performed that reports each*

tetrahedron in $\mathcal{M}$ once, and all features on the skeleton of $\mathcal{M}$ without duplication in $O(N/\lg B)$ I/Os.

Box Queries: Assume we are given an axis parallel box in $\mathbb{R}^3$. Select as a starting point one of the box's corner points, and assume we are given the tetrahedron $t \in \mathcal{M}$ containing this point. We modify the entry face selection rule so that only faces from among those that intersect the interior of the box can be selected. We then build an implicit tree in $\mathcal{M}^*$ that visits all tetrahedra that intersect the box interior. Again using our structure this allows us to report the intersection in $O(K/\lg B)$ I/Os.

Plane Intersection Queries: As a visualization tool, we wish to intersect a convex tetrahedral mesh $\mathcal{M}$ by a plane $\mathcal{H}$ and report the intersection of $\mathcal{M}$ and $\mathcal{H}$. Let K be the number of tetrahedra in $\mathcal{M}$ intersected by $\mathcal{H}$, including tetrahedra that intersect $\mathcal{H}$ only in their skeleton. Further, assume that we are given some tetrahedra $t_s \in \mathcal{T}$ that is part of the intersection set. The intersection of $\mathcal{M}$ by $\mathcal{H}$ produces a 2 dimensional planar subdivision mapped onto $\mathcal{H}$ which we will denote $\mathcal{M}_H$. Faces in $\mathcal{M}_H$ correspond to the intersection of $\mathcal{H}$ with the interior of a tetrahedron $t \in \mathcal{M}$. Edges in $\mathcal{M}_H$ correspond to the intersection of $\mathcal{H}$ with a face of t, while vertices correspond to the intersection of $\mathcal{H}$ with an edge of t. A face of $t \in \mathcal{M}$ which lies directly on $\mathcal{H}$ appears as a face in $\mathcal{M}_H$. Likewise edges and vertices from $\mathcal{M}$ that fall exactly on $\mathcal{H}$ appear as edges and vertices respectively on $\mathcal{M}_H$.

Let $\mathcal{H}^+$ be the open half-space above $\mathcal{H}$, and $\mathcal{H}^-$ the open half-space below $\mathcal{H}$. Let $\hat{\mathcal{M}}_{\mathcal{H}}$ be the set of tetrahedra in $\mathcal{M}$ which intersect both $\mathcal{H}$ and $\mathcal{H}^+$. Tetrahedra in $\hat{\mathcal{M}}_{\mathcal{H}}$ have some point in their interior in $\mathcal{H}^+$ and intersect $\mathcal{H}$. We select entry faces as before with one minor modification. We measure the distance between p_s and tetrahedron t on the plane $\mathcal{H}$, and disqualify as a candidate entry face any face which does not extend into $\mathcal{H}^+$.

Lemma 9 *It is sufficient to visit the tetrahedra in $\hat{\mathcal{M}}_{\mathcal{H}}$ to report all faces, edges, and vertices of $\mathcal{M}_H$.*

Proof. We must show that every vertex, face, or edge in $\mathcal{M} \cup \mathcal{H}$ is adjacent to a tetrahedron $t \in \hat{\mathcal{M}}_{\mathcal{H}}$. A face that lies exactly on $\mathcal{H}$ separates two tetrahedra one of which must lie in $\mathcal{H}^+$. Likewise any edge or vertex which lies directly on $\mathcal{H}$ is adjacent to a tetrahedron in $\mathcal{H}^+$. Since these tetrahedron are adjacent to features on $\mathcal{H}$ they are included in $\hat{\mathcal{M}}_{\mathcal{H}}$ and therefore are reported. □

Lemma 10 *Let t be a tetrahedron in $\hat{\mathcal{M}}_{\mathcal{H}}$. Let $t' \in \mathcal{M}$ be the tetrahedron adjacent to t' across the face* `entry`(t)*, then t' is also an element of $\hat{\mathcal{M}}_{\mathcal{H}}$.*

Now consider the dual graph $\mathcal{M}^*$ of $\mathcal{M}$. Selecting the tetrahedra in $\hat{\mathcal{M}}_{\mathcal{H}}$, defines a subset of the vertices of $\mathcal{M}^*$. Let $\hat{\mathcal{M}}^*_{\mathcal{H}}$ be the subgraph induced on this subset.

Lemma 11 *Defining the set of entry faces as above produces an implicitly tree rooted at $v(t_s)$ on $\hat{\mathcal{M}}^*_{\mathcal{H}}$.*

Proof. The proof is essentially the same as for Lemma 6. Since the entry edges are still unique, there is no danger of introducing cycles. By Lemma 10 each entry face leads to another tetrahedron in $\hat{\mathcal{M}}_{\mathcal{H}}$, so connectivity is maintained. □

Lemma 11 leads to a simple traversal algorithm for reporting the intersection of $\mathcal{M}$ with $\mathcal{H}$. We perform a depth first traversal of $\hat{\mathcal{M}}^*_{\mathcal{H}}$ on the implicit tree defined by the entry edges and report the intersection of each tetrahedron with $\mathcal{H}$. If we visit a total of K tetrahedra the total path length of the traversal is $O(K)$ steps, so if we represent $\hat{\mathcal{M}}^*_{\mathcal{H}}$ with the I/O efficient representation of $\mathcal{M}$ we can report the intersection in $O(K/\lg B)$ I/Os.

The correctness of the box query algorithm does not rely on the fact that the box is axis parallel. In fact given a starting point interior to any arbitrarily oriented convex shape we can report the intersection in $O(K/\lg B)$ I/Os. Plane intersection can be viewed as a degenerate case of reporting an arbitrarily oriented box query with a box of depth zero. For both the box and plane intersection queries we have presented we have assumed that we are given a starting tetrahedron t_s, interior to the query region. Efficiently finding t_s represents the next step in our research.

References

[1] L. Arge, M. de Berg, H. Haverkort, K. Yi The priority R-tree: A practically efficient and worst case optimal R-tree ACM Transactions on Algorithms, 4(1), 2008.

[2] P.K. Agarwal, L. Arge, T.M. Murali, K.R. Varadarajan, and J.S. Vitter I/O-efficient algorithms for contourline extraction and planar graph blocking. *SODA*, 117-126, 1998.

[3] M. de Berg, M. van Kreveld, R. van Oostrum, and M.H. Overmars Simple traversal of a subdivision without extra storage. *Int. J. Geo. Info. Science*, 11(4):359-373, 1997.

[4] C. Dillabaugh, M. He, A. Maheshwari, and N. Zeh I/O and space efficient path traversal in planar graphs. *ISAAC*, 1175-1184, 2009.

[5] G. N. Frederickson Fast algorithms for shortest paths in planar graphs, with applications. *SIAM J. Comput.*, 16(6):1004-1022, 1987.

[6] G. L. Miller, S.-H. Teng, W.P. Thurston, and S.A. Vavasis Geometric separators for finite-element meshes. *SIAM J. Sci. Comput.*, 19(2):364-386, 1998.

[7] G. L. Miller, S.-H. Teng, W.P. Thurston, and S.A. Vavasis Separators for sphere-packings and nearest neighbor graphs. *J. ACM*, 44(1):1-29, 1997.

[8] M.H. Nodine, M.T. Goodrich, and J.S. Vitter Blocking for external graph searching. *Algorithmica*, 16(2):181-214, 1996.