Amortized Analysis of Smooth Box Subdivisions in All Dimensions

Huck Bennett * Chee Yap †

January 27, 2014

Abstract

Quadtrees are a well-known data structure for representing geometric data in the plane, and naturally generalize to higher dimensions. A basic operation is to expand the tree by splitting any given leaf. A quadtree is *smooth* if any two adjacent leaf boxes differ by at most one in height. In this paper, we analyze quadtrees that maintain smoothness with each split operation. Our main result shows that the smooth-split operation has an amortized cost of $O(1)$ time for quadtrees of any fixed dimension D. We also present examples demonstrating the ineffectiveness of related models in order to motivate our approach, and prove a related lower bound.

1 Introduction

Quadtrees [dBCvKO08, FB74, Sam90b] are a well-known data structure for representing geometric data in two dimensions. In this case there exists a natural one-to-one correspondence between quadtree nodes v and boxes B in the underlying subdivision which allows us to refer to boxes and nodes interchangeably. Here we consider the extension to an aligned subdivision of a D-dimensional box in which an internal node is a box with 2^D congruent subboxes. We refer the reader to Chapter 14 in [dBCvKO08] whose nomenclature we largely follow.

Two boxes (or nodes in a quadtree) are *adjacent* if the boxes share a $(D-1)$-dimensional facet, but have disjoint interiors. The *neighbors* of a box B are those leaf boxes adjacent to B. We follow [Moo92] in calling a quadtree *smooth* if any two adjacent leaf boxes differ by at most one in height. Other sources use the term *balanced* to refer to this condition, which we avoid in order to avoid conflation with the standard meaning of balanced trees in computer science.

A basic operation is a *split* of a leaf box B, written `split`(B). This divides B into 2^D congruent subboxes which become its children (B is no longer a leaf). A `split` operation is a useful abstraction of many common operations performed on quadtrees including point insertion and mesh refinement.

Our quadtrees support two operations: `ssplit` and `neighbor_query`. Define a *smooth split* operation or `ssplit`(B) to be `split`(B) followed by a `smooth` of the resulting tree. A neighbor query `neighbor_query`(B, d) returns a neighbor of B in direction d at least as large as B but of minimal size. The neighbor returned by `neighbor_query`(B, d) is unique if it exists (it may not if B is on the boundary of the subdivision). Neighbor queries are useful in quadtree applications such as motion planning [WCY13].

The motivation for using smooth quadtrees comes from multiple domains including good mesh generation [dBCvKO08, BEG94] and motion planning [WCY13]. One advantage is that a given

*Department of Computer Science, Courant Institute, New York University. hbennett@cs.nyu.edu

†Department of Computer Science, Courant Institute, New York University. yap@cs.nyu.edu

Algorithm 1: Smooth Split (`ssplit`)

```
Input: Smooth quadtree T, Leaf v ∈ T to split
Output: Smooth quadtree T′
split(v)
foreach v′ ∈ v.neighbors \ v.siblings do
    if v′.depth < v.depth then
        ssplit(v′)
    end
end
```

unsplit box has $O(1)$ neighbors, meaning that by associating a constant number of neighbor pointers with each box we can perform `neighbor_query` operations in $O(1)$ time. This contrasts with the $O(h)$ time operation in basic quadtrees that involves traversing to the nearest common ancestor.

1.1 Our Results

In this paper we present and analyze a quadtree variant that we call a *dynamically smoothed quadtree* that maintains smoothness as an invariant between splits, allowing for performing the `neighbor_query` operation in $O(1)$ time. This variant has been proposed before such as in Exercise 14.8 in [dBCvKO08], although to the best of our knowledge bounds on the complexity of smooth splits have never been studied rigorously.

The primary contribution of this paper is a proof that amortized $O(1)$ additional split operations is sufficient for each smooth split operation in quadtrees of any fixed dimension. We prove this result as Theorem 22 in section 3.5 in Appendix B, and give a more elementary (but similar) proof of the 2-dimensional case section 2.2. More formally Theorem 22 shows,

THEOREM 1. Starting from an initially trivial subdivision consisting of one box, the total cost of any sequence of smooth splits $\texttt{ssplit}(B_1), \ldots, \texttt{ssplit}(B_n)$ is $O(n)$. Thus the amortized cost of a smooth split is $O(1)$.

Additionally, we give counterexamples motivating our data structure and analysis. We first show that without smoothing we cannot achieve an amortized $O(1)$ cost for both splits and neighbor queries. Second, we address a claim made in [LSS13b] that smoothness can be restored in worst-case $O(1)$ time in a related quadtree model in the appendix.

We also address the question of the constant in the $O(1)$ amortized bound on the number of splits per smooth split, and particularly the dependence on dimension (we generally consider the dimension to be fixed). In addition to the $O(2^D(D+1)!)$ upper bound that we get from the proof of Theorem 22, we also prove a lower bound of $\Omega(2^D(D+1))$ in Appendix C, Theorem 28.

1.2 Data Structure

Table 1 compares the cost of standard operations on quadtrees. We use n to denote the number of nodes in and h the height of a quadtree T. We achieve improvements to the `neighbor_query` and `smooth` operations at the cost of `split` requiring amortized rather than worst-case $O(1)$ time. The $O(1)$ time bounds for the `ssplit` and `split` operations are for the local operations – when

	Smooth quadtrees	Basic quadtrees
`neighbor_query`	$O(1)$	$O(h)$
`ssplit/split`	Amortized $O(1)$	$O(1)$
`smooth`	(Maintained as invariant)	$O((h+1)n)$
Space used	$O(n)$	$O(n)$

Table 1: Comparison of operations with basic quadtrees in fixed dimension. All costs are worst-case except for splitting smooth quadtrees.

the algorithm already has a pointer to the box it wishes to split such as the scenario described in [WCY13]. Traversing from the root to obtain this pointer takes time $O(h)$.

Algorithm 1 shows the simplicity of the algorithm for performing smooth splits: simply recursively check whether any neighbors of a node need to be split to regain smoothness. Nevertheless, the analysis of the algorithm is subtle.

1.3 Related Work

The following theorem is a well-known result, saying that an arbitrary quadtree can be smoothed using $O(n)$ splits and $O((h+1)n)$ time:

Fact 1 (Theorem 14.4 in [dBCvKO08], Theorem 3 in [Moo95]). Let T be a quadtree with n nodes and of height h. Then the smooth version of T has $O(n)$ nodes and can be constructed in $O((h+1)n)$ time.

Fact 1 gives a bound for *monolithic* tree smoothing, the operation that we call `smooth` in Table 1. It says that given an arbitrary quadtree we can smooth it all at once in $O(n)$ time. Here we study *dynamic* tree smoothing in which we smooth the tree after each split, instead of performing an arbitrary number of splits before smoothing.

Intuitively a single splitting operation does not unsmooth a quadtree much, so only a few additional splits should be required to resmooth a tree after one split. To show this formally one might try applying the analysis given by Fact 1 to a sequence of smooth splits $\texttt{ssplit}(B_1), \ldots, \texttt{ssplit}(B_n)$. However that analysis does not consider any measure of how smooth the starting tree is, and only gives a worst-case linear time bound of $O(i)$ for smoothing after the ith split in a sequence $\texttt{split}(B_1), \ldots \texttt{split}(B_n)$ where $\texttt{split}(B_1)$ is applied to the root. This analysis shows that a sequence of smooth splits $\texttt{ssplit}(B_1), \ldots, \texttt{ssplit}(B_n)$ requires $\sum_{i=1}^{n} O(i) = O(n^2)$ time for an amortized bound of $O(n)$ which is then no better than the worst-case bound.

We note that Theorem 1 proves a stronger bound than Fact 1 on the number of splits required to smooth a quadtree. This is because Theorem 1 shows that only $O(n)$ additional smooth splits are needed to maintain smoothness in any sequence of n splits. Therefore, after n (non-smooth) splits, we could still perform these $O(n)$ smooth splits to achieve smoothness.

1.3.1 Other Results

In recent work Löffler et al. [LSS13b] recognize that maintaining smoothness "could cause a linear 'cascade' of cells needing to be split." This cascading behavior – what we define formally in terms of *forcing chains* – is the focus of our analysis and main result. They claim an $O(1)$ worst-case algorithm for performing smooth splits in a related quadtree model, but there are problems with their presented algorithm which we address in this paper, and which make their result unsuitable for our setting.

Moore [Moo92, Moo95] proves that "monolithic" smoothing of arbitrary quadtrees requires $O(n)$ splits. Although this result seems to have been known earlier, in [Moo95] Moore reproves this result in basic quadtrees using a gadget called a "barrier", and then extends the result to generalizations of quadtrees including triangular quadtrees, higher degree quadtrees, and higher dimensional quadtrees. Fact 1 states this result in the standard setting.

In [dBRS12], de Berg et al. study *refinement* of compressed quadtrees. They consider a refinement T_1 of a quadtree T_0 to be extension of T_0 in which all boxes that were in T_0 have $O(1)$ neighbors in T_1. This is a relaxation of the notion of balancing both in terms of the precise number of neighbors that a box may have (which is simply assumed to be bounded, but not by a particular constant) and in the sense that boxes in T_1 need not be smooth with respect to each other. The authors prove that a refinement of a compressed quadtree may be performed in $O(n)$ time, where n is the size of the quadtree. This result has a similar flavor to the well-known "monolithic" balancing result described in Fact 1.

Amortized analysis of quadtree operations has appeared in previous work. Park and Mount [PM12] introduce the *splay quadtree*, in which they use amortized analysis to analyze the cost of a sequence of data accesses in a quadtree whose smoothness is dynamically updated using rotations in a similar manner to standard splay trees. Overmars and van Leeuwen [OvL82] analyze dynamic quadtrees, studying the amortized (what they call average-case) cost of insertions into quadtrees.

Recently Sheehy [She] proposed extending results in his previous work on optimal mesh sizes [She12] to prove the efficient balancing results presented in this paper. Essential future work involves studying the continuous techniques used in this approach, and determining whether it is both viable and leads to better bounds than those given by the combinatorial approach.

1.4 Motivation for Approach

The motivation for studying the quadtree model presented in this paper comes from the ineffectiveness of other natural models to support both efficient `neighbor_query` and `split` operations. We make this notion rigorous by examining two attempts to achieve this, and show that they fail in our setting. First, we analyze what happens if we use our model but without smoothing. Additionally in the appendix we discuss a paper [LSS13b] that claims a related result.

1.4.1 Neighbor Pointers without Smoothing

Suppose that we maintain neighbor pointers to minimal neighbors of equal or greater size in an unsmoothed subdivision. The following result gives an amortized $O(\log n)$ lower bound on the complexity of a split in this model:

LEMMA 2. Let B_1 denote the root box. In the worst case, a sequence of n splits $\texttt{split}(B_1), \ldots, \texttt{split}(B_n)$ has complexity $\Omega(n \log n)$.

Proof. We refer to the setup shown in Figure 1, where the boxes are first subdivided as shown on the left, and then further subdivided as shown on the right where the boxes on the boundary of the halves are at depth $k+1$ in the quadtree.

After an initial split, each half requires $\sum_{i=1}^{k} 2^i = 2^{k+1} - 2$ additional splits. The total number of splits is therefore $n = 1 + 2(2^{k+1} - 2) = 2^{k+2} - 3 \Rightarrow n = \Theta(2^k)$.

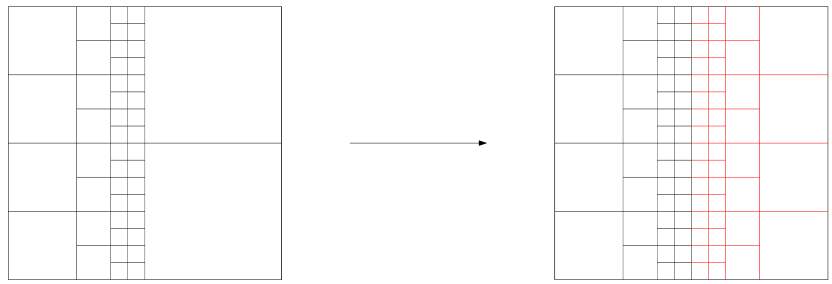


Figure 1: A sequence of splits leading to an unsmooth subdivision (left) and a sequence of matching splits that requires amortized $\log n$ pointer updates between boxes on opposite sides of the dotted center line per split (shown in red, right)

For the lower bound we consider only updates to the pointers straddling the vertical center line in the second splitting phase, as shown by the red boxes. For each splitting level i, we must update 2^{k-i} pointers in each of 2^i boxes. We therefore must update $\sum_{i=1}^{k} 2^i 2^{k-i} = \sum_{i=1}^{k} 2^k = k2^k = \Theta(n \log n)$ pointers. $\square$

The failure of this attempt and the strategy for worst-case $O(1)$ balancing given in the appendix to give efficient, correct algorithms for both `split` and `neighbor_query` operations provides evidence that a balancing algorithm achieving worst-case $O(1)$ time per split would have to be more sophisticated and non-localized.

2 2-Dimensional Case

We start with an elementary, self-contained proof of Theorem 1 for 2-dimensional quadtrees that develops most of the essential ideas for the d-dimensional case.

2.1 Definitions

Suppose that a box B is adjacent to a box B' and $B.\mathtt{depth} > B'.\mathtt{depth}$. In that case, we say that B *forces* B' or $B \Longrightarrow B'$. The forcing terminology comes from our main application, the analysis of smoothing: suppose B, B' belongs to a subdivision S. If we split B, then we are forced to split B' and possibly other boxes in order to smooth the resulting subdivision. More precisely, let $B.\mathtt{depth} - B'.\mathtt{depth} = k \geq 1$. Then we must split B' and recursively split exactly $k-1$ proper descendants of B' in order to maintain smoothness in S. Of course if S was originally smooth, then no child of B' needs to be further split. We will mostly deal with the case where S is originally smooth and in this case we always have $k = 1$.

A *forcing chain* $B_1 \Longrightarrow B_2 \cdots \Longrightarrow B_n$ is a sequence of boxes $B_1, \ldots, B_n$ such that $B_i \Longrightarrow B_{i+1}$ for every $i \in [n-1]$. Call B_1 the *head* of this chain.

We write $B \overset{d}{\Longrightarrow} B'$ if $B \Longrightarrow B'$ and B' is a d-thern neighbor of B. Here a direction d is specified by a standard normal unit vector u_i or its negation $-u_i$. We write $* \Longrightarrow B$ if there exists B' such that $B' \Longrightarrow B$, and similarly write $B \Longrightarrow *$ if there exists B' such that $B \Longrightarrow B'$. Lastly, we denote the parent of a box B as $p(B)$, and the kth ancestor of a box as $p^k(B)$.

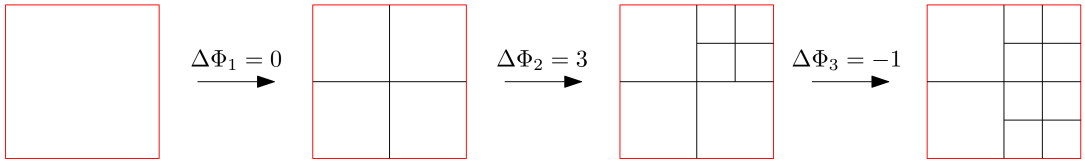


Figure 2: Example of the three cases presented in Equation 3. We consider the change each split has on $\Phi(v)$, where v corresponds to the outer red box in each case.

2.1.1 Potential Function

We define the following potential function for a node $v \in T$:

$$\Phi(v) = \begin{cases} 0 & \text{if no children of } v \text{ have been split} \\ \#\text{ of unsplit children of } v & \text{otherwise} \end{cases} \tag{1}$$

We also extend this definition to give a potential function for the quadtree:

$$\Phi(T) = \sum_{v \in T} \Phi(v) \tag{2}$$

We note that $\Phi(v) = 0$ if either all or none of the children of v are split. Furthermore, if v is itself a leaf then $\Phi(v) = 0$ vacuously. It follows that only parents of leaf nodes have non-zero contribution to the potential $\Phi(T)$. Furthermore, splitting a box changes the potential of at most one node (its parent).

Let T be a quadtree, and T' be the quadtree resulting from splitting a leaf v. Splitting v does not change the potential of v, but changes the potential of the parent $p(v)$ of v by either 3 if $p(v)$ had no split children or -1 if $p(v)$ had other split children. A leaf v always has a parent except in the degenerate case where v is the root of the tree. We then get the following:

$$\Delta\Phi = \Phi(T') - \Phi(T) = \begin{cases} 0 & \text{If } v \text{ is the root of } T \\ 3 & \text{If } v \text{ has no split siblings} \\ -1 & \text{If } v \text{ has a split sibling} \end{cases} \tag{3}$$

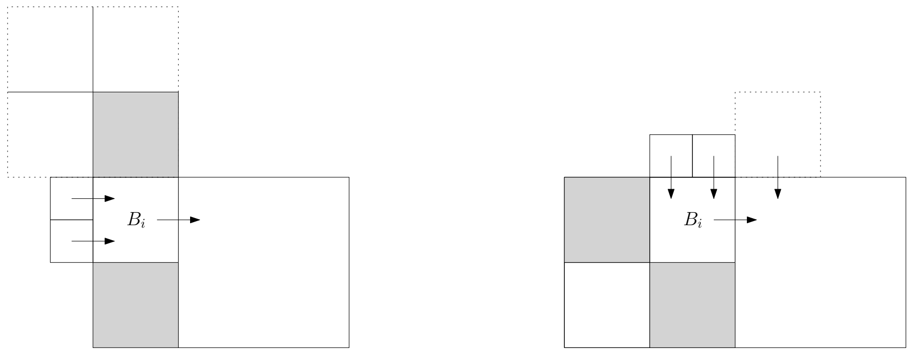
Because the first case only occurs on the first split, in which case only a single box splits and $\Delta\Phi = 0$, it suffices to consider the last two cases for our analysis. Note that we may write $\exists v'\ p(v) = p^2(v')$ to formalize "v has a split sibling."

2.2 Lemmas

The following sequence of lemmas leads to the proof of Theorem 1.

Lemma 3. There are at most two chains caused by splitting a box B.

Proof. We get an immediate upper bound of 2 on the number of chains that can be headed by a box B_1 since a box will never force in the direction of an adjacent sibling of which every box has two. Furthermore, we show that $* \Longrightarrow B_i$ implies that there exists at most one box B_{i+1} such that $B_i \Longrightarrow B_{i+1}$. Since the head B_1 of a splitting chain B_i is the only box in a splitting chain which may



Case I Case II

Figure 3: Case I: $p(B_{i-1})$ is a sibling of B_i and Case II: $p(B_{i-1})$ is not a sibling of B_i. Neighbors of B other than B_{i-1} which must be split to at least the level of B_i are colored gray. Boxes which necessarily exist assuming that the subdivision is smooth are outlined with dotted lines.

not be forced itself, this will imply that there are at most two splitting chains caused by splitting a box B_1.

We immediately have $*\overset{d}{\Longrightarrow}B_i \Rightarrow B_i \overset{-d}{\not\Longrightarrow}*$ since a box cannot simulatenousely have smaller and larger neighbors in the same direction. There are then 3 other directions B_i may force in. We consider two cases, as shown in Figure 3:

- Case I, ($p(B_{i-1})$ is a sibling of B_i): The dotted outer box must be split in order to be smooth with respect to B_{i-1}. A box in one of the remaining directions is a sibling B_i, while a box in another is a child of the dotted box (both shown in gray). These must both be split to at least the level of B_i, leaving a single possibility for B_{i+1}.

- Case II, ($p(B_{i-1})$ is not a sibling of B_i): Boxes in two of the possible three remaining directions are siblings of B_i (shown in gray) and must therefore be split to at least the depth of B_i, leaving a single possibility for B_{i+1}.

□

Lemma 4. Assume a smooth quadtree in which $*\overset{d}{\Longrightarrow}B_1\Longrightarrow B_2$ for some d. Then $*\overset{d}{\Longrightarrow}B_2$. In other words, if B_1 is d-forced and B_1 forces B_2, then B_2 is d-forced (not necessarily by B_1).

Proof. We again refer to Figure 3, and evaluate each case separately:

- Case I, ($p(B_{i-1}$ is a sibling of B_i): Here $B_{i-1}\overset{d}{\Longrightarrow}B_i\overset{d}{\Longrightarrow}B_{i+1}$ so the claim trivially holds.

- Case II, ($p(B_{i-1}$ is not a sibling of B_i): We have assumed that $B_{i-1}\overset{d}{\Longrightarrow}B_i\overset{d'}{\Longrightarrow}B_{i+1}$ where $d \neq d'$. In this case, either B_{i-1} or its d'-thern sibling must have the dotted box (call it B_i') as its d'-thern neighbor. However, the dotted box must be a $-d$-thern neighbor of B_{i+1}, but of greater depth. Therefore $B_i'\overset{d}{\Longrightarrow}B_{i+1}$ and the claim holds.

□

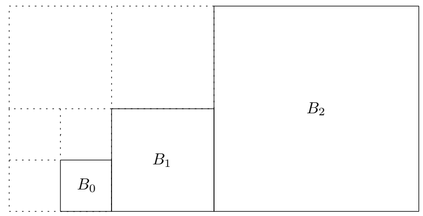


Figure 4: Assuming $B_0 \stackrel{d}{\Longrightarrow} B_1 \stackrel{d}{\Longrightarrow} B2$ the dotted boxes must exist. Therefore the parent of B_0 must be split and a sibling of B_1.

By transitivity we conclude:

Corollary 5. If $B_1 \stackrel{d}{\Longrightarrow} B_2 \Longrightarrow \cdots \Longrightarrow B_n$ then B_i is d-forced for $i \geq 2$.

The following additional corollary says that a split chain may go in at most two directions:

Corollary 6. Given a split chain $B_1 \stackrel{d_1}{\Longrightarrow} B_2 \stackrel{d_2}{\Longrightarrow} \cdots \stackrel{d_{n-1}}{\Longrightarrow} B_n$, we have that $|\{d_i : i \in [n-1]\}| \leq 2$.

Proof. We have that $* \stackrel{d}{\Longrightarrow} B \Rightarrow B \stackrel{d}{\not\Longrightarrow} B$, meaning that a box may force in at most two directions. However, Lemma 4 shows that $* \stackrel{d}{\Longrightarrow} B_i \Longrightarrow B_{i+1} \Rightarrow * \stackrel{d}{\Longrightarrow} B_{i+1}$, meaning that a box in a forcing chain is always forced in all of the directions as its predecessors. Therefore, if B_i is forced in two directions then B_j is also forced in the same two directions for all $j > i$, and cannot force in any additional directions.

□

Lemma 7. If for some boxes B_1, B_2, B_3 we have $B_1 \stackrel{d}{\Longrightarrow} B_2 \stackrel{d}{\Longrightarrow} B_3$ then B_2 has a split sibling.

Proof. Figure 4 shows the idea behind Lemma 7. Because $B_1 \stackrel{d}{\Longrightarrow} B_2$ we have that B_1 is a d-thern child of its parent, meaning that its $(d+2)$-thern neighbor of the same size is also its sibling.

Furthermore, because $B_0 \stackrel{d}{\Longrightarrow} B_1$, we have that B_0 is a $(d+2)$-thern neighbor of B_1. Because B_0 has side length exactly half that of B_1, it follows that $p(B_0)$ and B_1 are siblings. Furthermore, because $p(B_0)$ has children it is clearly split. □

Lemma 8 (Main Lemma). At most 3 nodes in a split chain $B_1 \stackrel{d_1}{\Longrightarrow} B_2 \stackrel{d_2}{\Longrightarrow} \cdots \stackrel{d_{m-1}}{\Longrightarrow} B_m$ have no split siblings.

Proof. We combine Corollaries 5 and 6 with Lemma 7 to prove the Main Lemma. If $B_{i-1} \stackrel{d}{\Longrightarrow} B_i \stackrel{d}{\Longrightarrow} B_{i+1}$ then B_i has a split sibling by Lemma 7.

We characterize which boxes may not have this property, showing that B_1, B_i, B_m may not have split siblings. Here B_i is the first box such that $d_i \neq d_1$ in $B_i \stackrel{d_i}{\Longrightarrow} B_{i+1}$.

Box B_1 need not be forced from any direction, and B_n need not force in any direction, so Lemma 7 does not apply. Furthermore if the chain goes in two directions B_i exists and is d_1-forced, but is not d_1 forcing so again again 7 does not apply.

To see that all other boxes must have split siblings we consider two cases:

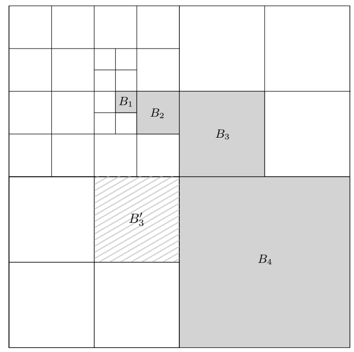


Figure 5: A split chain $B_1 \xRightarrow{d} B_2 \xRightarrow{d} B_3 \xRightarrow{d'} B_4$ of four nodes illustrating Lemma 8. Note that B_1 and B_3 have no split siblings, and B_4 may also be the northwest child of its parent, and therefore also may not have any split siblings. Box B_2, on the other hand, satisfies Lemma 7. Furthermore, B_4 is d'-forced although not by B_3.

- Case I, $(1 < j < i)$: We have that $B_{j-1} \xRightarrow{d_1} B_j \xRightarrow{d_1} B_{j+1}$ by assumption that $d_j = d_1$ for all $j < i$. Therefore Lemma 7 applies to B_j.

- Case II, $(i < j < n)$: We have that $B_j \xRightarrow{d_j} B_{j+1}$ where $d_j \in \{d_1, d_i\}$ since by Corollary 6 a split chain may go in at most two directions. Furthermore by Corollary 5 $* \xRightarrow{d_1} B_j$ and $* \xRightarrow{d_i} B_j$ meaning that either $* \xRightarrow{d_1} B_j \xRightarrow{d_1} B_{j+1}$ or $* \xRightarrow{d_i} B_j \xRightarrow{d_i} B_{j+1}$. In either case Lemma 7 applies to B_j.

□

We now give the proof of Theorem 1 using the main lemma:

Proof of Theorem 1 in 2 dimensions. We fix the cost of a split $\text{cost}_i = \texttt{split}(B_i)$ as 1. To prove a constant amortization bound, we need to show that there exists $\text{charge}_i = O(1)$ such that

$$\text{charge}_i \geq \text{cost}_i + \Delta\Phi_i \tag{4}$$

holds for each smooth split operation $\texttt{ssplit}(B_i)$. By equation 3 we have

$$\text{cost}_i + \Delta\Phi_i = \begin{cases} 4 & \text{if } v_i \text{ has no split siblings} \\ 0 & \text{if } v_i \text{ has a split sibling} \end{cases} \tag{5}$$

By Lemma 8 at most three boxes per split chain have no split siblings. Furthermore, by Lemma 3 a smooth split of a box B_0 causes at most two split chains. It therefore suffices to charge $4 \cdot 3 \cdot 2 = 24 = O(1)$ per smooth split operation. □

Remark 1. Although 24 is the best constant we can get for the upper bound on the amortized cost of a smooth split we conjecture that it is not tight.

3 General Case

In order to handle arbitrary dimensions, we will need to develop some notation and concepts. All missing proofs are in the Appendix.

3.1 Basic Terminology.

We give a brief summary of the concepts needed. The appendix contains more careful definitions. Here we rely on the intuitions that are well-known from quadtrees. We consider subdivision of the standard cube $[-1,1]^D$ in $D \geq 1$ dimensions. A subdivision tree $\mathbb{T}$ is a finite tree rooted at $[-1,1]^D$ whose nodes are subboxes of $[-1,1]^D$, and where each internal node has 2^D congruent children. The set leaves of $\mathbb{T}$ constitute a *subdivision* of $[-1,1]^D$. Nodes of $\mathbb{T}$ are also called "aligned boxes", and every aligned box has a natural depth. Conversely, given any subdivision $\mathbb{S}$ of aligned boxes, there is a unique subdivision tree $\mathbb{T}(\mathbb{S})$.

Let $j = -1, 0, 1, \ldots, D$. Two boxes B, B' are *j-adjacent* if $B \cap B'$ is a j-dimensional box. Four special cases are noteworthy:

- If they are D-adjacent, we say B and B' *overlap*.
- If they are $(D-1)$-adjacent, we say they are *neighbors*.
- 0-adjacency means they share a common corner only.
- (-1)-adjacency means the boxes are disjoint.

Fact 2. Let B, B' be overlapping aligned boxes. Then either $B \subseteq B'$ or $B' \subseteq B$.

By an *indicator* we mean an element $d \in \{1, 0, -1\}^D$. If d has exactly one non-zero component, we call it a *direction indicator*; if it has no zero components, we call it a *child indicator* (we do not need child indicators in this paper, but it will be useful in coding these algorithms). Two directions d and d' are *opposite* if $d = -d'$, and *adjacent* if $d \neq d'$ and they are not opposite. If B is a child of B', then we write $B \prec B'$, and write $\mathtt{p}(B) = B'$. E.g., $\mathtt{p}^2(B)$ is the grandparent of B.

If B and B' are $(D-1)$-adjacent, there is a unique direction indicator d such that B' is *adjacent to B in direction d*, which we denote by $B \xrightarrow{d} B'$. Moreover, $B \xrightarrow{d} B'$ iff $B' \xrightarrow{-d} B$. See Appendix for the formal definition of this relation.

Given a box B, we can project and co-project it in one of D directions: let $i \in \{1, \ldots, D\}$.

- (Projection) $\mathtt{Proj}_i(B) := \prod_{j=1, j\neq i}^{D} I_j$ be a $(D-1)$ dimensional box.
- (Co-Projection) $\mathtt{Coproj}_i(B) := I_i$ denote the ith interval of $B = \prod_{j=1}^{D} I_j$.

3.2 Forcing Chains

Let S be a subdivision of the standard cube $[-1,1]^D$. We say S is *smooth* if any two neighboring boxes B, B' in S differ in depth by at most 1. We are interested in maintaining smooth subdivisions. More precisely, if S is smooth, and we split a box in S, there is minimal set of additional boxes in S that must be split in order to maintain smoothness.

If $B \xrightarrow{d} B'$, and the $depth(B) > depth(B')$ then we denote this relationship by

$$B \overset{d}{\Longrightarrow} B'.$$

We say B *d-forces* B' (or simply, B forces B'). Intuitively it means that if B, B' are boxes in a subdivision and we split B, then we are forced to split B' if we want to make the subdivision smooth. Because we maintain smoothness as an invariant $B \Longrightarrow B'$ means $depth(B) = 1 + depth(B')$.

A sequence of such forcing relations

$$C : B_0 \overset{d_1}{\Longrightarrow} B_1 \overset{d_2}{\Longrightarrow} B_2 \cdots \overset{d_k}{\Longrightarrow} B_k \tag{6}$$

is called a *chain* with k *links*. The set $\{d_1, \ldots, d_k\}$ are the directions of C; we say C is *monotone* if its direction set does not contain any pair of opposite directions.

The following lemma follows from the definition of forcing:

Lemma 9 (Forcing). The forcing relationship $B \overset{d}{\Longrightarrow} B'$ is equivalent to the following two conditions:

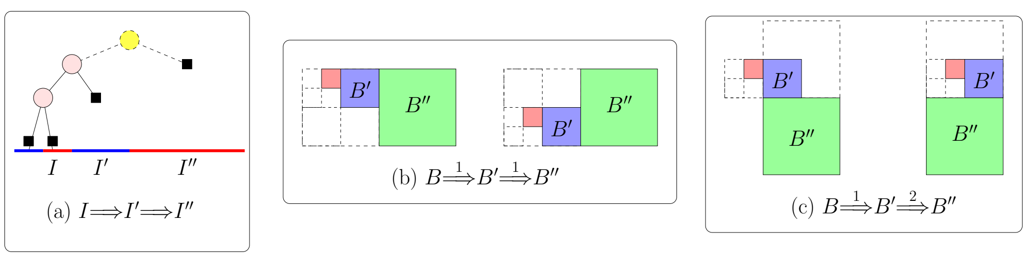
(i) $\texttt{Proj}_d(B) \prec \texttt{Proj}_d(B')$

(ii) $\texttt{Coproj}_d(B) \Longrightarrow \texttt{Coproj}_d(B')$

Note that conditions (i) and (ii) refer to forcing and child relationships in dimensions $D-1$ and 1, respectively.

3.3 Analysis of 2-Link Chains

In this part, we consider chains with 2-links: $B \Longrightarrow B' \Longrightarrow B''$. There are two separate phenomena to understand. The first phenomenon already arise in one dimension ($D = 1$):



(a) $I \Longrightarrow I' \Longrightarrow I''$

(b) $B \overset{1}{\Longrightarrow} B' \overset{1}{\Longrightarrow} B''$

(c) $B \overset{1}{\Longrightarrow} B' \overset{2}{\Longrightarrow} B''$

Figure 6: Analysis of 2-Link Chains

Lemma 10 (Single Direction). Suppose $I \Longrightarrow I' \Longrightarrow I''$ holds for intervals in a smooth subdivision. Then $\texttt{p}^2(I) = \texttt{p}(I')$.

We omit the easy proof, as illustrated by Figure 6(a). Note that we do not claim that $\texttt{p}^3(I) = \texttt{p}(I'')$ (this possibility is suggested by Figure 6(a), but it is not necessarily the case).

It is useful to understand the idiom "$\texttt{p}^2(B) = \texttt{p}(B')$" as telling us that $\texttt{p}(B)$ and B' are siblings.

We show that this works in higher dimensions as well, but we now need an addition condition. When $D = 1$, the fact that $I \Longrightarrow I' \Longrightarrow I''$ implies that there is a direction d such that $I \overset{d}{\Longrightarrow} I' \overset{d}{\Longrightarrow} I''$. In higher dimensions, we must explicitly specify this requirement.

See Figure 6(b) which illustrates two cases in $D = 2$.

Theorem 11 (Single Direction). Suppose $B \overset{d}{\Longrightarrow} B' \overset{d}{\Longrightarrow} B''$ holds for boxes in a smooth subdivision. Then $\texttt{p}^2(B) = \texttt{p}(B')$.

The second phenomenon arises for $D \geq 2$. Consider the chain

$$B \overset{d}{\Longrightarrow} B' \overset{d'}{\Longrightarrow} B''$$

where $d \neq d'$. For $D = 2$, we have this lemma:

Lemma 12 (Two Directions). Consider boxes in a smooth subdivision of $[-1, 1]^2$. Suppose $B \overset{d}{\Longrightarrow} B' \overset{d'}{\Longrightarrow} B''$ holds where $d \neq d'$. Then $\mathtt{p}^2(B) \neq \mathtt{p}(B')$.

We omit the elementary proof, which is illustrated in Figure 6(c). Two cases are illustrated by the figure: in both cases, we show $B \overset{1}{\Longrightarrow} B' \overset{2}{\Longrightarrow} B''$. In the first case, the subdivision is smooth and $\mathtt{p}^2(B) \neq \mathtt{p}(B')$, confirming our lemma. In the second case, $\mathtt{p}^2(B) = \mathtt{p}(B')$ but the subdivision is not smooth, thus confirming our lemma in the contrapositive.

We extend this to arbitrary dimensions.

Theorem 13 (Two Directions). Consider boxes in a smooth subdivision of $[-1, 1]^D$ $(D \geq 2)$. Suppose $B \overset{d}{\Longrightarrow} B' \overset{d'}{\Longrightarrow} B''$ holds where $d \neq d'$. Then $\mathtt{p}^2(B) \neq \mathtt{p}(B')$.

The next result is a kind of commutative diagram argument. It's proof will depend on the Two Directions result (Theorem 13). As usual, we prove the result in two dimensions first (see Figure 7).

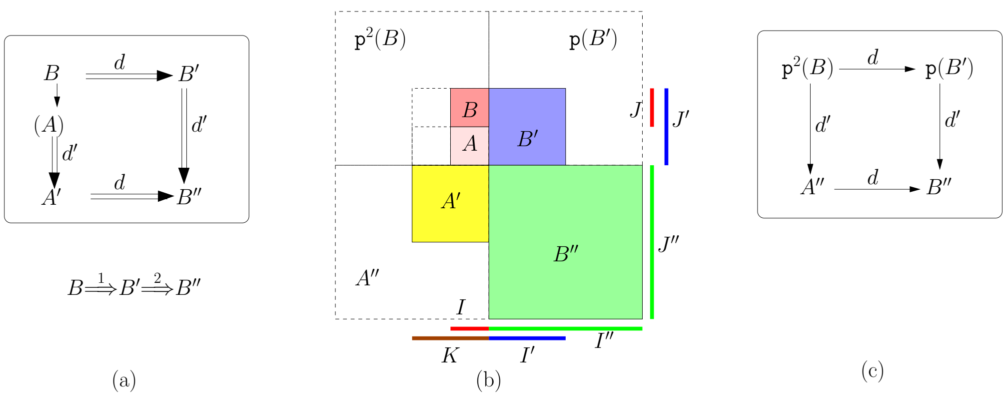


Figure 7: Commutative Diagram for Forcing

Lemma 14 (Commutative Diagram). Consider boxes in a smooth subdivision of $[-1, 1]^2$. Suppose $B \overset{d}{\Longrightarrow} B' \overset{d'}{\Longrightarrow} B''$ holds where $d \neq d'$. Then there exists a box A' such that $A' \overset{d}{\Longrightarrow} B''$.

This lemma is best understood in terms of a commutative diagram. It says that there exists some A where $p(A) = p(B)$ and some A' such that the relationships of (7) hold:

$$\begin{tikzcd} A \arrow[r, Rightarrow, "d"] \arrow[d, Rightarrow, "d'"'] & B' \arrow[d, Rightarrow, "d'"] \\ A' \arrow[r, Rightarrow, "d"'] & B'' \end{tikzcd} \tag{7}$$

Theorem 15 (Commutative Diagram). Consider boxes in a smooth subdivision $\mathbb{S}$ of $[-1,1]^D$ for $D \geq 2$. Suppose $B \overset{d}{\Longrightarrow} B' \overset{d'}{\Longrightarrow} B''$ holds for some $d \neq d'$. Then there exists a box A' in $\mathbb{S}$ such that $A' \overset{d}{\Longrightarrow} B''$.

3.4 Monotonicity in Smooth Subdivisions.

Theorem 15 motivates the following notions for boxes in a subdivision $\mathbb{S}$: for all $B \in \mathbb{S}$, if there exists $A \in \mathbb{S}$ such that $A \overset{d}{\Longrightarrow} B$ then we say B is *d-forced*, and write $* \overset{d}{\Longrightarrow} B$. Furthermore, let $R(B)$ denote the set of directions d such that B is d-forced, and $r(B) := |R(B)|$ is its cardinality. Note that $0 \leq r(B) \leq 2D$. Similarly, we write $B \overset{d}{\Longrightarrow} *$ if there exists $A \in \mathbb{S}$ such that $B \overset{d}{\Longrightarrow} A$, and let $S(B)$ denote the set of directions d such that $B \overset{d}{\Longrightarrow} *$; let $s(B) := |S(B)|$. Clearly, $0 \leq s(B) \leq D$.

Note some smooth subdivision $\mathbb{S}$ is normally implied in the use of this notation. Only for emphasis do we explicitly mention $\mathbb{S}$.

Note that if $A \overset{d}{\Longrightarrow} B$ and $B \overset{-d}{\Longrightarrow} B'$, then $p^2(A) \subseteq B'$. This is impossible since A, B' are boxes of a subdivision. In other words, $d \in R(B)$ implies $-d \notin S(B)$, and conversely $d \in S(B)$ implies $-d \notin R(B)$. Thus:

$$R(B) \cap -S(B) = \emptyset. \tag{8}$$

The following follows directly from Theorem 15:

Theorem 16. For boxes in a smooth subdivision, $B \Longrightarrow B'$ implies $R(B) \subseteq R(B')$ and hence $r(B) \leq r(B')$.

In a general subdivision, we could have non-monotone chains (i.e., a chain whose directions include both d and $-d$ for some d). We show that smoothness implies monotone chains:

Theorem 17. Chain in a smooth subdivision are monotone.

Proof. Consider any chain as in (6). It follows from the above corollary that $\{d_1, \ldots, d_i\} \subseteq R(B_i)$ for each i. It suffices to show that $-d_{i+1} \notin R(B_i)$. Note that $d_{i+1} \subseteq S(B_i)$. Therefore (8) implies $-d_{i+1} \notin R(B_i)$. **Q.E.D.**

If $A \Longrightarrow B$ and $\mathtt{p}^2(A) = \mathtt{p}(B)$, then $\mathtt{p}(A)$ is called a *split adjacent sibling* of B. The next lemma upper bounds $s(B)$ when B has split adjacent siblings:

Lemma 18.
(i) If B has exactly one split adjacent sibling, then $s(B) \leq 1$.
(ii) If B has at least two split adjacent siblings, then $s(B) = 0$.

The next result is critical. It shows that $r(B)$ must increase whenever B can force in more than one direction:

Lemma 19. Let $B \Longrightarrow B'$ in a smooth subdivision. If $s(B) > 1$ then $r(B) < r(B')$.

The next lemma shows that as $r(B)$ increases (up to $D+1$), we can predict a corresponding decrease on $s(B)$:

Lemma 20. For any non-root, $s(B) \leq \begin{cases} 0 & \text{if } r(B) > D, \quad (CASE\ 0) \\ 1 & \text{if } r(B) = D, \quad (CASE\ 1) \\ D - r(B) & \text{if } r(B) < D. \quad (CASE\ 2) \end{cases}$

Let $B \in \mathbb{S}(\mathbb{T})$. The *forcing graph* $F(B)$ of B is the directed acyclic graph rooted at B, whose maximal paths are all the maximal chains beginning at B. Note that the nodes in $F(B)$ belong to $\mathbb{S}(\mathbb{T})$. Evidently, the smooth split of B amounts to splitting every node in $F(B)$. Each node B' in $F(B)$ has $s(B')$ children; so B' is a leaf (or sink) iff $s(B') = 0$. If $s(B') > 1$, we call B' a *branching node*. Note that $F(B)$ would be a tree rooted at B if all the maximal chains are disjoint except at B. However, in general, maximal chains can merge.

Using the preceding two lemmas (Lemma 19 and Lemma 20) we can prove the following about $F(B)$:

THEOREM 21. Let B be a box in a smooth subdivision. There are at most $(D - r(B))!$ maximal paths in the forcing graph $F(B)$ where we define $x! = 1$ for $x \leq 0$.

3.5 Potential of Subdivision Tree.

We want to provided an amortized bound on the number of splits in a smooth split in a smooth subdivision $\mathbb{S}$. Our amortization argument refers to the subdivision tree $\mathbb{T} = \mathbb{T}(\mathbb{S})$ whose leaves constitute $\mathbb{S}$. Define the *potential* $\Phi(\mathbb{T})$ of the subdivision tree $\mathbb{T}$ to be the sum of the potential $\Phi(B)$ of all the nodes B in $\mathbb{T}$. The potential of node B is

$$\Phi(B) := \begin{cases} 0 & \text{if } B \text{ has no split children,} \\ \#\text{ of unsplit children of } B & \text{otherwise.} \end{cases} \tag{9}$$

Note that $\Phi(B) = 0$ iff it has no split children or all its children are split. Otherwise, $1 \leq \Phi(B) \leq 2^D - 1$. Intuitively, each unit of potential pays for the cost of a single split.

For $B \in \mathbb{S}(\mathbb{T})$, let $c(B)$ denote the number of nodes B' in $F(B)$ such that $\Phi(\mathtt{p}(B')) = 0$. But $\Phi(\mathtt{p}(B')) = 0$ iff $\mathtt{p}(B')$ has no split children or all of its children is split. Since B' is a leaf in $\mathbb{T}$, $\Phi(\mathtt{p}(B')) = 0$ implies that B' has no split siblings. Thus, $c(B)$ is counting the number of nodes in $F(B)$ with no split siblings.

THEOREM 22 (Main Theorem). Starting from the initial box $[-1, 1]^D$, a sequence of n smooth splits produces at most $(2^D(D+1)!)n$ splits. For fixed D, each smooth split produces an amortized $O(1)$ splits.

Proof. We use an amortization argument, generalizing the 2D argument. The smooth split of B amounts to splitting each node in its forcing tree $F(B)$. Recall that $c(B)$ is the number of nodes $B' \in F(B)$ with $\Phi(\mathtt{p}(B')) = 0$.

Claim: $c(B) \leq (D+1)!$.

We know that there are at most $D!$ maximal paths in $F(B)$. So the claim follows if each maximal chain

$$B = B_0 \xRightarrow{d_1} B_1 \xRightarrow{d_2} \cdots \xRightarrow{d_k} B_k$$

has at most $D+1$ indices $i = 1, \ldots, k$ such that $\Phi(\mathtt{p}(B_i)) = 0$. For such an i, we claim that $r(B_i) < r(B_{i+1})$. To show this, it suffices to prove that $d_{i+1} \notin R(B_i)$ because $d_{i+1} \in R(B_{i+1})$. Among the D adjacent siblings of B_i, there is one, say A, such that $A \xrightarrow{d_{i+1}} B_i$. If $d_{i+1} \in R(B_i)$ then $A' \xRightarrow{d_{i+1}} B_i$ for some child A' of A. Since $\Phi(\mathtt{p}(B_i)) = 0$, A has not been split and so A' does not exist. We have thus proved that $r(B_{i+1}) > r(B_i)$. It follows that if there are $\geq D+1$ such indices, the $D+1$-st index i has the property that $r(B_{i+1}) \geq D+1$. Then $s(B_{i+1}) = 0$ by Lemma 20. Hence B_{i+1} must be the last node B_k in the chain. This proves our claim.

The smooth split of B amounts to splitting each box $B' \in F(B)$. There are two cases of B': (A) $\Phi(\mathtt{p}(B')) > 0$. Then splitting B' can be charged to the corresponding unit decrease in potential $\Phi(\mathbb{T})$, since $\Phi(\mathtt{p}(B'))$ decreases by one when B' is split. (B) $\Phi(\mathtt{p}(B')) = 0$. Then splitting of B' will be charged 2^D, corresponding to one unit for splitting B' and $2^D - 1$ units for increase in $\Phi(\mathtt{p}(B'))$. It follows that the total charge for the smooth split of B is at most $2^D c(B) \leq 2^D (D+1)!$, as claimed. **Q.E.D.**

4 Conclusion

We have given a combinatorial proof that for any fixed dimension the amortized cost of performing a smooth split is $O(1)$. We did this by defining a suitable potential function based on the number of split siblings of a node, and by presenting a sequence of lemmas reasoning about how smooth splitting can propagate through the data structure.

We leave open a number of questions about amortized balancing costs for related quadtree models, including different notions of neighbors and balance, and for different subdivisions such as the alternatives considered in [Moo95].

In our model, we primarily leave open the tightening of our amortized cost upper bounds. In particular, we proved that a split can cause at most $d!$ chains, but our best lower bound shows only d chains. We conjecture that a closer analysis would lead to a much better upper bound. In particular, using the strategy outlined by Sheehy may lead to better bounds.

4.1 Acknowledgments

We would like to thank Don Sheehy for helpful conversations at the Fall Workshop on Computational Geometry (FWCG '13) and his subsequent outline of a strategy for attacking our problem using continuous techniques. We would also like to thank Joe Simons for answering questions about his co-authored paper [LSS13b].

References

[BEG94] Marshall W. Bern, David Eppstein, and John R. Gilbert. Provably good mesh generation. *J. Comput. Syst. Sci.*, 48(3):384–409, 1994.

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms (3. ed.)*. MIT Press, 2009.

[dBCvKO08] Mark de Berg, Otfried Cheong, Mark van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, Third edition, 2008.

[dBRS12] Mark de Berg, Marcel Roeloffzen, and Bettina Speckmann. Kinetic compressed quadtrees in the black-box model with applications to collision detection for low-density scenes. In Epstein and Ferragina [EF12], pages 383–394.

[EF12] Leah Epstein and Paolo Ferragina, editors. *Algorithms - ESA 2012 - 20th Annual European Symposium, Ljubljana, Slovenia, September 10-12, 2012. Proceedings*, volume 7501 of *Lecture Notes in Computer Science*. Springer, 2012.

[FB74] Raphael A. Finkel and Jon Louis Bentley. Quad trees: A data structure for retrieval on composite keys. *Acta Inf.*, 4:1–9, 1974.

[LSS13a] Maarten Löffler, Joe Simons, and Darren Strash. Dynamic planar point location with sub-logarithmic local updates. *arXiv*, abs/1204.4714, 2013.

[LSS13b] Maarten Löffler, Joseph A. Simons, and Darren Strash. Dynamic planar point location with sub-logarithmic local updates. In Frank Dehne, Roberto Solis-Oba, and Jörg-Rüdiger Sack, editors, *WADS*, volume 8037 of *Lecture Notes in Computer Science*, pages 499–511. Springer, 2013.

[Moo92] Doug Moore. *Simplicial Mesh Generation with Applications*. PhD thesis, Cornell University, 1992.

[Moo95] Doug Moore. The cost of balancing generalized quadtrees. In *Symposium on Solid Modeling and Applications*, pages 305–312, 1995.

[OvL82] Mark H. Overmars and Jan van Leeuwen. Dynamic multi-dimensional data structures based on quad- and k - d trees. *Acta Inf.*, 17:267–285, 1982.

[PM12] Eunhui Park and David M. Mount. A self-adjusting data structure for multidimensional point sets. In Epstein and Ferragina [EF12], pages 778–789.

[Sam90a] Hanan Samet. *Applications of spatial data structures - computer graphics, image processing, and GIS*. Addison-Wesley, 1990.

[Sam90b] Hanan Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, 1990.

[She] Donald R. Sheehy. private communication.

[She12] Donald R. Sheehy. New Bounds on the Size of Optimal Meshes. *Computer Graphics Forum*, 31(5):1627–1635, 2012.

[Sim] Joseph A. Simons. private communication.

[WCY13] Cong Wang, Yi-Jen Chiang, and Chee Yap. On soft predicates in subdivision motion planning. In *Proceedings of the twenty-ninth annual Symposium on Computational Geometry*, SoCG '13, pages 349–358, New York, NY, USA, 2013. ACM.

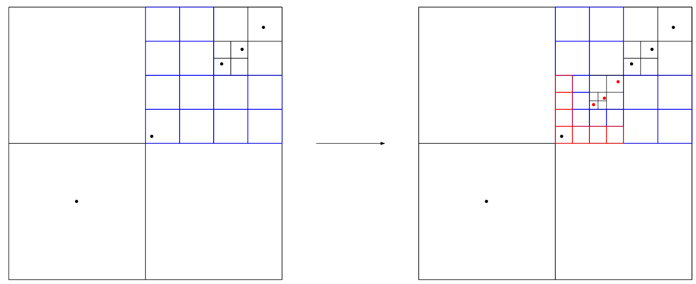


Figure 8: A counterexample to the algorithm sketched in the proof of Lemma 2.2 in [LSS13a]. True cells are shown in black, and B-cells in blue. After inserting the red points, the induced split of neighboring B-cells causes some new B-cells (shown in red) to be only 8-smooth with respect to their larger neighbors.

A Paper of Löffler et al. and Counterexample

We may generalize the notion of smoothness as follows: following Löffler et al. [LSS13b], call two neighbors *k-smooth* if the diameter of the boxes differ by at most a factor of k. In two dimensions this is equivalent to having at most k neighbors in a given direction. We have used the term "smoothness" to denote 2-smoothness.

A recent paper [LSS13b] claims that it is possible to maintain 4-smoothness in a related quadtree model in *worst-case* $O(1)$ time per split (presented as Lemma 2.2). The authors make this claim for quadtrees that are used to store point data, that use *compression*, and that consider boxes to be neighbors even if they only share a vertex (rather than requiring an edge). The subtree rooted at a node v is compressed if only one of the children of v contains points. As we show in a counterexample and as the authors themselves first determined in private correspondence [Sim] the problem with their algorithm stems from the case where points are inserted into a compressed part of the quadtree.

The extended version [LSS13a] of [LSS13b] contains a sketch of an algorithm that is intended to satisfy their assertion. It claims that after an insertion operation into a box B checking whether the neighbors of B are 2-smooth with respect to B (and splitting them if they are not) suffices to ensure that the entire tree is 4-smooth. The counterexample shown in Figure 8 demonstrates that the presented smoothing algorithm violates its stated invariants.

The authors distinguish between two types of boxes. They define *true* boxes as those that would exist in any unsmooth quadtree. That is, the parent of a true box contains at least two points. They define B-boxes as those that are introduced only for smoothness. True boxes are required to be 2-smooth with respect to their neighbors, whereas B-boxes are only required to be 4-smooth. Suppose we insert a point into a box B. The algorithm for regaining smoothness after this insertion is not given rigorously, but amounts to first splitting as necessary B (if it already contains a point), and then splitting the neighbors of a box into which a point is inserted if necessary to regain smoothness.

The quadtree and associated point data shown on the left in Figure 8 do not violate any required invariants, nor have any of its previous states – the true boxes (shown in black) are 2-smooth and the B-boxes (shown in blue) are 4-smooth with respect to their larger neighbors. After inserting the 3 red points on the right, according to their algorithm the three neighboring B-cells must split again for local smoothness. Their algorithm does not consider splitting the neighbors of the neighbors, which would need to split to achieve global smoothness. This results in some of the new, smaller B-cells being only 8-smooth with respect to their neighbors, which violates the required global 4-smoothness invariant.

A fundamental problem seems to be inserting points into B-cells (i.e. compressed parts of the tree). After the insertion of the first red point into box A all of the siblings of A, which are 4-smooth B-cells, become true cells. However, the sketched algorithm only considers promoting a single box (the one into which a point is inserted) to true per operation. Therefore the siblings of A, which are only 4-smooth with respect to their larger neighbors violated a required invariant even after just the first insertion. This seems to be a fundamental problem since a point insertion into a highly compressed quadtree may change a box arbitrarily many adjacency steps away into a true cell.

In private correspondence [Sim] the authors recognize compressed quadtrees as the primary issue. They prove a weaker claim, namely that it's possible to restore smoothness in worst-case $O(1)$ time if the quadtree does not need to compress. They also give a new algorithm which only handles inserting points into true cells, claiming that they "define [smoothness] on on uncompressed subtrees, and consider the whole quadtree [smooth] if each compressed subtree is [smooth]," meaning that this claim and algorithm suffice for their applications.

In this paper we consider the quadtree smooth only if all components are smooth with respect to each other, and allow for splits (and by proxy insertions) into arbitrary leaf boxes, including those originally created only for smoothness. It follows that the approach described by Löffler et al. approach does not work in our setting, and moreover that a similar approach is unlikely to work. This shows that our approach is robust: 4-balance and vertex neighbors, which are natural ways of tweaking our quadtree model, do not allow for a worst-case $O(1)$-time, local balancing algorithm.

B Proofs for Upper Bound in Arbitrary Dimensions

We define the necessary terminology for arbitrary dimensions.

B.1 Boxes, adjacencies and neighbors

We consider nice subsets of the Euclidean D-space $\mathbb{R}^D$, for some $D \geq 1$. The *standard cube* of dimension D is $[-1,1]^D$. Let $\mathbb{T}^D_\infty$ be the infinite tree rooted at $[-1,1]^D$ where each node in the tree is a box $B \subseteq [-1,1]^D$ with exactly 2^D congruent children whose interiors are pairwise disjoint, and whose union is equal to B. The nodes of $\mathbb{T}^D_\infty$ are called *aligned boxes*. Every aligned box B has a natural $depth(B) \geq 0$, corresponding to its depth in $\mathbb{T}^D_\infty$. The following is a useful fact about aligned boxes:

FACT 3. Let $D = d + d'$ for some $d, d' \geq 1$. If B and B' are boxes of $\mathbb{T}^d_\infty$ and $\mathbb{T}^{d'}_\infty$ (respectively), both of depth equal to $k \geq 0$, then $B \times B'$ is a box of depth k in $\mathbb{T}^D_\infty$. Conversely, every aligned box of $\mathbb{T}^D_\infty$ can be decomposed in this way.

A (box) *subdivision tree* $\mathbb{T}$ is any finite subtree of $\mathbb{T}^D_\infty$ that is rooted at $[-1,1]^D$ where every internal node has 2^D children. The set $\mathbb{S}(\mathbb{T})$ of leaves of $\mathbb{T}$ is called a (box) *subdivision* of the standard cube. Conversely, given any subdivision $\mathbb{S}$ of the standard cube into a set of aligned boxes, there is a unique subdivision tree $\mathbb{T}(\mathbb{S})$. When $D = 2$ ($D = 3$), $\mathbb{T}$ is usually called a quadtree (octree). Unless otherwise indicated, all boxes are aligned boxes (of various dimension $\leq D$). Note that boxes are closed sets. Let $j = -1, 0, 1, \ldots, D$. Two boxes B, B' are *j-adjacent* if $B \cap B'$ is a j-dimensional box. Four special cases are noteworthy:

- If they are D-adjacent, we say B and B' *overlap*.
- If they are $(D-1)$-adjacent, we say they are *neighbors*.
- 0-adjacency means they share a common corner only.
- (-1)-adjacency means the boxes are disjoint.

FACT 4. Let B, B' be overlapping aligned boxes. Then either $B \subseteq B'$ or $B' \subseteq B$.

The above definitions extend naturally to these lower dimensional boxes. In particular: if B, B' are boxes of dimension $c \leq D$, we say they are *neighbors* if $B \cap B'$ has dimension $c - 1$, and they *overlap* if $B \cap B'$ has dimension c.

B.2 Indicators: Directions and Children

Let an *indicator* be any element d in the set $\{-1, 0, 1\}^D$. Call d is a *child indicator* if there are no 0 components. E.g., $d = (1, -1, 1)$ or $d = (-1, -1, -1)$. Thus we can specify any non-root B as a d-child of its parent. Call d a *direction indicator* it has exactly one non-zero component. E.g., $d = (1, 0, 0)$ or $d = (0, -1, 0)$. The *opposite direction* to d is just $-d$. E.g., the opposition direction of $(1, 0, 0)$ is $(-1, 0, 0)$. Two directions are *adjacent* if they are different but not opposites of each other. E.g., $(1, 0, 0)$ and $(0, \pm 1, 0)$ are adjacent. Each box B at depth k has exactly 2^D subboxes at depth $k + 1$, called its *children*. These children can be indexed by each of the 2^D child indicators: if c is a child indicator, then the c-th child of B can be denoted by $B[c]$. If B is a c-th child of B', we may write

$$B \prec B' \qquad \text{or} \qquad B \overset{c}{\prec} B. \tag{10}$$

Let $\mathtt{p}(B)$ denote the *parent* of box B (this is well-defined except in the case $B = [-1,1]^D$). We can iterate this notation: $\mathtt{p}(\mathtt{p}(B)) = \mathtt{p}^2(B)$ denote the grandparent of B. This notation generalizes to $\mathtt{p}^n(B)$ for any $n \ge 0$ where $\mathtt{p}^0(B) = B$ and for $n \ge 1$, $\mathtt{p}^n(B) = \mathtt{p}(\mathtt{p}^{n-1}(B))$.

B.3 Projections and Co-Projections along a direction.

Given a box B, and $i \in \{1, \ldots, D\}$, then

- (Projection) $\mathtt{Proj}_i(B) := \prod_{j=1, j\neq i}^{D} I_j$ be a $(D-1)$ dimensional box.
- (Co-Projection) $\mathtt{Coproj}_i(B) := I_i$ denote the ith interval of $B = \prod_{j=1}^{D} I_j$.

We define the *indexed Cartesian product* $\otimes_i$ such that any box B can be recovered from its correspond projection and co-projection:

$$B = \underset{i}{\mathtt{Coproj}}(B) \otimes_i \underset{i}{\mathtt{Proj}}(B). \tag{11}$$

CONVENTION: If d is a direction indicator with a non-zero i-th component, then we may write $\mathtt{Proj}_d(B)$ instead of $\mathtt{Proj}_i(B)$. This convention can be extended to co-projections: $\mathtt{Coproj}_d(B)$ may be written instead of $\mathtt{Coproj}_i(B)$.

B.4 d-Neighbors.

Suppose B, B' are neighbors. Then there is a unique direction d such that B' is a "d-neighbor" of B. For $D = 1$, an interval B' is a $(+1)$-*neighbor* of B is the left-end point of B' equals the right-end point of B; equivalently, B is a (-1)-neighbor of B'. Suppose $D > 1$, and B, B' are neighbors. Then there is some $i \in \{1, \ldots, D\}$ such that $I = \mathtt{Coproj}_i(B)$ and $I' = \mathtt{Coproj}_i(B')$ are 0-adjacent, and $\mathtt{Proj}_i(B)$ and $\mathtt{Proj}_i(B'$ are $(D-1)$-adjacent. Thus I' is a (δ)-neighbor of I for some $\delta \in \{-1, +1\}$. This defines a direction d whose ith component is equal to δ. We then call B' a *d-neighbor* of B, and write

$$B \xrightarrow{d} B' \tag{12}$$

It follows from this definition that $B \xrightarrow{d} B'$ iff $B' \xrightarrow{-d} B$. We use the convention that, if the i-th component of d is 1 (resp., -1), then we can write $B \xrightarrow{i} B'$ (resp., $B \xrightarrow{-i} B'$) instead of $B \xrightarrow{d} B'$.

THEOREM 11[SINGLE DIRECTION]. Suppose $B \overset{d}{\Longrightarrow} B' \overset{d}{\Longrightarrow} B''$ holds for boxes in a smooth subdivision. Then $\mathtt{p}^2(B) = \mathtt{p}(B')$.

Proof. Wlog, let $d = (1, 0, \ldots, 0)$. Then

$$\begin{array}{rcl} B &=& I \times E \\ B' &=& I' \times E' \\ B'' &=& I'' \times E'' \end{array}$$

where $I \Longrightarrow I' \Longrightarrow I''$ and $E \prec E' \prec E''$. This implies that $\mathtt{p}(E) = E'$ or

$$\mathtt{p}^2(E) = \mathtt{p}(E') = E''. \tag{13}$$

By Lemma 10, we conclude that

$$\mathtt{p}^2(I) = \mathtt{p}(I'). \tag{14}$$

But (13) and (14) together imply

$$\mathtt{p}^2(I \times E) = \mathtt{p}(I' \times E')$$

which is what our theorem claims. **Q.E.D.**

THEOREM 13[TWO DIRECTIONS]. Consider boxes in a smooth subdivision of $[-1,1]^D$ $(D \geq 2)$. Suppose $B \overset{d}{\Longrightarrow} B' \overset{d'}{\Longrightarrow} B''$ holds where $d \neq d'$. Then $\mathtt{p}^2(B) \neq \mathtt{p}(B')$.

Proof. We know that d and d' must be adjacent directions, and without loss of generality, let $d = (1, 0, 0, \ldots, 0)$ and $d' = (0, 1, 0, \ldots, 0)$. We can thus write

$$\begin{aligned} B &= I \times J \times E \\ B' &= I' \times J' \times E' \\ B'' &= I'' \times J'' \times E'' \end{aligned}$$

where the I's and J's are intervals. From the premise $B \overset{1}{\Longrightarrow} B' \overset{2}{\Longrightarrow} B''$, we conclude that

$$\begin{array}{ccccc} I & \Longrightarrow & I' & \prec & I'', \\ J & \prec & J' & \Longrightarrow & J'', \\ E & \prec & E' & \prec & E''. \end{array}$$

Therefore

$$(I \times J) \overset{1}{\Longrightarrow} (I' \times J') \overset{2}{\Longrightarrow} (I'' \times J'')$$

and therefore Lemma 12 implies that

$$\mathtt{p}^2(I \times J) \neq \mathtt{p}(I' \times J').$$

This implies

$$\mathtt{p}^2(B) \neq \mathtt{p}(B').$$

Q.E.D.

LEMMA 14*[Commutative Diagram].* Consider boxes in a smooth subdivision of $[-1,1]^2$. Suppose $B \overset{d}{\Longrightarrow} B' \overset{d'}{\Longrightarrow} B''$ holds where $d \neq d'$. Then there exists a box A' such that $A' \overset{d}{\Longrightarrow} B''$.

Proof. Let

$$\begin{aligned} B &= I \times J \\ B' &= I' \times J' \\ B'' &= I'' \times J'', \end{aligned}$$

as illustrated by Figure 7(b). Wlog, let $d = (1, 0)$ and $d' = (0, 1)$ so that

$$\begin{array}{ccccc} I & \Longrightarrow & I' & \prec & I'' \\ J & \prec & J' & \Longrightarrow & J''. \end{array}$$

According to Lemma 12, $\mathtt{p}^2(B) \neq \mathtt{p}(B')$. And since $B \overset{d}{\Longrightarrow} B'$, $B \subseteq \mathtt{p}^2(B)$ and $B' \subseteq \mathtt{p}(B')$, we conclude

$$\mathtt{p}^2(B) \overset{d}{\longrightarrow} \mathtt{p}(B').$$

Likewise, $B' \overset{d'}{\Longrightarrow} B''$ implies $\mathtt{p}(B') \overset{d'}{\longrightarrow} B''$. Summarizing, we have shown that

$$\mathtt{p}^2(B) \overset{d}{\longrightarrow} \mathtt{p}(B') \overset{d'}{\longrightarrow} B''. \tag{15}$$

Since $\mathtt{p}^2(B)$, $\mathtt{p}(B')$ and B'' are all at the same depth, (15) implies

$$\begin{array}{ccccc} \mathtt{p}^2(I) & \longrightarrow & \mathtt{p}(I') & = & I'' \\ \mathtt{p}^2(J) & = & \mathtt{p}(J') & \longrightarrow & J'' \end{array}$$

By an application of Fact 3, there is an aligned box $A'' = \mathtt{p}^2(I) \times J''$ at the depth of B'' that completes (15) into the following commutative diagram:

$$\begin{tikzcd} \mathtt{p}^2(B) \arrow[r, "d"] \arrow[d, "d'"'] & \mathtt{p}(B') \arrow[d, "d'"] \\ A'' \arrow[r, "d"'] & B'' \end{tikzcd} \tag{16}$$

As illustrated in Figure 7(b,c), the commutative diagram involves four adjacent boxes at the same depth. From (16), we see that there is a box A in the subdivision with $\mathtt{p}(A) = \mathtt{p}(B)$ and

$$A \overset{d}{\Longrightarrow} B', \qquad A \overset{d'}{\longrightarrow} A''.$$

This last relationship would violate smoothness if A'' belongs to our subdivision, since $depth(A'') - depth(A) = 2$. Hence there is a child A' of A' such that

$$A \overset{d'}{\Longrightarrow} A' \overset{d}{\Longrightarrow} B''.$$

Moreover, A' must belong to the subdivision because otherwise, if it split, it would have a child $C \overset{d}{\Longrightarrow} B''$, which would violate smoothness. We thus have the following commutative (forcing) diagram which establishes our lemma:

$$\begin{tikzcd} A \arrow[r, Rightarrow, "d"] \arrow[d, Rightarrow, "d'"'] & B' \arrow[d, Rightarrow, "d'"] \\ A' \arrow[r, Rightarrow, "d"'] & B'' \end{tikzcd} \tag{17}$$

Q.E.D.

THEOREM 15[COMMUTATIVE DIAGRAM]. Consider boxes in a smooth subdivision $\mathbb{S}$ of $[-1,1]^D$ for $D \geq 2$. Suppose $B \overset{d}{\Longrightarrow} B' \overset{d'}{\Longrightarrow} B''$ holds for some $d \neq d'$. Then there exists a box A' in $\mathbb{S}$ such that $A' \overset{d}{\Longrightarrow} B''$.

Proof. We claim that there is some A and A' such that

$$A \overset{d'}{\Longrightarrow} A' \overset{d}{\Longrightarrow} B'',$$

as illustrated in Figure 7(b) for $D = 2$.

To do this construction of A and A', let us assume wlog that $d = (1, 0, 0, \ldots, 0)$ and $d' = (0, 1, 0, \ldots, 0)$. We can thus write

$$\begin{array}{rcl} B &=& I \times J \times E \\ B' &=& I' \times J' \times E' \\ B'' &=& I'' \times J'' \times E'' \end{array}$$

where the I's and J's are intervals. From the premise $B \xRightarrow{1} B' \xRightarrow{2} B''$, we conclude that

$$\begin{array}{lllll} I & \Longrightarrow & I' & \prec & I'', \\ J & \prec & J' & \Longrightarrow & J'', \\ E & \prec & E' & \prec & E''. \end{array}$$

Therefore,

$$I \times J \quad \xRightarrow{d} \quad I' \times J' \quad \xRightarrow{d'} \quad I'' \times J'',$$

and by Lemma 14, there exists $\widehat{A}$ such that

$$\widehat{A} \xRightarrow{d} I'' \times J''.$$

Therefore,

$$\widehat{A} \times E' \xRightarrow{d} I'' \times J'' \times E''.$$

Our theorem follows by choosing $A' = \widehat{A} \times E'$. **Q.E.D.**

Lemma 18.
(i) If B has exactly one split adjacent sibling, the $s(B) \leq 1$.
(ii) If B has at least two split adjacent siblings, then $s(B) = 0$.

Proof. (i) By assumption, there is a direction d and box A such that such that $A \xRightarrow{d} B$ and $\mathtt{p}^2(A) = \mathtt{p}(B)$. By way of contradiction, assume $s(B) \geq 2$. Then there is some $d' \neq d$ and B' such that $A \xRightarrow{d} B \xRightarrow{d'} B'$. By Theorem 13, $\mathtt{p}^2(A) \neq \mathtt{p}(B)$, contradiction.

(ii) By assumption, there are two directions $d \neq d'$ and boxes A, A' such that $A \xRightarrow{d} B$ and $A' \xRightarrow{d'} B$, and $\mathtt{p}^2(A) = \mathtt{p}^2(A') = \mathtt{p}(B)$. By way of contradiction, assume $s(B) > 0$. Then there exists B' such that $B \xRightarrow{d''} B'$ for some d''. So $d'' \neq d$ or $d'' \neq d'$. Wlog, suppose $d'' \neq d$. Since $A \xRightarrow{d} B \xRightarrow{d''} B'$, Theorem 15 implies that $\mathtt{p}^2(A) \neq \mathtt{p}(B)$, contradiction. **Q.E.D.**

Lemma 19. Let $B \Longrightarrow B'$ in a smooth subdivision. If $s(B) > 1$ then $r(B) < r(B')$.

Proof. Since $s(B) > 1$, there are two directions d, d' such that $B \xRightarrow{d} *$ and $B \xRightarrow{d'} *$. Without loss of generality, let $B \xRightarrow{d} B'$ and $B \xRightarrow{d'} A'$ for some A' in the subdivision. We already know that $r(B) \leq r(B')$. Clearly, $d \in R(B')$. So the inequality $r(B) < r(B')$ follows if we show that $d \notin R(B)$. By way of contradiction, assume $d \in R(B)$. So there exists a box A in the subdivision such that $A \xRightarrow{d} B \xRightarrow{d} B'$. By Theorem 11, $\mathtt{p}^2(A) = \mathtt{p}(B)$. However, we also have $A \xRightarrow{d} B \xRightarrow{d'} A'$. By Theorem 13, $\mathtt{p}^2(A) \neq \mathtt{p}(B)$. This is our contradiction. **Q.E.D.**

Lemma 20. For any non-root, $s(B) \le \begin{cases} 0 & \text{if } r(B) > D, \quad (CASE\ 0) \\ 1 & \text{if } r(B) = D, \quad (CASE\ 1) \\ D - r(B) & \text{if } r(B) < D. \quad (CASE\ 2) \end{cases}$

Proof. Since B is not the root (else we have a trivial subdivision), there are D sibling $A_1, \ldots, A_D$ and directions $d_1, \ldots, d_D$ such that $A_i \xrightarrow{d_i} B$. Clearly, $S(B) \subseteq \{d_1, \ldots, d_D\}$.

CASE 0: Suppose $r(B) > D$. There are two possibilities: if $R(B) \cap \{d_1, \ldots, d_D\}$ has more than one element, then Lemma 18 implies $s(B) = 0$, as desired. Otherwise, $R(B) \cap \{d_1, \ldots, d_D\}$ has exactly one element, say d_1. This can only mean that $r(B) = D + 1$, and the other D elements in $R(B)$ must be $-d_1, \ldots, -d_D$. This clearly implies $s(B) = 0$.

CASE 1: Suppose $r(B) = D$. If $R(B) \cap \{d_1, \ldots, d_D\}$ has one element, then Lemma 18 implies $s(B) \le 1$, as desired.

CASE 2: Suppose $r(B) < D$. If $R(B)$ contains at least one of the directions in $\{d_1, \ldots, d_D\}$ then $s(B) \le 1$, as desired. Otherwise, $R(B) \cap \{d_1, \ldots, d_D\}$ is empty, and so $R(B) \subseteq \{-d_1, \ldots, -d_D\}$. Since $S(B) \subseteq \{-d_1, \ldots, -d_D\} \setminus R(B)$, we conclude that $s(B) \le D - r(B)$, as desired. **Q.E.D.**

Theorem 21. Let B be a box in a smooth subdivision. There are at most $(D - r(B))!$ maximal paths in the forcing graph $F(B)$ where we define $x! = 1$ for $x \le 0$.

Proof. Write r for $r(B)$. The result is true if $r \ge D - 1$ or if there are no branching nodes. In these cases, $F(B)$ consists of a single path, and $(D - r)! = 1$.

So assume $r \le D - 2$ and there are branching nodes. There is a unique branching node $B' \in F(B)$ of minimum depth. Suppose B' has children $A_1, \ldots, A_s$ ($s = s(B')$) in $F(B)$. From Lemma 20, $s \le D - r(B') \le D - r$, and Lemma 19, $r(A_i) \ge r(B') + 1 \ge r + 1$. By induction on $D - r$, we may assume that in $F(A_i)$ ($i = 1, \ldots, s$) has at most $k!$ maximal paths where $k \le D - r(A_i) \le D - r - 1$. Thus the number of maximal paths in $F(B)$ is $\le s \cdot k! \le (D - r)(D - r - 1)! \le (D - r)!$.

Q.E.D.

We now prove the main result showing an amortized cost of $2^D(D + 1)! = O(1)$ splits per smooth split. To complement this bound, Appendix D proves a lower bound of $2^D(D + 1)$ on this amortized cost.

Theorem 22. Starting from the initial box $[-1, 1]^D$, a sequence of n smooth splits produces at most $(2^D(D + 1)!)n$ splits. For fixed D, each smooth split produces an amortized $O(1)$ splits.

Proof. We use an amortization argument, generalizing the 2D argument. The smooth split of B amounts to splitting each node in its forcing tree $F(B)$. Recall that $c(B)$ is the number of nodes $B' \in F(B)$ with $\Phi(\mathtt{p}(B')) = 0$.

Claim: $c(B) \le (D + 1)!$.

We know that there are at most $D!$ maximal paths in $F(B)$. So the claim follows if each maximal chain

$$B = B_0 \xRightarrow{d_1} B_1 \xRightarrow{d_2} \cdots \xRightarrow{d_k} B_k$$

has at most $D + 1$ indices $i = 1, \ldots, k$ such that $\Phi(\mathtt{p}(B_i)) = 0$. For such an i, we claim that $r(B_i) < r(B_{i+1})$. To show this, it suffices to prove that $d_{i+1} \notin R(B_i)$ because $d_{i+1} \in R(B_{i+1})$. Among the D adjacent siblings of B_i, there is one, say A, such that $A \xrightarrow{d_{i+1}} B_i$. If $d_{i+1} \in R(B_i)$ then $A' \xRightarrow{d_{i+1}} B_i$ for some child A' of A. Since $\Phi(\mathtt{p}(B_i)) = 0$, A has not been split and so A' does not exist. We have thus proved that $r(B_{i+1}) > r(B_i)$. It follows that if there are $\ge D + 1$ such indices, the

$D+1$-st index i has the property that $r(B_{i+1}) \geq D+1$. Then $s(B_{i+1}) = 0$ by Lemma 20. Hence B_{i+1} must be the last node B_k in the chain. This proves our claim.

The smooth split of B amounts to splitting each box $B' \in F(B)$. There are two cases of B': (A) $\Phi(\mathtt{p}(B')) > 0$. Then splitting B' can be charged to the corresponding unit decrease in potential $\Phi(\mathbb{T})$, since $\Phi(\mathtt{p}(B'))$ decreases by one when B' is split. (B) $\Phi(\mathtt{p}(B')) = 0$. Then splitting of B' will be charged 2^D, corresponding to one unit for splitting B' and $2^D - 1$ units for increase in $\Phi(\mathtt{p}(B'))$. It follows that the total charge for the smooth split of B is at most $2^D c(B) \leq 2^D (D+1)!$, as claimed. **Q.E.D.**

C Exponential Lower Bound Construction

We now give a construction to show that the exponential dependence on D is unavoidable. But we first give the bounds for $D = 1$ and $D = 2$ to build the intuition.

¶1. Interval Trees For $D = 1$, we obtain the following tight bound:

LEMMA 23. Every sequence of n smooth splits starting from an initial interval has a total cost of $\leq 4n$. Moreover, the constant of 4 is optimal.

The upper bound comes from the general potential argument. In this case, the potential of a node I (i.e., interval) of the interval tree is 1 if it has one split child, and one unsplit child. All other nodes has 0 potential. The smooth split of I_0 induces a unique chain $I_0 \Longrightarrow I_1 \Longrightarrow \cdots \Longrightarrow I_k$, and we only need to charge the cost of splitting the first I_1 and last interval I_k because the others can be paid for by a corresponding decrease in potential. The charge for I_1 and I_k is ≤ 4 units (two units to do the splitting, and two units for possible increase in potential).

To see that 4 is tight, consider the sequence of smooth splits on:

$$I, I.(-e), I.(-e)e, I.(-e)e^2, \ldots, I.(-e)e^n \tag{18}$$

where $e = (+1)$ is a child indicator. Each of these smooth splits (except for the first) will cause 2 splits, or $2n - O(1)$ overall. At the end of this sequence, we do two more smooth splits:

$$I.(-e)e^n(-e), \quad I.(e)(-e)^n e. \tag{19}$$

Each of these will cause about n more splits. This achieves $4n - O(1)$. This proves:

LEMMA 24. For interval trees ($D = 1$), any sequence of n smooth splits can cause at most $4n + O(1)$ splits. Moreover, there is a sequence of $n + O(1)$ smooth splits that has $4n$ splits.

¶2. Quadtrees We generalize the one dimensional example to $D = 2$:

Let $c = (1, 1)$ be the child indicator. Beginning with an initial box B, we will perform chain splits on the following sequence of boxes:

$$B, \quad B.(-c), \quad B.(-c).c, \quad B.(-c).c^2, \quad \ldots, \quad B.(-c).c^n. \tag{20}$$

This is illustrated in Figure 9, where the result of the third smooth split is illustrated in the transition from (b) to (c): notice that in this smooth split, four actual splits occur.

Thus, in analogy to (18), we get $4n - O(1)$ splits using n smooth splits of (20).

Next, we do the analogy of (19): if we smooth split $B.(-c).c^n.(-c)$, we will get $2n - O(1)$ splits in the box $B.(-c)$. Likewise, we can do three other smooth splits to yield $2n - O(1)$ splits each. These are splits (respectively) of subboxes in $B.c, B.c_1, B.c_2$ – see Figure 9(a). This gives us $4(2n - O(1)) = 8n - O(1)$ overall. Combined with the $4n - O(1)$, the overall number is $12n - O(1)$.

As for upper bound, we apply the above general amortization bound to this case. We have at most two chains in a smooth split, and up to 5 of the splits are not accounted for, and we need to charge 4 units for each (3 units for increase in potential and 1 unit for the split). Thus the cost is $20n$ for a sequence of n splits. This proves:

LEMMA 25. For quadtrees ($D = 2$), any sequence of n smooth splits can cause at most $20n + O(1)$ splits. Moreover, there is a sequence of $n + O(1)$ smooth splits that causes $12n$ splits.

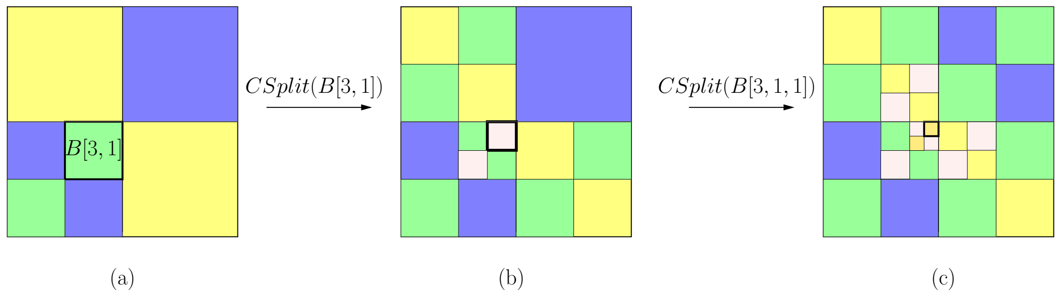


Figure 9: Smooth split in quadtrees ($D = 2$)

¶3. Arbitrary Dimensions The argument to be presented will be a direct generalization of the $D = 2$ case.

Suppose $B = \prod_{i=1}^{D}[m_i \pm r]$. For any $j = 1, \ldots, D$, we can also write B in the form $A \otimes_j [m_j \pm r]$ where $A = \mathtt{Proj}_j(B) = \left(\prod_{i=1, i\neq j}^{D}[m_i \pm r]\right)$.

A child indicator c can be written as

$$c = \sum_{i=1}^{D} d_i = \sum_{i=1}^{D} \delta_i e_i \tag{21}$$

where $d_i = \delta_i e_i$ with $\delta_i \in \{-1, 1\}$. If $B = \prod_{i=1}^{D}[m_i \pm r]$, the c-th child of B is defined as

$$B.c := \prod_{i=1}^{D}[m_i + \frac{1}{2}\delta_i \cdot r \pm \frac{1}{2} \cdot r]. \tag{22}$$

If $\sigma = c_1 c_2 \cdots c_n$ is a sequence child indicators, then we inductively define $B.\sigma$ as $(B.\sigma') \cdot c_n$ where $\sigma' = c_1 \cdots c_{n-1}$.

Two child indicators c and c' are said to be *neighbors* if $c' = c + 2d$ for some direction indicator d. For any box B and child indicators c and c', the following are equivalent:

(a) $B.c$ and $B.c'$ are neighbors.

(b) c and c' are neighbors as child indicators.

(c) $c' = c + 2d$ for some direction indicator d.

These equivalences comes from the definitions of neighbor relationships. The next lemma shows the precise role of d in these neighbor relationships:

Lemma 26. Let c and $c+2d$ be child indicators for some direction indicator d, and B, B' are aligned boxes.

(R1) $B.c \xrightarrow{d} B.(c + 2d)$. Equivalently, $B.(c + 2d) \xrightarrow{-d} B.c$.

(R2) $B \xrightarrow{d} B'$ implies $B.(c + 2d) \xrightarrow{d} B'$.

(R3) $B \xrightarrow{d} B'$ implies $B \xrightarrow{d} B'.c$.

Proof. (R1): Let $c' = c + 2d$ where $d = \delta e_j$ (for some $j = 1, \ldots, D$ and $\delta \in \{-1, +1\}$). Using the notation of (22),

$$\begin{aligned} B.c &= \underset{j}{\mathtt{Proj}}(B.c) \otimes_j \underset{j}{\mathtt{Coproj}}(B.c) \\ &= \underset{j}{\mathtt{Proj}}(B.c) \otimes_j [m_j - \frac{1}{2}\delta_j \cdot r \pm \frac{1}{2} \cdot r] \\ B.c' &= \underset{j}{\mathtt{Proj}}(B.c') \otimes_j \underset{j}{\mathtt{Coproj}}(B.c') \\ &= \underset{j}{\mathtt{Proj}}(B.c') \otimes_j [m_j + \frac{1}{2}\delta_j \cdot r \pm \frac{1}{2} \cdot r]. \end{aligned}$$

Since $c' = c + 2d = c + 2\delta e_j$, we conclude that $\delta = \delta_j$ and
(I) $\mathtt{Coproj}_j(B.c) \xrightarrow{(\delta)} \mathtt{Coproj}_j(B.c')$, and
(II) $\mathtt{Proj}_j(B.c) = \mathtt{Proj}_j(B.c')$.
From (I) and (II), we conclude that $B.c \xrightarrow{d} B.c'$ (using Lemma 9). This conclusion is clearly equivalent to $B.c' \xrightarrow{-d} B.c$.

(R2-R3) in case $D = 1$ is easy to see: we have $c + 2d$ is a child indicator iff $c = (-\delta)$ and $d = (\delta)$ for some $\delta \in \{+1, -1\}$. Then for intervals I and I', if $I \xrightarrow{(\delta)} I'$ then

$$I.(\delta) \xrightarrow{(\delta)} I', \qquad I \xrightarrow{(\delta)} I'.(-\delta).$$

I.e.,

$$I.(c + 2d) \xrightarrow{(\delta)} I', \qquad I \xrightarrow{(\delta)} I'.c.$$

(R2) for $D \geq 2$: Say $d = \delta e_j$ for some $j = 1, \ldots, D$ and $\delta \in \{+1, -1\}$. Then we have
(a) $\mathtt{Coproj}_j(B) \xRightarrow{(\delta)} \mathtt{Coproj}_j(B')$, and
(b) $\mathtt{Proj}_j(B) \subseteq \mathtt{Proj}_j(B')$ or $\mathtt{Proj}_j(B') \subseteq \mathtt{Proj}_j(B)$.
Note that (b) is a consequence of B, B' being aligned.
(A) It follows from the case $D = 1$ that $\mathtt{Coproj}_j(B.(c + 2d)) \xRightarrow{(\delta)} \mathtt{Coproj}_j(B')$, and
(B) $\mathtt{Proj}_j(B.(c + 2d)) \subseteq \mathtt{Proj}_j(B')$ or $\mathtt{Proj}_j(B') \subseteq \mathtt{Proj}_j(B.(c + 2d))$.
Moreover, (A) and (B) implies $B.(c + 2d)) \xRightarrow{d} B'$. This proves (R2).

(R3) for $D \geq 2$: this is shown in the same way as (R2). **Q.E.D.**

We can think of (R1)–(R3) as transformation rules.

Lemma 27. Let $c' = c + 2d$ for some direction indicator d. For $n > m \geq 0$, and any box B, we have the forcing relationships:

(F1) $B.c.(-c)^n \xRightarrow{d} B.c'.(-c')^m$

(F2) $B.c.(c')^n \xRightarrow{d} B.c'.c^m$

Proof. Lemma 26(R1) shows that

$$B.c \xrightarrow{d} B.(c + 2d). \tag{23}$$

To show (F1), we observe that that $-c$ has the form $-c = -c' + 2d$. Therefore Lemma 26(R2) applied to (23) yields $B.c.(-c) \xrightarrow{d} B.(c + 2d)$. Hence inductively, for all $n \geq 0$:

$$B.c.(-c)^n \xrightarrow{d} B.(c + 2d). \tag{24}$$

Again, Lemma 26(R3) shows that $(-c - 2d)$ can be appended to the right hand side of (24), giving us

$$B.c.(-c)^n \xrightarrow{d} B.(c + 2d)(-c - 2d).$$

Hence inductively, for all $m \geq 0$, we obtain

$$B.c.(-c)^n \xrightarrow{d} B.(c + 2d)(-c - 2d)^m. \tag{25}$$

When $n > m$, the depth of the left hand side is greater than the right hand side. Thus (25) represents a forcing relationship:

$$B.c.(-c)^n \overset{d}{\Longrightarrow} B.(c + 2d)(-c - 2d)^m.$$

This proves (F1). (F2) is shown in the same way. **Q.E.D.**

¶4. An Exponential Lower Bound. We want to see how the forcing relationships in Lemma 26(F1) are propagated as we perform the following sequence of $n+2$ smooth splits on the following boxes:

$$B,\ B.(-c),\ B.(-c).c,\ B.(-c).c^2\ ,\dots,\ B.(-c).c^n. \tag{26}$$

We may assume that $n \geq D$. After the 2nd operation $\texttt{sSplit}(B.(-c))$, we have created forcing relationships of the form (F1), namely

$$B.(-c).c \overset{d}{\Longrightarrow} B.(-c_1) \tag{27}$$

for each neighbor $c_1 = c - 2d$ of c. This implies that the 3rd operation $\texttt{sSplit}(B.(-c)c)$ would induce the split of $B.(-c_1)$.

There are D such splits. However, new forcing relationships

$$B.(-c).c^2 \overset{d}{\Longrightarrow} B.(-c_1).c_1 \tag{28}$$

are created. In other words, the forcing relationship (27) is sustained in (28), albeit at the "next level". Moreover, we also see a forcing chain with two links: if c_2 is a neighbor of c_1 but not of c, then (28) is really the prefix of a longer chain:

$$B.(-c).c^2 \overset{d}{\Longrightarrow} B.(-c_1).c_1 \overset{d'}{\Longrightarrow} B.(-c_2) \tag{29}$$

where $c_2 = c_1 - 2d'$.

To give a complete description to this phenomenon, let us consider the set of 2^D children indicators, $\{-1, +1\}^D$. Fix any $c_0 \in \{-1, +1\}^D$ and consider the following DAG rooted at c_0: the nodes at level $i \geq 0$ of the DAG are those indicators c whose Hamming distance from c_0 is exactly i. The edges of the DAG goes from c in level i to c' in level $i+1$ iff c, c' are neighbors. The DAG

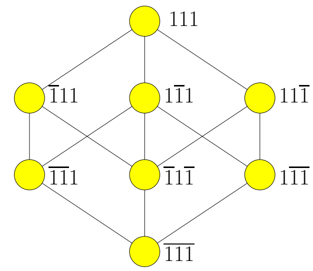


Figure 10: Lattice of child indicators

is a lattice with top element c_0 and bottom element $-c_0$, as illustrated by Figure 10 (writing $\overline{1}$ instead of -1).

Suppose $(c_0, c_1, \ldots, c_D)$ is a path of length D in this lattice (so $c_D = -c_0$). It follows from the foregoing that, after $m+1$ operations in (26), assuming $m \geq D$, we obtain the following forcing chain with D links:

$$B.(-c_0).c_0^m \Longrightarrow B.(-c_1).c_1^{m-1} \Longrightarrow B.(-c_2).c_2^{m-2} \Longrightarrow \cdots \Longrightarrow B.(-c_D).c_D^{m-D}. \tag{30}$$

Writing B_m for the box $B.(-c_0).c_0^m$, it follows that the size of the forcing graph $F(B_m)$ is at least 2^D since there are 2^D distinct boxes. Thus the $m+2$-nd smooth split will cause 2^D splits.

¶5. Stronger Lower Bound. The foregoing proves that the the amortized cost of each smooth split is at least 2^D. The argument only exploit the forcing relationships of Lemma 26(F1). To push this lower bound a little further, we will need the forcing relationships of Lemma 26(F2).

Theorem 28. For all $n \geq 1$, there is a sequence of $n + O(1)$ smooth splits that causes $n(D+1)2^D$ splits.

Proof. We begin with a sequence of $n+2$ smooth splits on the boxes (26). For $m \geq D$, we know that the m-th smooth split causes 2^D splits.

Next consider any child indicator c_1, and look at the box $B.c_1$. If the Hamming distance between c_1 and c is h, then for $m \geq D$, the $m+2$-nd smooth split causes $B.(-c_1)\cdot c_1^{m-h}$ to split (see (30)). We now want to exploit the potential that is stored up in the subbox $B.(-c_1)$.

Consider $B.(-c_1)$. We have established that the sequence of smooth splits (26) causes the following smooth splits in subboxes of $B.(-c_1)$:

$$B.(-c_1),\ B.(-c_1).c_1,\ B.(-c_1).c_1^2\ ,\ldots,\ B.(-c_1).c_1^m \tag{31}$$

for $m = n-2-h$. Next suppose c_2 is any neighbor of c_1, say $c_2 = c_1 + 2d_1$. Then the smooth splits in (31) produces the following sequence of boxes:

$$B.(-c_1).c_2,\ B.(-c_1).c_1.c_2,\ B.(-c_1).c_1^2.c_2,\ ,\ldots,\ B.(-c_1).c_1^m.c_2 \tag{32}$$

Moreover, for any two consecutive boxes in (32), there is a forcing relationship:

$$B.(-c_1).c_1^k.c_2 \xrightarrow{d_1} B.(-c_1).c_1^{k-1}.c_2, \quad (k \geq 1). \tag{33}$$

To see this, we first note that (R1) applied to $B' = B.(-c_1).c_1^{k-1}$ implies

$$B.(-c_1).c_1^k = B'.c_1 \xrightarrow{d_1} B'.c_2 = B.(-c_1).c_1^{k-1}.c_2$$

since $c_2 = c_1 + 2d_1$. Next, (R2) applied to $B'.c_1 \xrightarrow{d_1} B'.c_2$ implies that $B'.c_1.c_2 \xrightarrow{d_1} B'.c_2$; this proves (33). By looking at the depths of both sides of (33), we conclude that it is actually a forcing relationship. This means that we can rewrite (32) in reverse as a forcing chain,

$$B.(-c_1).c_1^m.c_2 \overset{d_1}{\Longrightarrow} B.(-c_1).c_1^{m-1}.c_2 \overset{d_1}{\Longrightarrow} B.(-c_1).c_1^{m-2}.c_2, \overset{d_1}{\Longrightarrow} \ldots \overset{d_1}{\Longrightarrow} B.(-c_1).c_2. \tag{34}$$

We need one final observation: Applying (R1) to $B' = B.(-c_1).c_1^{m-1}$ with $c_2 = c_1 + 2d_1$, we obtain

$$B.(-c_1).c_1^m = B'.c_1 \xrightarrow{d_1} B'.c_2 = B.(-c_1).c_1^{m-1}.c_2$$

Next, applying (R2) to the previous relation with $-c_1 = -c_2 + 2d_1$, we obtain

$$B.(-c_1).c_1^m(-c_1) \overset{d_1}{\Longrightarrow} B.(-c_1).c_1^{m-1}.c_2.$$

It follows that if we smooth split this

$$B.(-c_1).c_1^m(-c_1)$$

then we will cause D chain reactions for each c_2 that is a neighbor of c_1.

Since there are 2^D choices of c_1, this will cause a sequence of $D2^D$ such chain reactions. Each chain is $n - O(1)$ long. This completes our proof. **Q.E.D.**