The Cost of Compatible Refinement of Simplex Decomposition Trees

F. Betul Atalay[1] and David M. Mount[2]

[1] Mathematics and Computer Science Department, Saint Joseph's University, Philadelphia, PA. fatalay@sju.edu

[2] Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD. mount@cs.umd.edu

Summary. A hierarchical simplicial mesh is a recursive decomposition of space into cells that are simplices. Such a mesh is *compatible* if pairs of neighboring cells meet along a single common face. Compatibility condition is important in many applications where the mesh serves as a discretization of a function. Enforcing compatibility involves refining the simplices of the mesh further, thus generates a larger mesh. We show that the size of a simplicial mesh grows by no more than a constant factor when compatibly refined. We prove a tight upper bound on the expansion factor for 2-dimensional meshes, and we sketch upper bounds for d-dimensional meshes.

1 Introduction

Hierarchical data structures based on repeated subdivision of space have been widely used in application areas such as finite element analysis, computer graphics, scientific visualization, geometric modeling, image processing and geographic information systems. In many such applications, the spatial decomposition serves as a discretization of the domain of a scalar or vector field, which associates each point of real d-dimensional space with a scalar or vector value, respectively. The field values are sampled at the vertices of the subdivision, and for any other query point the field value could be computed by an appropriate (often linear) interpolation of the field values at the vertices of the cell that contains it. The subdivision is adaptively refined to improve the approximation of the field at regions of high variation.

Our interest in this paper is on simplicial decompositions, particularly on *regular hierarchical simplicial meshes* [10, 2]. This is a generalization of the concept of hierarchical regular triangulation in the plane. Each element of such a mesh is a d-simplex, that is, the convex hull of $d + 1$ affinely independent points [4]. A simplicial mesh is said to be *regular* if the vertices of the mesh are regularly distributed and the process by which a cell is subdivided is identical for all cells. The regular simplicial mesh that we consider is generated by a process of repeated bisection applied to a hypercube that has been initially subdivided into $d!$ congruent simplices. The subdivision pattern repeats itself on a smaller scale at every d levels.

A simplicial mesh is called *compatible* if pairs of neighboring cells meet along a single common face. A compatible simplicial mesh is also referred to as a simplicial complex. (See Fig. 1 for a 2-dimensional example.) The compatibility condition is important since otherwise cracks may occur along the faces of the subdivision, which in turn causes discontinuities in the function and presents problems when using the mesh for interpolation. A compatible mesh ensures at least C^0 continuity and is desirable for many applications. 2-dimensional simplicial meshes have been used for multi-resolution terrain modeling and rendering [5, 9, 3, 12, 6]; 3-dimensional meshes for volume rendering of 3-dimensional scalar fields (such as medical datasets) [7, 14], and 4-dimensional meshes for visualization of time-varying flow fields.

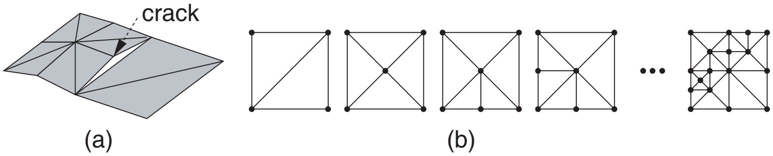


Fig. 1. (a) A crack **(b)** Compatible simplicial mesh in the plane

Refining a simplicial mesh to enforce compatibility requires refining additional simplices if they share split faces with their neighbors. The cost of compatible refinement is that a larger mesh will be generated. Our goal in this paper is to show that when a simplicial mesh is refined to enforce compatibility, its size will grow by no more than a constant factor. We prove a tight upper bound on the expansion factor for 2-dimensional meshes, and upper bounds for d-dimensional meshes.

Previously, Weiser [13] and Moore [11] proved results on the cost of *restricting* quadtrees. A *restricted quadtree* is a quadtree in which two neighboring leaf cells in the quadtree may differ at most by one level [8]. Moore calls a restricted quadtree as a *1-balanced quadtree*. Weiser showed that a square quadtree grows no more than nine times bigger and a triangular quadtree grows no more than thirteen times bigger when refined to provide 1-balance [13]. Moore later showed Weiser's bounds could be reduced by showing that a square quadtree grows at most eight times larger, and triangular quadtrees grow ten times larger when refined for 1-balance [11]. Moore also showed that his bounds are tight.

We follow Moore's methodology to show similar results for the family of bisection-based regular hierarchical simplicial meshes. Note that, Moore's triangular quadtrees are constructed by repeatedly subdividing a triangle into four smaller triangles which are similar to the original triangle. Moore also analyzes degree-three triangular “quadtrees” where each triangle is subdivided into nine smaller similar triangles. In any case, all the triangles in the entire triangular quadtree are similar triangles. The simplicial meshes that we consider in this paper arises from a different family of meshes, which are constructed by repeatedly subdividing a simplex into two child simplices which are congruent to each other but not similar to their parent. Thus, the subdivision rule is different from a triangular quadtree and there is more than one similarity class of simplices in the mesh. This bisection-based subdivision rule can be applied in any dimension d. In addition, Moore’s analysis is based on 1-balancing, whereas we are interested in compatibility refinement which imposes a tighter requirement.

The remainder of the paper is organized as follows. In Section 2 we present basic definitions and describe regular hierarchical simplicial meshes. In Section 3 we prove an upper bound on the size of a 2-dimensional compatibly refined simplicial mesh. In Section 4, we prove that this upper bound is asymptotically tight. In Section 5, we sketch an upper bound for d-dimensional meshes.

2 Preliminaries

The *simplex decomposition tree* (*sd-tree* for short) is a collection of binary trees representing a regular hierarchical simplicial mesh in real d-dimensional space. Assume that the domain of interest has been scaled to lie within a unit *reference hypercube*. The reference hypercube is initially subdivided into $d!$ congruent simplices that share the major diagonal. It is well known that the collection of these simplices fully subdivide the hypercube, and further that this subdivision is compatible [1]. These $d!$ simplices form the starting point of our simplicial decomposition. Simplices are then refined by a process of repeated subdivision, called *bisection*, in which a simplex is bisected by splitting one of its edges at its midpoint. The edge to be bisected is determined by a specific vertex ordering [10, 2]. Intuitively, this bisection scheme alternates bisecting the major diagonal of the hypercube first, then the diagonals of the $d-1$ faces, then the diagonals of the $d-2$ faces, and so on, finally bisecting the edges (1-faces) of the hypercube. (In the 2-dimensional and the 3-dimensional case, this bisection scheme is equivalent to bisecting the longest edge of the simplex.) Hence, each of the $d!$ coarse simplices at the highest level is the root of a separate binary tree, which are conceptually joined under a common super-root corresponding to the hypercube. See Fig. 2 for a 2-dimensional subdivision and the corresponding *sd-tree*.

Define the *level*, ℓ, of a simplex to be the *depth* modulo the dimension, that is, $\ell = (p \bmod d)$, where p denotes the depth of a simplex in the tree. The depth of a root simplex is zero, and of any other simplex is one more than the depth of its parent.

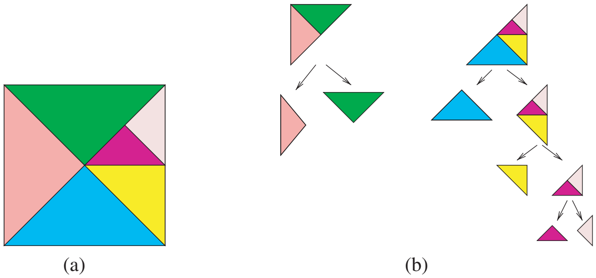


Fig. 2. (a) 2-dimensional simplicial subdivision **(b)** The corresponding sd-tree

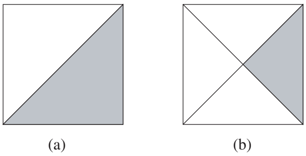


Fig. 3. (a) *level-0* simplex **(b)** *level-1* simplex

Maubach [10] showed that with every d consecutive bisections, the resulting simplices are similar copies of their d-fold grandparent, subject to a uniform scaling by 1/2. Thus, the pattern of decomposition repeats every d levels in the decomposition. Since the two children of any simplex are also *congruent*, it follows that all the simplices at any given level of the decomposition tree are congruent to each other. Thus, all the similarity classes can be represented by d canonical simplices, one per level. In Fig. 3(a) and (b) the shaded simplices denote the two canonical simplices for a 2-dimensional subdivision. Notice, for example, that any simplex in the subdivision shown in Fig. 1 is congruent to either one of the two canonical simplices. Consequently, it suffices to consider only the canonical simplices when analyzing the structure of the tree.

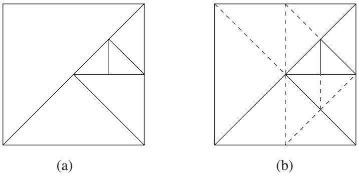


Fig. 4. (a) Before and **(b)** after compatible refinement

A d-dimensional simplex decomposition tree is said to be *compatible*, if each simplex in the subdivision shares a $(d-1)$-face with exactly one neighbor simplex. If the subdivision is not compatible, we can further refine simplices to provide compatibility. Consider the non-compatible subdivision in Fig. 4(a). The simplices that share a bisected edge with already bisected simplices need to be bisected as well. However, note that new bisections will possibly trigger more bisections at the higher levels of the tree. In Fig. 4(b) the dashed edges illustrate the bisections triggered due to compatibility refinement of the subdivision shown in (a).

3 Upper Bound for 2-Dimensional Decompositions

In this section, we prove that the size of a 2-dimensional regular hierarchical simplicial mesh grows only by a constant factor when compatibly refined.

Theorem 1. *A non-compatible simplex decomposition tree in 2-dimensional space with n nodes can be compatibly refined to form a simplex decomposition tree with no more than $14n$ nodes.*

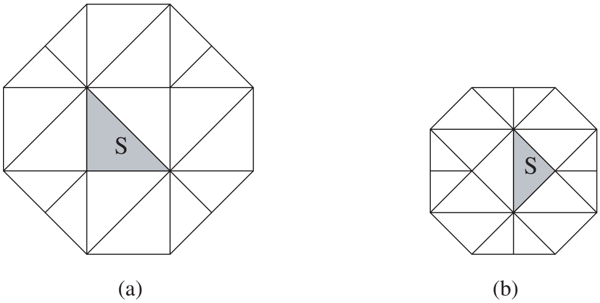


Fig. 5. (a) Barrier of a *level-0* simplex **(b)** Barrier of a *level-1* simplex

We follow a similar method as Moore [11] to prove this theorem. We start by finding a *barrier*, that is, a configuration of simplices around a particular simplex S. Then, we show that if such a barrier is produced after a series of splits (possibly none), then simplex S will never split during compatibility refinement. Recall that, in the 2-dimensional case, we have two canonical simplices which are shown in Fig. 3(a) and (b). We call them *level-0* simplices and *level-1* simplices, respectively. The barriers for each class of simplices are illustrated in Fig. 5(a) and (b). Before proving the above theorem, we first introduce the notion of a safe simplex and prove a lemma that shows that such simplices cannot be split.

Definition 1. (Safe Simplex) *A simplex S is* safe *if none of the barrier elements are split initially or such a barrier comes into existence after any series of splits.*

Lemma 1. *A safe simplex S will never split during compatibility refinement.*

Proof. (of *Lemma 1*)

We will prove the lemma by induction on the depth of the simplex in the simplex decomposition tree. Let p denote the depth of the simplex S.

Basis

If S is the deepest leaf, S will not split since a simplex will only split if it has a split neighbor.

Induction Step

Assume that our inductive hypothesis holds for simplices at depth $p + 1$. We will show that the inductive hypothesis holds for a simplex S at depth p. We need to consider the two canonical simplices separately.

Let us first consider the case where S is a *level-0* simplex, that is, p mod $2 = 0$. Note that for S to split, either one of the three neighboring simplices shown in Fig. 6(a) should split. If none of the three neighbor simplices labeled 0, 1 and 2 split, S will never split. Suppose that S is initially surrounded by the barrier shown in Fig. 6(b), such that no barrier element is split. Even if all the boundary barrier elements that are at depth p split due to compatibility refinement, we will have the structure depicted in Fig. 6(c). Notice that the neighbors labeled 0 and 2 are at depth $p+1$, and they have their own barriers. Thus, these neighbors are safe. The barrier for neighbor 0 is depicted with thick lines in Fig. 6(d), and the barrier for neighbor 2 is symmetric to that. Therefore, by the inductive hypothesis that was assumed to hold for simplices at depth $p + 1$, neighbors 0 and 2 need not split.

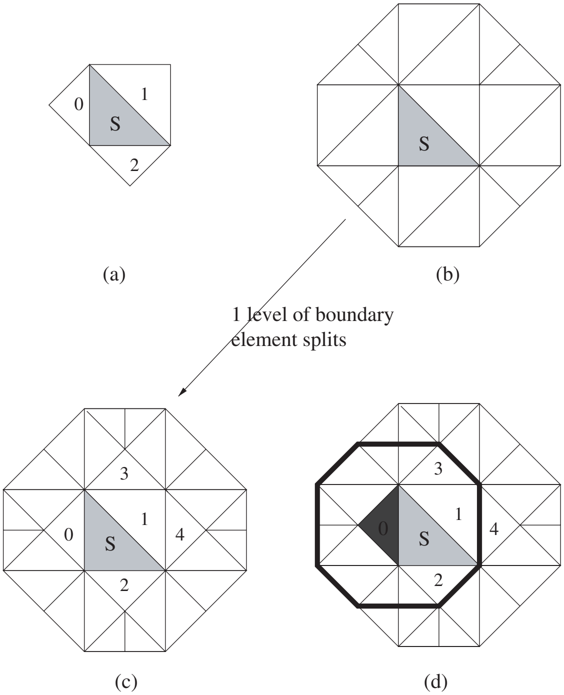


Fig. 6. Induction step for a *level-0* simplex

Similarly simplices labeled 3 and 4 at depth $p + 1$ are surrounded by a barrier and they will not split. If 3 and 4 do not split, neighbor simplex 1 will not split. Thus, none of the neighbor simplices of S will split ensuring that S will not split. This concludes the case of a *level-0* simplex.

The case of S being a *level-1* simplex can be proved similarly. Suppose that S is initially surrounded by the barrier shown in Fig. 7(a), such that no barrier element is split. Even if all the boundary barrier elements that are at depth p split due to compatibility refinement, we will have the structure depicted in Fig. 7(b). Notice that if none of the three neighbor simplices of S which are labeled 0, 1 and 2 split, S will never split. Neighbors 1 and 2 which are at depth $p+1$ have their own barriers, and so, they are safe. The barrier for neighbor 1 is depicted with thick lines in Fig. 7(c). The barrier for neighbor 2 is symmetric. Therefore, neighbors 1 and 2 need not split. Similarly, simplices labeled 3 and 4 are at depth $p + 1$ and have their own barriers, preventing neighbor 0 to split. Since none of the neighbor simplices of S split, S will not split. □

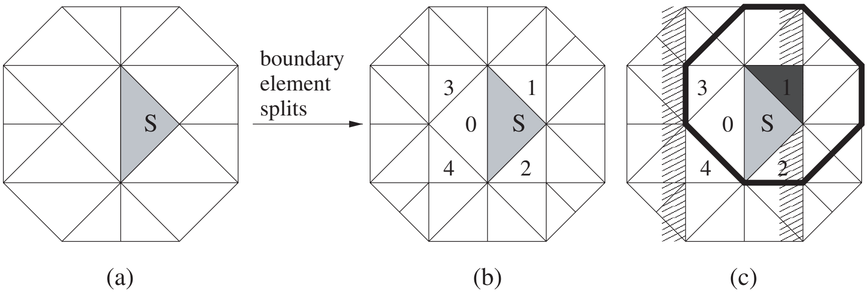


Fig. 7. Induction step for a *level-1* simplex

Proof. (of *Theorem 1*)

Using Lemma 1, if a simplex S splits, it must not be safe, that is, one or more of its barrier elements is initially split. We will hold one of its barrier elements responsible for splitting of S. A split element R may be responsible for splitting 13 other split elements, since it may be in the barrier of 13 possible elements as depicted in Fig. 8 when R is a *level-0* simplex, and in Fig. 9 when R is a *level-1* simplex. In Fig. 8, part (a) shows the nine possible *level-0* simplices whose barriers contains R, and part (b) shows the four possible *level-1* simplices whose barriers may contain R. Thus, compatibly refining a simplex decomposition tree could increase the number of nodes by at most a factor of 14. □

4 Tightness of the Upper Bound

In this section, we demonstrate that the upper bound of *Theorem 1* is asymptotically tight by constructing an infinite family of simplex decomposition trees, each of which grow fourteen times larger less an additive constant, when

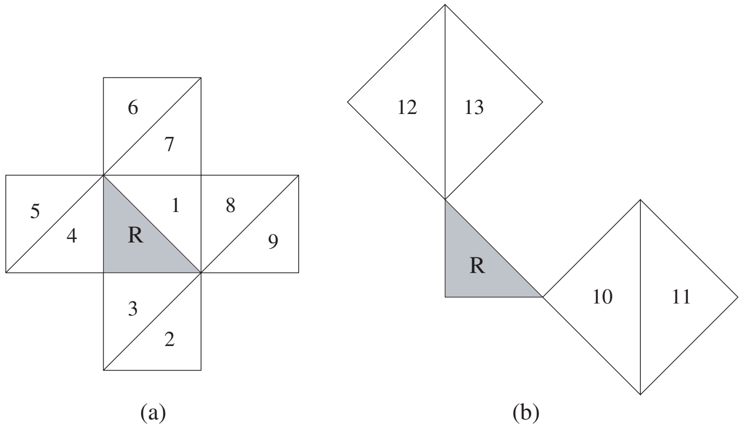


Fig. 8. 13 simplices whose barriers contain *level-0* simplex R

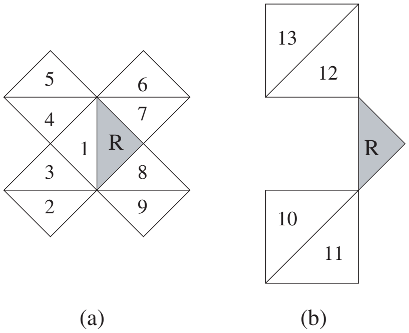
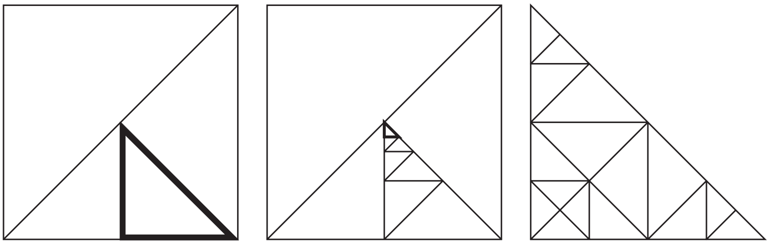


Fig. 9. 13 simplices whose barriers contain *level-1* simplex R

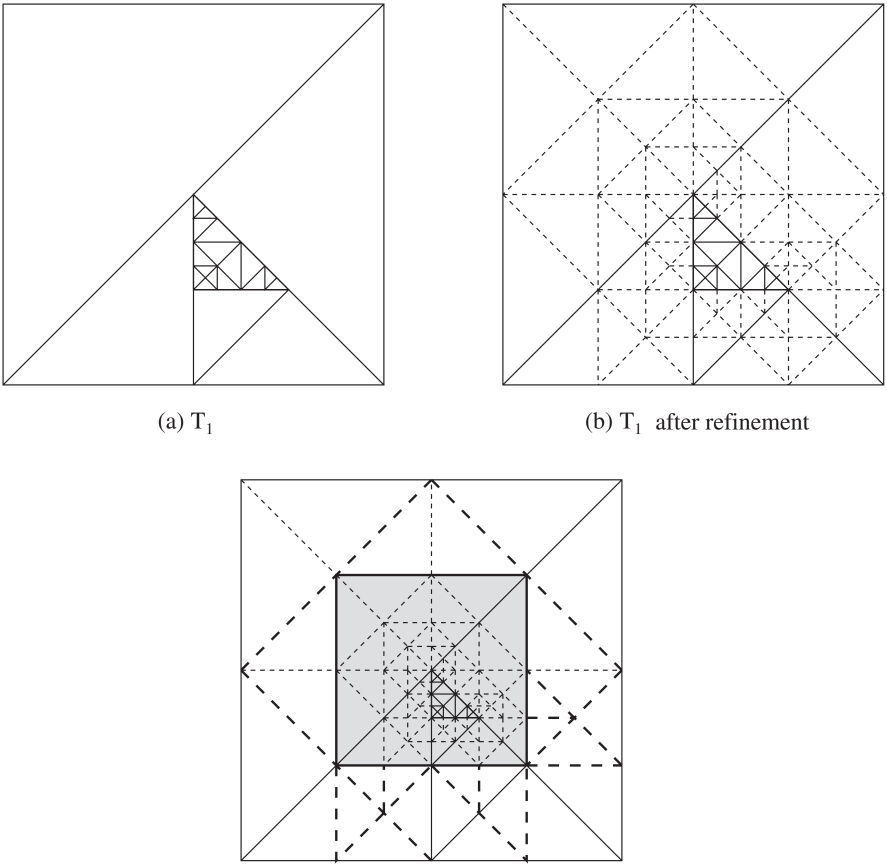
compatibly refined. Let T_i denote the i-th tree in this family of trees. T_i is constructed as follows. We start with the tree shown in Fig. 10(a). Initially, we designate the simplex with thick borders as the *next* simplex to split. Each split generates two new child simplices. After the split we update *next* to be the child simplex which has the central vertex of the subdivision as one of its vertices. To construct T_i, we split the *next* simplex $2i$ times. Figure 10(b) shows such a subdivision after six splits ($i = 3$). To complete the construction of T_i, we replace the *next* simplex with the subdivision shown in Fig. 10(c).

Figure 11 and Fig. 12 show the first three trees of this family. Figure 11(a) and (b) show T_1 before and after compatible refinement, respectively. Splits performed during compatible refinement are depicted with dashed lines. T_1 has 19 internal nodes before refinement, and 97 internal nodes after refinement. Fig. 11(c) shows T_1 and T_2 such that T_1 is within the gray inner square with thick borders. T_2 is defined within the outer square. The thicker dashed lines correspond to the additional splits (i.e., internal nodes generated) due to compatible refinement of T_2.



(a) (b) (c)

Fig. 10. Construction of T_i



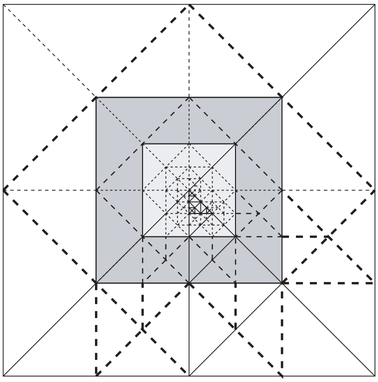
(a) T_1

(b) T_1 after refinement

(c) T_1 after refinement

Fig. 11. T_1 within the inner gray square and T_2 within the outer square

Figure 12 shows T_1, T_2 and T_3 such that T_1 is within the innermost light gray square, T_2 is within the dark gray next outer square and T_3 is defined within the outermost square.



	number of internal nodes	
	before	after refinement
T_1 within light gray square	19	97
T_2 within dark gray square	21	125
T_3 within outermost square	23	153

Fig. 12. T_3

From these first three trees of the sequence we observe a pattern that, T_{i+1} contains two more internal nodes than T_i, and the compatible refinement of T_{i+1} produces twenty six more internal nodes than the compatible refinement of T_i. See Fig. 13 for a depiction of how T_{i+1} is related to T_i. In this figure, the thick solid lines represent the two additional splits in T_{i+1} compared to T_i, and the thick dashed lines (seven of them are on the border of T_i) constitute the twenty six additional splits needed for the compatible refinement of T_{i+1} than were necessary for the compatible refinement of T_i. (The thin solid lines were already accounted for in previous trees of the sequence.) The number of internal nodes before and after the compatible refinement for the first three trees of the family is also given in Fig. 12.

Based on above observations, any tree of this family with n nodes generates a tree with $14n - 169$ nodes after compatible refinement. As n increases, the expansion factor approaches the upper bound of 14.

5 The Expansion Factor in Higher Dimensions

Unlike the 2-dimensional case, we do not have tight bounds on the expansion factor for dimensions 3 and higher. However, we will sketch an upper bound on their size after compatible refinement. In our results for 2-dimensional trees described in previous sections, we have chosen the minimum barrier for a simplex in order to prove a tight upper bound. Our approach will be to generate a larger barrier, but one that is easier to analyze. Such a barrier for a *level-0* simplex in the 2-dimensional case is shown in Fig. 14. This barrier contains 18 simplices all of the same depth. (All of them are *level-0* simplices.)

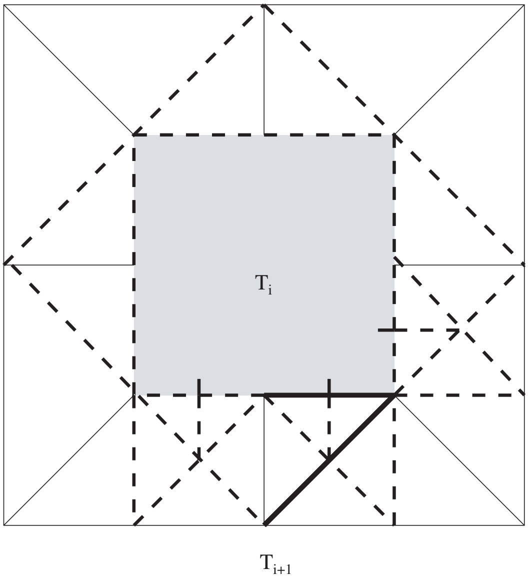


Fig. 13. T_i and T_{i+1}

Consequently, a *level-0* simplex could be in the barriers of 17 other simplices, meaning that if it were split, it could be responsible for 17 other element splits. Thus, compatibly refining a 2-dimensional simplex decomposition tree could increase the number of nodes by at most a factor of 18.

If we analyze the construction of the barrier, we note that the square containing S in Fig. 14 is surrounded by 8 squares and each square contains 2 *level-0* simplices. We can generalize such a barrier to d-dimensional case as follows. Consider a *level-0* simplex S within a d-dimensional hypercube H. Surround H by $3^d - 1$ hypercubes such that each face of H is shared by a neighbor hypercube. Each of these neighbor hypercubes contains $d!$ simplices. This results in a barrier containing $3^d d!$ *level-0* simplices including S. Consequently, a *level-0* simplex could be in the barriers of $(3^d d!) - 1$ simplices, therefore, if it is split, it could be responsible for $(3^d d!) - 1$ other element splits. Thus, a d-dimensional simplex decomposition tree could grow by at most a factor of $3^d d!$ when compatibly refined.

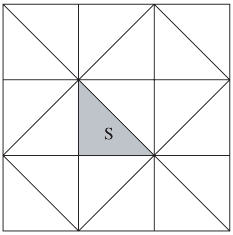


Fig. 14. A naive barrier for a *level-0* simplex in 2-dimensions

However, above analysis is only for *level-0* simplices. In the d-dimensional case, we have d canonical simplices to be considered. We can construct the barrier for a *level-k* simplex S as follows. S is contained in a d-dimensional hypercube H. Surround H by $3^d - 1$ hypercubes as before, but consider that each hypercube is subdivided into $d! \cdot 2^k$ *level-k* simplices instead. This results in a barrier containing $3^d d! \cdot 2^k$ *level-k* simplices including S. Consequently, a *level-k* simplex could be in the barriers of $3^d d! \cdot 2^k - 1$ simplices. Since k could be $d - 1$ at most, an d-dimensional simplex decomposition tree could grow by at most a factor of $3^d d! \cdot 2^{d-1}$ when compatibly refined.

6 Conclusion

We have shown that when compatibly refined the size of a 2-dimensional simplex decomposition tree grows at most by a factor of 14 and this is tight. This is a worst-case bound, however, and our preliminary experiments on randomly generated sd-trees suggest that, in practice the expansion factor is much smaller. For example, over a 100 randomly generated 2-dimensional sd-trees of maximum height 32, the average expansion factor was found to be only 4.7, and the maximum expansion factor was found to be 5.9. For 3-dimensional sd-trees of maximum height 32, the average was 31.2 and the maximum was 36.1. For 4-dimensional sd-trees of maximum height 32, the average was 227.6 and the maximum was 244.6.

For dimensions higher than 2, we have sketched an upper bound, but a more complete analysis would be needed to prove tight bounds. Since a d-dimensional sd-tree contains simplices from d different similarity classes, and for a complete analysis each canonical simplex has to be considered separately, it would therefore be much more challenging to prove tight upper bounds for general dimensions.

References

1. E. Allgower and K. Georg. Generation of triangulations by reflection. *Utilitas Mathematica*, 16:123–129, 1979.
2. F. B. Atalay and D. M. Mount. Pointerless implementation of hierarchical simplicial meshes and efficient neighbor finding in arbitrary dimensions. In *Proc. International Meshing Roundtable (IMR 2004)*, pages 15–26, 2004.
3. M. Duchaineau, M. Wolinsky, D.E. Sigeti, M.C. Miller, C. Aldrich, and M.B. Mineev-Weinstein. Roaming terain: Real-time optimally adapting meshes. In *Proc. IEEE Visualization'97*, pages 81–88, 1997.
4. H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1987.
5. W. Evans, D. Kirkpatrick, and G. Townsend. Right-triangulated irregular networks. *Algorithmica.*, 30(2):264–286, 2001.
6. T. Gerstner. Multiresolution visualization and compression of global topographic data. *GeoInformatica*, 7(1):7–32, 2003.
7. T. Gerstner and M. Rumpf. Multiresolutional parallel isosurface exraction based on tetrahedral bisection. In *Proc. Symp. Volume Visualization*, 1999.
8. B. Von Herzen and A. Barr. Accurate triangulations of deformed intersecting surfaces. *Computer Graphics*, 21(4):103–110, 1987.

9. P. Lindstrom, D. Koller, W. Ribarsky, L.F. Hodges, N. Faust, and G.A. Turner. Real-time, continuous level of detail rendering of height fields. In *Proc. of SIGGRAPH 96*, pages 109–118, 1996.
10. J. M. Maubach. Local bisection refinement for N-simplicial grids generated by reflection. *SIAM J. Sci. Stat. Comput.*, 16:210–227, 1995.
11. D. Moore. The cost of balancing generalized quadtrees. In *Proc. ACM Solid Modeling*, 1995.
12. R. Pajarola. Large scale terrain visualization using the restricted quadtree triangulation. In *Proc. IEEE Visualization'98*, pages 19–26, 1998.
13. A. Weiser. *Local-Mesh, Local-Order, Adaptive Finite Element Methods with a Posteriori Error Estimators for Elliptic Partial Differential Equations.* PhD thesis, Yale University, 1981.
14. Y. Zhou, B. Chen, and A. Kaufman. Multiresolution tetrahedral framework for visualizing regular volume data. In *Proc. IEEE Visualization'97*, pages 135–142, 1997.