A Unified Approach to Tail Estimates for Randomized Incremental Construction

Sandeep Sen
Department of CSE, I.I.T. Delhi, India
ssen@cse.iitd.ac.in

Abstract

By combining several interesting applications of random sampling in geometric algorithms like point location, linear programming, segment intersections, binary space partitioning, Clarkson and Shor [4] developed a general framework of randomized incremental construction (*RIC*). The basic idea is to add objects in a random order and show that this approach yields efficient/optimal bounds on **expected** running time. Even quicksort can be viewed as a special case of this paradigm. However, unlike quicksort, for most of these problems, sharper tail estimates on their running times are not known. Barring some promising attempts in [15, 3, 20], the general question remains unresolved.

In this paper we present a general technique to obtain tail estimates for *RIC* and and provide applications to some fundamental problems like Delaunay triangulations and construction of Visibility maps of intersecting line segments. The main result of the paper is derived from a new and careful application of Freedman's [9] inequality for Martingale concentration that overcomes the bottleneck of the better known Azuma-Hoeffding inequality. Further, we explore instances, where an *RIC* based algorithm may not have inverse polynomial tail estimates. In particular, we show that the *RIC* time bounds for trapezoidal map can encounter a running time of $\Omega(n \log n \log\log n)$ with probability exceeding $\frac{1}{\sqrt{n}}$. This rules out inverse polynomial concentration bounds within a constant factor of the $O(n \log n)$ expected running time.

2012 ACM Subject Classification Theory of Computation → Computational Geometry

Keywords and phrases ric, tail estimates, martingale, lower bound

Digital Object Identifier 10.4230/LIPIcs.STACS.2019.58

1 Introduction

One of the most natural and elegant paradigm for designing geometric algorithms is randomized incremental construction or *RIC* for short. It can be viewed as generalization of Quicksort and evolved over a sequence of papers [18, 2] eventually culminating in a very general framework of *configuration space* by Clarkson and Shor [4]. The basic procedure is described in Figure 1. Quicksort itself can be viewed through this paradigm as refinement of the current partially ordered set (partitions) by *inserting* the next splitter and updating the partitions.

An abstract configuration space, that we will refer to as $\Pi(S)$ is defined by the given set S of n elements. A configuration σ is a subset of the Euclidean space that is defined by $O(1)$ objects of S denoted by $d(\sigma)$. The *conflict list* of a configuration σ is denoted by $\ell(\sigma) = \sigma \cap \{S - d(\sigma)\}$, i.e. the elements of S that intersect σ, not including $d(\sigma)$. We define $\Pi^i(S) = \{\sigma : |\ell(\sigma)| = i\}$ and $\Pi(S) = \bigcup_{i=0}^{n} \Pi^i(S)$. For analyzing *RIC* , the properties of $\Pi^0(R)$ for a randomly chosen subset $R \subset S$ turns out to be very crucial. In particular, they characterize how the *uninserted* elements of S interact with the current partially constructed structure, denoted by $H(R)$. For notational simplicity, the conflict list of any $\sigma \in \Pi^0(R)$, $\ell(\sigma) = \sigma \cap S$ (instead of $\sigma \cap R$) which will be an important parameter in the analysis.

We illustrate the framework and the notations in the context of quicksort. A configuration σ is an interval defined by $\{x_i, x_j\} \in S$, i.e., $|d(\sigma)| = 2$ and $\ell([x_i, x_j]) = S \cap [x_i, x_j]$, i.e., the points of S that lie in this interval. The configurations space $\Pi(S)$ is the set of intervals

© Sandeep Sen;
licensed under Creative Commons License CC-BY

```
Procedure RIC($S$).
1 $\mathcal{N} = [x_1, x_2 \ldots x_n]$ : a random permutation of $S$. ;
2 $T \leftarrow \phi$ , $H$ is some data structure appropriately initialized;
3 for $i = 1$ to $n$ do
4     $T \leftarrow T \cup \{x_i\}$;
5     Update $H(T)$
6 Return $H(T)$ ;
```

Figure 1 Randomized Incremental Construction.

defined by all pairs $x_i, x_j \in S$. Note that $\Pi^0(S)$ consists of intervals defined by the sorted set of points in S. The associated data structure $H(R)$ for a sample $R \subset S$ may be thought of as an *incidence relation* between intervals $\sigma \in \Pi^0(R)$ and $\ell(\sigma) \in S - R$. This is precisely the ordered intervals induced by the points in R and the uninserted points $S - R$ partially ordered by these intervals.
When the next randomly chosen element $x \in S - R$ is added to R, $H(R)$ is updated to $H(R \cup \{x\})$ and this cost contributes to the running time of *RIC* . In [4] , the data-structure is maintained as a *conflict graph* that maintains relation between $\sigma \in \Pi^0(R)$ and $S - R$ as a bipartite graph. Although configurations are created and destroyed, the total cost can be shown to be twice the cost of new configurations created and the ones destroyed can be charged to the cost of its creation. In the case of quicksort, $H(S)$ yields the information about the sorted set since it contains all the ordered intervals. The reader is referred to [4, 17] for further details regarding this framework. We include a brief description in the Appendix.

Although the initial analysis in [4] was somewhat intricate and complex, subsequent papers [1, 19] simplified the analysis using a clever technique called *backward analysis*. In this paper, we will appeal to the simpler analysis. Often the full conflict graph information can be replaced by simpler relations (see [10, 19]. However, the conflict graph approach is very general and works for diverse problems.

A related, but a somewhat distinct approach was developed in the work of Seidel [19, 20, 1] that maintains a solution inductively, that is recomputed from scratch if the next insertion modifies it. For example, the closest pair can be computed in this similar manner ([12]). Although our techniques can be applied to the latter work also, we will focus primarily on the Clarkson-Shor incremental paradigm of a configuration space.

While the primary focus was on deriving bounds on the expected running time of *RIC*, it was clear that obtaining concentration bounds on the expected running time would make the *RIC* more practical and attractive. The conjecture was that the running times are concentrated around their expected values but to the best of our knowledge, there has been little progress in this direction barring some papers related to computing line segment intersections using *RIC* [3, 15] and on fixed dimensional linear programming [20]. For problems like planar hulls, high probability bounds can be proved based on linear ordering that do not extend to higher dimensions. Although, by resampling $\Omega(\log n)$ times, we can obtain inverse polynomial concentration bounds, it comes at the expense of the increasing the running time by an $O(\log n)$ factor.

In this paper, we revisit the problem and present a general methodology to obtain tail bounds for specific problems like *Delaunay triangulation, 3-D convex hulls, and line segment intersections* that are based on *RIC* - these are summarized in Table 1. For the case of

Table 1 Summary of results for some representative problems using our technique.
w.p. : "with probability" $\alpha(n)$: inverse Ackerman function, $\gamma > 0$ some constant
The third result alludes to a version that doesn't use conflict lists explicitly.

Problem	**Expected Running time**	**Tail estimates**
Quicksort	$O(n \log n)$	$O(c\gamma n \log n)$ w.p. $\geq 1 - n^{-c}$
Delaunay Triangulation	$O(n \log n)$	$O(c\gamma n \log n)$ w.p. $\geq 1 - 2^{-c}$
Segment intersections/ Trapezoidal maps	$O(n \log n + m)$ n: segments, m: intersections	$O(n \log n + m)$ w.p. $1 - \exp -(\frac{m+n\log n}{n\alpha(n)})$

finding intersection of line segments, our bounds are not only better than [15] but also distinctly less involved in terms of calculations. Moreover, we develop a unified approach based on a well-known Martingale inequality, unlike the previous approaches that were arguably more ad-hoc.

We also provide some evidence of the non-existence of such generalized tail estimates by constructing an instance of the trapezoidal maps (based on maintaining conflict lists) for which inverse polynomial tail estimates is possible only for running times exceeding $\Omega(n \log n \log\log n)$ and rules out concentration bounds within constant factor of expectation.

▶ Remark. In *RIC* based algorithms, the term *running time* is often interchangeably used with *structural changes* caused by each insertion, particularly when the underlying data structure is a conflict-graph.

1.1 Main techniques and organization

We begin by introducing a useful probabilistic inequality, viz., Freedman's inequality [9] for Martingales that will be used to model the running time of the generic *RIC* algorithms. In the following section, we illustrate the use of this technique for analyzing quicksort that can also be viewed within the framework of *RIC* . The application to quicksort doesn't yield any better result than what was previously known, but it provides a stepping stone to the more complex and general framework. In particular, even the more commonly used Azuma-Hoeffding bound is not known to be effective for quicksort concentration bounds because its dependence on the worst-case bound (sum of bounded differences).

It is unlikely that the previously known techniques for concentration bound of quicksort can be extended to generic *RIC* analysis as the intermediate structures in *RIC* are more complex and can be bound only in an *expected* sense. Starting with quicksort in section 3 which has a predictable intermediate structure consisting of $i + 1$ intervals in stage i, we tackle increasingly complex scenarios. In the case of Delaunay triangulation (in section 4), although the number of triangles in the i-th step is $O(i)$, the number of additional triangles created in the i-th step can be bound only in expectation. In section 5 we consider the case of line segment intersections where the size of the intermediate structure can be bound only in an expected sense and may have a large variance.

In the last section, we construct a family of inputs for the *RIC* for trapezoidal maps to show that inverse polynomial concentration bound is not attainable if we rely on conflict-list based update mechanism.

2 Basic framework and tools

Let $S = \{x_1, x_2 \dots x_n\}$ be a set of n objects. A permutation π of S is a 1-1 function $\pi(i) = j$ where $i, j \in \{1, 2, \dots n\}$ that produces a permutation $x_{\pi(1)}, x_{\pi(2)} \dots x_{\pi(n)}$. A *random* permutation of S is one of the $n!$ permutation function chosen uniformly at random. A *k-prefix* of a permutation π is the sequence of the first k objects and denoted by $\pi^{(k)}$ consisting of $x_{\pi^{-1}(1)}, x_{\pi^{-1}(2)} \dots x_{\pi^{-1}(k)}$. Note that the permutation x_3, x_1, x_2 is defined as $\pi(1) = 2; \pi(2) = 3; \pi(3) = 1$, so the permutation is $x_{\pi^{-1}(1)}, x_{\pi^{-1}(2)}, x_{\pi^{-1}(3)}$.

Let $X_1, X_2 \dots X_n$ where $X_i = x_{\pi^{-1}(i)}$ corresponding to a random permutation π. Further, let $\bar{X}^{(k)}$ denote a sequence of k random variables that will also be used it to refer to a fixed choice of the k variables, i.e., $\bar{X}^k = \pi^{(k)}$.

Let $(\Omega, \mathcal{U})$ denote the space of all possible permutations of n objects and $\mathcal{U}$ is the uniform probability distribution. For $0 \leq i \leq n$, let $\mathcal{B}_i$ consist of *blocks* of permutations, partitioned into some equivalent classes according to the i-prefixes where $\mathcal{B}_0$ is a single block consisting of all permutations. In this context, let us define $Z_i = \mathbb{E}[Z|\bar{X}^{(i)}]$ for any well-defined random variable Z over the probability space $(\Omega, \mathcal{U})$ where the conditioning is over the blocks of $\mathcal{B}_i$. The sequence Z_i defines a martingale sequence that is widely known as a *Doob Martingale* [5, 7, 8]. The reader may recognize that these blocks form the basis of a nested *filter* sequence that formally defines a martingale sequence.

It is more intuitive to visualize the above process as a tree τ, where the level i nodes correspond to *blocks* of $\mathcal{B}_i$ with arity $n - i$ and each sub-block is connected to its parent block by an edge directed from the parent. Any node in the j-th level of this tree can be labelled by the (unique) sequence $X^{(j)}$ leading to it. An edge is labelled by the (random) choice made at that level and also has an associated weight $w(\tau(\bar{X}^{(i-1)}), \tau(\bar{X}^{(i)}))$ that corresponds to the cost of the i-th incremental step. Here $\tau(\bar{X}^{(i)})$ represents the node of the tree τ after adding/deleting a sequence of random variable $\bar{X}^{(i)}$, we have We will use $\bar{w}$ to denote an upper bound of $w()$ in the context of specific algorithms. Let $Y = \sum_{j=0}^{j=n} w(\tau(\bar{X}^{(j-1)}), \tau(\bar{X}^{(j)}))$ be a random variable that corresponds to the sum of the cost of the edges on a path that corresponds to the cost of the RIC. Let $Y_i = \mathbb{E}[Y|\mathbb{F}_i] = \mathbb{E}[Y|\bar{X}^{(i)}]$.

Even if we interpret the sequence as *deletion* sequence starting from $\{x_1, x_2, x_3\}$, the reader can easily verify that this preserves the nesting property of blocks, and hence a valid filter. For example, $\mathcal{B}_1$ would contain the blocks $(x_1x_2x_3, x_1, x_3, x_2)$ corresponding to deletion of x_1 and $\mathcal{B}_2$ would contain the block (x_1, x_2, x_3) when x_2 is the next element deleted and so on. Since every insertion sequence has a unique backward deletion sequence, this interpretation amounts to running the *RIC* in the *reverse* direction with each step associated with the same cost as the forward direction. This perspective is is very similar to the idea of *backward analysis* [21].

In the *Doob's martingale* sequence, Y_0 denotes the expected running time of the RIC. We would like to bound the deviation $|Y_n - Y_0|$ for any run of the algorithm with *high probability*, which in the context of this paper will be inverse polynomial, unless otherwise mentioned. Likewise the random deletion sequence also defines a *Doob's martingale* on the subsets which will be referred to as the *backward-sequence* martingale (BSM henceforth).

The following martingale tail bound is the basis of many later results in this paper which is distinct from Azuma's inequality and referred to as the *Method of bounded variance.*

▶ **Theorem 1** (Freedman [9]). *Let $X_1, X_2 \dots X_n$ be a sequence of random variables and let Y_k, a function of $X_1 \dots X_k$ be a martingale sequence, i.e., $\mathbb{E}[Y_k|X_1 \dots X_k] = Y_{k-1}$ such that $\max_{1 \leq k \leq n}\{|Y_k - Y_{k-1}|\} \leq M_n$. Let*

$$W_k = \sum_{j=1}^{k} \mathbb{E}[(Y_j - Y_{j-1})^2 | X_1 \dots X_{j-1}] = \sum_{j=1}^{k} Var(Y_j | X_1 \dots X_{j-1})$$

where Var is the variance using $\mathbb{E}[Y_j] = Y_{j-1}$. Then for all λ and $W_n \leq \Delta^2$, $\Delta^2 > 0$,

$$\Pr[|Y_n - Y_0| \geq \lambda] \leq 2\exp\left(-\frac{\lambda^2}{2(\Delta^2 + M_n \cdot \lambda/3)}\right)$$

Note that the term Δ^2 can be bound by $\sum_{j=1}^{n} \max_{X_1, X_2 \ldots X_j} Var(Y_j|X_1 \ldots X_{j-1})$ i.e., the worst case bounds over all choices of length j prefix $X^{(j)}$. If the inner term can be bound by some function of j, say, $\omega(j)$, then we may obtain an upper bound on the probability of deviation for any sequence $\bar{X}^{(n)}$ as $\sum_{j=1}^{n} \omega(j)$ which can be viewed as a function of n.

Further, we will actually use a minor variation of this result (see [6, 13]). Suppose $\Pr[M_n \geq g(n)] \leq \frac{1}{f(n)}$ for some non-decreasing functions g, f. If we let B denote the event $M_n \geq g(n)$, then the overall bound becomes

$$\Pr[|Y_n - Y_0| \geq \lambda] \leq 2\exp\left(-\frac{\lambda^2}{2(\Delta^2 + g(n)\lambda/3)}\right) + \Pr(B) \tag{1}$$

Similarly it can also be extended to the case where $W_n \leq \Delta^2$ holds with probability $1 - \frac{1}{f(n)}$. Henceforth, in the remaining paper, we will appeal to the following version of Freedman's inequality where the bounds on M_n and W_n hold with high probability. Often the term $\frac{1}{f(n)}$ will be the dominant term, so the final tail bound will effectively be $O(\frac{1}{f(n)})$.

▶ **Corollary 2.** *Let* $\Pr[\max_{1 \leq k \leq n}\{|Y_k - Y_{k-1}|\} \geq M_n, W_n \geq \Delta^2] \leq \frac{1}{f(n)}$, *then,*

$$\Pr[|Y_n - Y_0| \geq \lambda] \leq 2\exp\left(-\frac{\lambda^2}{2(\Delta^2 + M_n \cdot \lambda/3)}\right) + O(1/f(n))$$

3 Application to Quicksort and related problems

Let us consider quicksort in the *RIC* framework and without loss of generality, let the input elements be $\{1, 2 \ldots n\}$. The j-th pivot, $1 \leq j \leq n$, partitions the input into $j+1$ ordered sets, by splitting some existing partition P. Any element $x \in P$ is charged the cost of comparison with the pivot - any element $x' \notin P$ is not charged. The running time of the algorithm can be bound by the cumulative charges accrued by each element. In this analysis we will bound the charge of each element *with high probability* (w.h.p. henceforth) to denote probability exceeding $1 - 1/n^{\alpha}$ for some appropriate constant $\alpha > 0$, and the overall running time bound follows from multiplying by n.

The associated weight with each edge is either 1 or 0 depending on whether the latest random choice is one of the boundary elements of the interval containing x. We define a random variable $I_j^x = \{1$ if interval containing x changes in step j 0 otherwise$\}$. From backward analysis, the probability of this is at most $\frac{2}{j}$ for a uniformly chosen child node[1]. For completeness, we have included a detailed description of backward analysis in the appendix. We will also omit the superscript x and just use I_j since we will obtain a worst case bound over all choices of x. The reader may note that the bound on $\mathbb{E}[I_j]$ is only a function of j and not $\bar{X}^{(j)}$ over all random choices of *any* prefix of j elements.

[1] Using a simple trick of circular ordering (see [21]), this probability can be made exactly equal to $\frac{2}{j}$. Subsequently, Chernoff bounds can be applied easily by arguing about te independence of I_js.

It will also help to focus on the BSM for quicksort. A random deletion sequence creates a nested sequence of random subsets starting from the all the elements and ending in the empty sequence. An edge of this tree $(K, K - \{y\})$ is given a value 1 for a subset K and an element $y \in K$ if in the (forward) quicksort algorithm, selecting y as a pivot and leading to K (all the pivots selected) forces a comparison between y and x. Clearly two edges from any subset will be given a value 1, so that the expected cost for a random deletion is $\frac{2}{n-j}$ in the j-th level, $n \geq j \geq 0$. Figure 2 gives a depiction of this random variable in the quicksort process.

Consider a path $\mathcal{P} = v_0 v_1 \ldots v_n$ from root to a leaf-node in this tree. The cost of this path is given by $w(\mathcal{P}) = \sum_{i=1}^{n} w(v_i, v_{i+1})$. A *random* path corresponds to one where v_{i+1} is a child of v_i chosen uniformly at random among the $n - i$ children. The expected cost of such a random path is given by

$$\mathbb{E}_{random\ \mathcal{P}}[w(\mathcal{P})] = \mathbb{E}[\sum_{i=1}^{n} w(V_i, V_{i+1})] \text{ where } V_{i+1} \text{ is a random child of node } V_i$$

We will follow the convention that small letters will denote fixed choices, i.e., v_i where the capital letters will correspond to random variables, i.e., V_i. Let $\mathbb{E}_j[Z]$ denote $\mathbb{E}_{X_j}[Z|\bar{X}^{(j-1)}]$ for some random variable Z. Note that $\bar{X}^{(j-1)}$ represents a fixed path from the root of the tree τ corresponding to the deletion sequence $X_1 X_2 \ldots X_{j-1}$, to a level $j-1$ node, say V_{j-1}. Then,

$$\mathbb{E}_j[Y] = Y_j = \sum_{k=0}^{j-1} w(v_k, v_{k+1}) + \mathbb{E}_j\left[\sum_{k=j}^{n-1} w(V_k, V_{k+1})\right] = \sum_{k=0}^{j-1} w(v_k, v_{k+1}) + \sum_{k=j+1}^{n} \mathbb{E}[I_k]$$

It follows that $Y_0 = 2H_n$ and we want to obtain a tail estimate for $Y_n - Y_0$.

So,

$$Y_j - Y_{j-1} = w(v_{j-1}, v_j) + \left(\sum_{k=j+1}^{n} \mathbb{E}[I'_k]\right) - \left(\sum_{k=j}^{n} \mathbb{E}[I_k]\right) \tag{2}$$

$$= I_j - E[I_j] \text{ assuming } I_j, I'_j\text{'s have the same distribution} \tag{3}$$

To see this, the reader may recall that the probability that a fixed element x is affected by the pivot is a function only of the number of elements deleted (in the backward sequence) and not on the elements themselves or how they are distributed. So $\mathbb{E}_j[(Y_j - Y_{j-1})^2] = \mathbb{E}_j[(I_j - E[I_j])^2]$ which shows that the value of Y_j differs from Y_{j-1} because of the specific choice of the random variable X_j.

For quicksort, we can complete the analysis as follows.

$$\mathbb{E}_j[(I_j - E[I_j])^2] = \mathbb{E}_j[I_j^2] - \mathbb{E}_j^2[I_j] \ \leq \mathbb{E}_j[I_j^2] - \frac{4}{(n-j)^2}$$

$$\leq \frac{2}{n-j} \text{ since } I_j^2 \text{ is also a 0-1 indicator rv}$$

So $\sum_{j=1}^{n} \mathbb{E}_{j-1}[(Y_j - Y_{j-1})^2] \leq \sum_{j=1}^{n} \frac{2}{n-j} \leq 2H_n$where $H_n \leq \log n$. Plugging in $\lambda = 2c \log n$ for some constant c and using Freedman's theorem, we obtain

$$\Pr[|Y_n - Y_0| \geq c \log n] \leq \exp\left(-\frac{4c^2 \log^2 n}{2(\log n + c \log n/3)}\right) \leq \frac{1}{n^c}.$$

Note that $M_n = Y_i - Y_{i-1} \leq 1$.

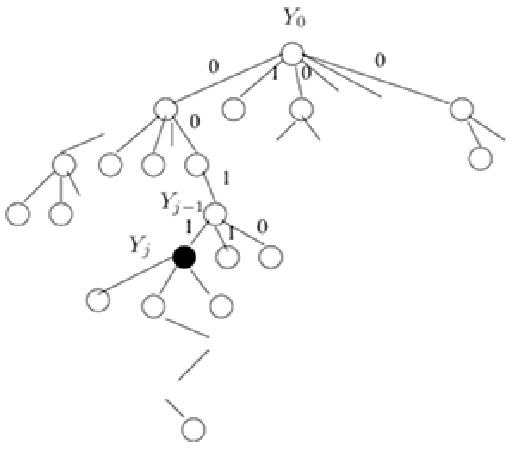

Figure 2 Tree corresponding to the Backward Sequence Martingale corresponding to the comparisons for a fixed element x. The root corresponds to Y_0 which denotes the expected running time. Every edge has cost 0 or 1 depending on whether x and X_i belong to the same interval and a path in this tree reveals the indicator variables I_j.

This shows that a single element incurs at most $O(\log n)$ cost with high probability and therefore quicksort runs in $O(n \log n)$ time with high probability.

▶ Remark. A straightforward application of the classic Azuma-Hoeffding bound [16] $\Pr[|Y_n - Y_0| \geq t] \leq \exp\left(\frac{-t^2}{\sum_{i=1}^{n} c_i^2}\right)$ would not have been effective since the the bound $c_i = M_n = 1$ makes the denominator too large for an $O(\log n)$ deviation bound. In [21], the author obtained a similar bound by using Chernoff bounds for binomial distribution that require *independence* of I_js across different levels. Also note that, there exists a superior bound of $O(n^{-\Omega(\log\log n)})$ for Quicksort obtained in [14].

The above argument can be directly extended to obtain a concentration bound on the *dart throwing* game that has many applications (Mulmuley [18]). Consider throwing n darts randomly in n ordered locations, say numbered $\{1, 2 \ldots n\}$. Let $S(i)$ be a random variable that denotes the *smallest* numbered location among the first i randomly thrown darts. Let $Z(i) = 1$ if $S(i) \neq S(i-1)$ and $Z(1) = 1$. So $Z(i)$ is the number of times $S(i)$ changes among the first i darts thrown. We are interested in $\mathbb{E}[Z(n)]$ which can be shown to be $\sum_{i=1}^{n} \frac{1}{i} = H_n$, the n-th harmonic. This follows from *backward analysis* by observing that among a set of i randomly chosen numbers, the probability of picking the smallest number as the last number is $\frac{1}{i}$. This is related to many visibility problems in geometry as well as the analysis of Trieps. Using the Freedman's inequality, we can easily show the following from the previous argument and looking at the changes in the leftmost interval induced by the darts.

▶ **Corollary 3.**

$$\Pr[|Z(n) - H_n| \geq 0.9 \log n] \leq \exp(-0.7 \log n) \leq \frac{1}{n^{0.7}}$$

This implies that $\Pr[0.1 \log n \leq Z(n) \leq 1.9 \log n] \geq 1 - n^{-0.7}$. The above result has been stated in a slightly weaker manner so that we can claim a lower bound on $Z(n)$ that will be invoked later to show the limitations of *RIC* .

The analysis in this section also extends to problems like constructing trapezoidal maps that can be used for point location (Seidel [19]). Since a trapezoid can be defined by at most 4 segments, the expected work for point location is $\sum_{i=1}^{n} \frac{4}{j} \leq 4 \log n$. Using a straightforward extension of the previous arguments, the following result can be obtained.

▶ **Lemma 4.** *Given a set of n non-intersecting line segments, a trapezoidal map can be constructed using RIC such that for any query point q, the number times the trapezoid containing q changes can be bound by $O(\log n)$ with inverse polynomial probability.*

This result will turn out to be very useful for some later results.

3.1 Extension to more general cost function

The bound obtained in Equation 3 can be extended to a more general situations of *RIC* where a single change can affect multiple "intervals" (more precisely, configurations). More specifically, when $\bar{w}$ not bounded by a constant we have the following generalization as long as $\mathbb{E}[w(V_{j-1}, V_j)]$ are same across all nodes in level $j-1$ for a random choice of the next node. Let $\mathcal{W}_j = w(V_{j-1}, V_j)$, then by generalizing the calculations in Equation 3, we obtain

$$\mathbb{E}_j[(Y_j - Y_{j-1})^2] \leq \mathbb{E}_j[\mathcal{W}_j^2] - \mathbb{E}_j^2[\mathcal{W}_j] \leq \mathbb{E}_j[\mathcal{W}_j^2] \tag{4}$$

Although the generalized analysis of *RIC* is not described here is details, we appeal to the intuitions of the reader that the expected cost of the j-th step depends on j and not the actual choice of the elements - see Equation 5. Note that in the general analysis of *RIC* , we obtain an upper bound on the $\mathbb{E}[\mathcal{W}_j]$ as a function of j. The upper bounds can be considered as the (identical) expected cost of the i-th step and the martingale bounds can be applied on these costs, so the final bounds would still hold as deviation from this (uniform) expected upper-bounds.

3.2 Comparison with an earlier bound

We briefly recall the framework of Mehlhorn, Sharir and Welzl [15] to model the general RIC algorithm. A rooted (n, r) tree T is either a single node for $r = 0$ or (for $r > 0$) the tree has n children which are recursively defined $(n-1, r-1)$ subtrees. Each of the n edges has an associated weight d_i corresponding to the i-th child and $\max_{i=1}^{n} d_i \leq d(n)$ and $\sum_i d_i \leq M(n)$. The expected cost of a path in this recursively defined tree is $A = \sum_{i=1}^{n-1} \frac{M(n-i)}{(n-i)}$. One of the main results in the paper is the following tail bound (Theorem 1 in [15]).

$$\Pr(X \geq B) \leq \left(\frac{e}{1 + B/A} \right)^{B/d(n)} \quad \text{for all } B \geq 0$$

Although this bound looks somewhat simpler to use, this is not directly comparable to Freedman's bound except for some special cases like Lemma 4 and quicksort where the concentration results are similar. It may be noted that the authors [15] analyze the backward execution of the algorithm for these results. This bound becomes weaker if $d(n)$ is not a constant - for some of the later applications $d(n)$ may be larger than A in the worst case. The authors improve the bound for the specific problem of building visibility maps of line segments by using the expected value of $M(n)$. However, there is no generalization given for other problems.

4 Incremental Delaunay Triangulation

We will now consider somewhat more complex scenarios like construction of Delaunay Triangulation and three dimensional convex hull (see Guibas Knuth and Sharir [10]). Broadly speaking these algorithms have two distinct components -

(i) Updating the (partial) structure of the points inserted thus far.
(ii) Updating the point-location data structure of the uninserted points.

For concreteness, we will address the problem of Delaunay Triangulation. The analysis corresponding to updating the point location structure is similar to the analysis of quicksort given above. For the update of structural complexity, it was shown in [10] that the expected cumulative structural change can be bound by $O(n)$, whereas for the latter, the expected work over all the n (random) insertions sequence $O(n \log n)$. We will do a combined analysis since we are interested in obtaining tail estimates on the work including all data structural updates.

In the remaining part of the paper, we will be alluding to the BSM framework and make use of Equation 4 for deriving the tail estimates. To avoid any confusion, we will use stage/level k to refer to the forward algorithm when k objects have been added and do all calculations in this order. Although the martingale has been defined for the backward execution, substituting $n-k$ by k, consistently will not not affect anything except the order of the summations. This will also help us use the random sampling bounds without having to restate them in the flipped order.

We will make extensive use of the following result of [4, 11].

▶ **Theorem 5.** *At any stage i of the RIC of Delaunay triangulation, the i randomly chosen points is a uniform random subset of the n points. So the number of unsampled points within each triangle is bounded by $O(\frac{n}{i}\log n)$ with probability $1-1/n^c$ for any constant $c > 1$. Moreover, all the triangles that emerges in the course of edge flips also satisfy the above bounds.*

▶ Remark.

(i) A random sample of size $r+1$ is constructed by adding a random element to a sample of size r. So the properties of random sampling applies to the intermediate steps as well with high probability. In general, we will use a similar bound on $\max_{\sigma\in\Pi^0(R)} |\ell(\sigma)|$. This is crucial for the application of the generalized version of Freedman's inequality given in Equation 1 where the event B contains all insertion sequences where the bound in Theorem 5 fails during one of more stages. In other words, the complement of B consists of all insertion sequences where the bound holds during all stages.

(ii) All the triangles that show up in the course of edge flips belong to $\Pi^0(R)$. Although some of them are not Delaunay triangles and therefore, only temporary, they can contribute to the running time, depending on the data structure one maintains for the intermediate partitions.

To apply Freedman's bound, we will bound the variance. Unlike the analysis of quicksort, we will consider the work done for all the n points (actually $n-i$ uninserted points in stage i) together. Each edge flip involves four triangles - two old and two new and redistributes the points in the two new triangles. Since each triangle contains $O(\frac{n}{i}\log n)$ points w.h.p, each edge flip can be be done in $O(\frac{n}{i}\log n)$ w.h.p. Since the maximum degree of a Delaunay graph of i points is i, the total number of edge flips in the i-th stage is bounded by i. Therefore we can claim the following.

▶ **Lemma 6.** *The work in stage i of the algorithm, $i \leq n$ can be bound by $O(n \log n)$ w.h.p.*

Let $\Pi_s(R)$ denote the configurations in $\Pi^0(R \cup s)$ adjacent to s (or defined by s). The following claims can be easily derived from some general random-sampling lemmas in [4]

▶ **Lemma 7.** *(i)* $\mathbb{E}[\sum_{\sigma \in \Pi_s(R)} \ell(\sigma)] = O(\frac{n}{r})\mathbb{E}[|\Pi_s(R)|]$
(ii) $\mathbb{E}[\sum_{\sigma \in \Pi_s(R)} \ell^2(\sigma)] = O(\frac{n^2}{r^2})\mathbb{E}[|\Pi_s R)|]$

Bounding Variance. We will need the following result

▶ **Lemma 8.** *For real numbers x_i $1 \leq i \leq r$* $\left(\sum_{i=1}^r x_i\right)^2 \leq r\left(\sum_{i=1}^r x_i^2\right)$

Proof. Using the convexity of the square function, from Jensens inequality it follows that $\frac{\sum_{i=1}^r x_i^2}{r} \geq \left(\frac{\sum_{i=1}^r x_i}{r}\right)^2$. Multiplying both sides by r^2 yields the required result. ◀

Let R^k denote the random subset of the first k sites[2] and let $DT(R^k)$ represent the Delaunay triangulation of R^k which is a planar graph having $2k - h_k - 2$ triangles and $3k - h_k - 3$ edges where h_k is the number of points on the convex hull of R^k. The work done when a site $v \in DT(R^k)$ is picked by the *RIC* , is proportional to the number of unsampled points in the triangles adjoining v. If $l(\sigma)$ is the number of points in a triangle σ, then the work is proportional to $T_k = \sum_{\sigma \in \Delta(v)} l(\sigma)$ where $\Delta(v)$ denotes triangles adjacent to v. Squaring T_k and taking expectation

$$\begin{aligned}
\mathbb{E}[T_k^2] &= \frac{1}{k}\mathbb{E}[\sum_{v \in R^k}(\sum_{\sigma \in \Delta(v)} l(\sigma))^2] \leq \frac{1}{k}\mathbb{E}[\sum_{v \in R^k}\sum_{\sigma \in \Delta(v)} |\Delta(v)| l^2(\sigma)] \text{ from Lemma 8} \\
&= \frac{1}{k}\sum_{v \in R^k} |\Delta(v)| \mathbb{E}[\sum_{\sigma \in \Delta(v)} l^2(\sigma)] \leq \frac{1}{k}\sum_{v \in R^k} \frac{n^2}{k^2}|\Delta(v)|^2 \text{ from Lemma 7} \\
&= O(\frac{n^2}{k}) \text{ as } \sum_v |\Delta(v)|^2 = O(\sum_v \Delta(v))^2) = O(k^2)
\end{aligned}$$

Following Equation 4, this yields $W_n \leq \sum_{k=1}^{k=n} \mathbb{E}[T_k^2] \leq \sum_{k=1}^{k=n} O(\frac{n^2}{k}) = O(n^2 \log n)$. Plugging the bound of $M_n = O(n \log n)$ from Lemma 6 in Freedman's theorem, we obtain the following bound.

▶ **Lemma 9.** *Let $T(n)$ denote the running time of ric based construction of Delaunay Triangulation and let $\lambda = cn \log n$ for a suitable constant c. Then*

$$\Pr[T(n) \geq \alpha(n)\lambda] \leq \exp\left(-\frac{(\alpha(n)cn\log n)^2}{2(n^2\log n + \alpha(n)c \cdot n^2 \log^2 n/3)}\right) \leq \exp(-\alpha(n))$$

▶ Remark. The above Lemma gives high probability bound for $T(n)$ exceeding $\Omega(n \log^2 n)$ for $\alpha = \Omega(\log n)$. However, this bound is superior to the straightforward Markov's bound applied on the expected work as well as preferable to restarting the original algorithm using independent random bits each time.

[2] We will use this term to distinguish between the input points defining the triangulation and the unadded points

This analysis can be extended to the three-dimensional convex hull algorithm presented in Mulmuley [17]. In [3], the authors obtain similar tail estimates for the for the space complexity (alternately referred to as *conflict history*) of the algorithm of [10]. For fixed dimensional linear programming Seidel [20] proved a similar property and this can be extended to *RIC* algorithms like closest pair [12]. These algorithms typically have the property, that in stage i, with probability $\Omega(\frac{1}{i})$, the algorithm re-builds the data structure. This makes inverse polynomial bound challenging - say for $i = \sqrt{n}$, the *RIC* for Delaunay triangulation could spend $O(n \log(\sqrt{n})$ time to rebuild the associated point location data structure if the i-th point has degree $\Omega(i)$.

5 More generalized RIC: segment intersections

We now consider a more general scenario in RIC (Randomized Incremental Construction). Using a *conflict graph* update model of RIC (see Appendix), we obtain the following expression for expected work.

$$\mathbb{E}[\ \#\text{edges created in the conflict graph}] = \sum_{\sigma \in \Pi^0(R \cup s)} l(\sigma) \cdot \Pr\{\sigma \in \Pi^0(R \cup s) - \Pi^0(R)\}$$

From *backward analysis* this probability is the same as deleting a random element from $R \cup s$ which is $\frac{d(\sigma)}{r+1}$ where $r = |R|$. By substituting this we obtain

$$\sum_{\sigma \in \Pi^0(R \cup s)} l(\sigma) \cdot \frac{d(\sigma)}{r+1} = \frac{d(\sigma)}{r+1} \sum_{\sigma \in \Pi^0(R \cup s)} l(\sigma) = O(\frac{d(\sigma)}{r} \cdot \frac{n}{r} \mathbb{E}[\Pi^0(R \cup s)]) \tag{5}$$

Therefore the expected work over the sequence of random insertions is $\sum_{r=1}^{n} O(\frac{d(\sigma)}{r} \cdot \frac{n}{r} \mathbb{E}[\Pi^0(R \cup s)])$.

For the case of line segment intersections, it can be shown that $\mathbb{E}[\Pi^0(R \cup s)] = O(r + \frac{m \cdot r^2}{n^2})$ from which it follows that the expected work is

$$\sum_{r=1}^{n} O(\frac{d(\sigma)}{r} \cdot \frac{n}{r} \cdot O(r + \frac{m \cdot r^2}{n^2}) = \sum_{r=1}^{n} \left(\frac{dn}{r} + \frac{dm}{n} \right) = O(n \log n + m).$$

Here $d(\sigma) \leq 6$ which the maximum number of segments that define a σ (trapezoid in this case).

Tail bounds for this problem has been elusive despite some significant attempts (see [15]). We will show that our previous methodology can be extended to obtain tail estimates on the work done.

Consider an arrangement of n segments with m intersections ($0 \leq m \leq \binom{n}{2}$). In the trapezoidal map $\mathcal{T}$ of the n segments (also known as a vertical visibility diagram), let us denote the set of trapezoids adjacent to segment s_i by T_i. Any trapezoid $\sigma \in \mathcal{T}$ is defined by at most six segments. Since it is a planar map, and there are at most $2n + 2m$ vertices, it follows that $\sum_i |T_i| = O(n + m)$. We would like to obtain a bound on $\sum_i |T_i|^2$. Let us denote by n_i and m_i respectively, the number of segments end-points and intersection points visible to segment s_i. It follows that $|T_i| = O(n_i + m_i)$ and

$$\sum_i |T_i|^2 = O(\sum_i (n_i + m_i)^2) = O(\sum_i n_i^2 + \sum_i n_i \cdot m_i + \sum_i m_i^2)$$

where $m_i \leq O(n)$ from the zone theorem bound. Moreover $\sum_i n_i = O(n)$ and $\sum_i m_i = O(m)$ as each point is visible from the closest segments above and below.

The first expression can be bound by $(\sum_i n_i)^2 = O(n^2)$ and the second expression by $2(\sum_i n_i) \cdot (\sum_i m_i) = O(n \cdot m)$ (Cauchy-Schwartz inequality). The third expression is less than $(m/n) \cdot n^2 = mn$. So, the overall expression can be bound by $O(m \cdot n + n^2)$.

For a uniformly chosen sample R^k of size k, the expected number of intersections in the sample is $\frac{mk^2}{n^2}$, so the variance can be bound by

$$\mathbb{E}[T_k^2|R^k] = \frac{1}{k}\mathbb{E}[\sum_{s_i \in R^k} (\sum_{\sigma \in T_i} l(\sigma))^2]$$

To simplify calculations, we recall (Theorem 5) that $l(\sigma) \leq O(\frac{n \log n}{k})$ with high probability. So, plugging this in the previous expression, and using the previous bound on $\sum_i |T_i|^2$, we obtain (w.h.p.)

$$\mathbb{E}[T_k^2|R^k] \leq \frac{1}{k} \cdot O(\frac{n^2 \log^2 n}{k^2}) \cdot \mathbb{E}[k^2 + k \cdot m_k]$$

where m_k is the number of intersections in R^k. Taking expectation over all choices of R^k, we obtain the unconditional expectation as

$$\mathbb{E}[T_k^2] \leq \frac{n^2 \log^2 n}{k^3} \cdot \mathbb{E}[k^2 + km_k] \leq \frac{n^2 \log^2 n}{k^3} \cdot (k^2 + \frac{mk^3}{n^2})) \leq \frac{n^2 \log^2 n}{k} + m \log^2 n$$

This uses the bound $\mathbb{E}[m_k] = O(k + \frac{m \cdot k^2}{n^2})$. This bound is relevant for the maintenance of conflict graphs.

In contrast, for an algorithm like Mulmuley [18], where only the trapezoids are maintained, the work done[3] can be bound by using $l(\sigma) = 1$ in the expression for $\mathbb{E}[T_k^2]$. This yields $\mathbb{E}[T_k^2] = O(k + m\frac{k^2}{n^2})$. We shall return to this case later.

So

$$W_n \leq \sum_{k=1}^{n} \mathbb{E}[T_k^2] \leq O(n^2 \log^3 n + mn \log^2 n)$$

To obtain high probability bounds using Freedman's inequality, we want to bound this expression by $\frac{\lambda^2}{\log n}$ where $\lambda = c(n \log n + m)$. M_n and $M_n \cdot \lambda$ can be bound by $O(n \log n \cdot \alpha(n))$ and $mn\alpha(n)$ respectively with high probability. This follows from a bound of $O(t\alpha(t))$ on the zone of a segment that intersects t segments in an arrangement of n segments ([18]).

So $\frac{\lambda^2}{W_n + M_n \cdot \lambda}$ can be bound by

$$\frac{\Omega(m^2 + mn \log n + n^2 \log^2 n)}{O(mn \log^2 n + n^2 \log^3 n + mn\alpha(n) \log n + n^2 \alpha(n) \log^2 n} = \frac{\Omega(m^2 + mn \log n + n^2 \log^2 n)}{O(mn \log^2 n + n^2 \log^3 n}.$$

So from Freedman's inequality we obtain a tail bound of $\exp(-\frac{m^2}{mn \log^2 n})$ for $m \geq n \log^2 n$.

▶ **Theorem 10.** *Let $T(n)$ represent the work done in the conflict-graph based segment intersection algorithm, then there exists constant β, such that for $m \geq \beta n \log^2 n$, $\Pr[T(n) \geq m] \leq \exp(-\frac{m}{n \log^2 n})$.*

To the best of our knowledge, no prior concentration bound was known for the conflict-graph based approach for segment intersection given by Clarkson and Shor [4]. The paper by [15] noted that their methods could not be extended to this algorithm.

[3] there is some additional cost for point location that can be bound using Lemma 4

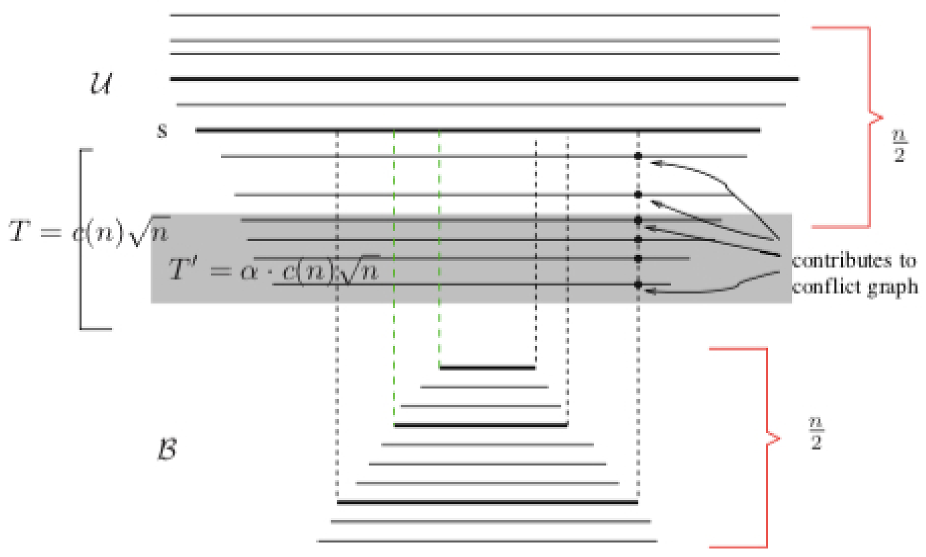

Figure 3 A bad input for segments intersections (trapezoidal maps). The thicker segments correspond to the sampled set.

For the specific case of $m = 0$, the bound can be improved by observing that the zone of a segment can be at most $O(n)$ (instead of $n\alpha(n)$) as there are no intersections. Setting $m = 0$ in the previous bound for W_n, we obtain the following

▶ **Corollary 11.** *For constructing the trapezoidal map of n non-intersecting line segments using RIC , the work done $T(n)$ satisfies $\Pr[T(n) \geq c\beta n \log^2 n] \leq n^{-\beta^2}$ for some constant $c > 1$.*

This shows that we can obtain inverse polynomial concentration bounds around a running time that exceeds the expected running time by a factor of $\Omega(\log n)$.

We now return to the algorithms of [18] and [4] that do not maintain conflict-graphs but only involves segment end-points. As observed before, the quantity W_n can be bound by $\sum_{k=1}^{n} O(k + m\frac{k^2}{n^2}) = O(n^2 + mn)$. We summarize as follows.

▶ **Lemma 12.** *In the segment intersection algorithms of [18, 4] that do not maintain conflict-graphs explicitly, the probability that the work exceeds $c(m + n\log n)$ can be bound by*

$$\exp - \left(\frac{\Omega(m^2 + mn\log n + n^2\log^2 n)}{O(mn\alpha(n) + n^2\alpha(n)\log n)} \right) \leq \exp(-\frac{\log n}{\alpha(n)}) \text{ since } M = O(n\alpha(n)).$$

For $m \geq n\log n$, the bound improves to $\exp(-\frac{m}{n\alpha(n)})$.

▶ Remark. This bound is better than the results in [15] where the authors show that for some constant $\delta > 0$. $\Pr[T(n) \geq Cm] \leq \exp\left(\frac{-\delta m}{n\log n}\right)$ for $m \geq n\log n\log\log\log n$.

6 Can we improve the tail bounds

Figure 3 shows an input of n horizontal segments divided into two sets $\mathcal{U}, \mathcal{B}$ each of which has $n/2$ segments. In the top pile $\mathcal{U}$ of the $n/2$ horizontal segments, let T denote the lowest $\sqrt{n} \cdot c(n)$ segments for some function $c(n)$ that will determined in the analysis.

We will consider the *RIC* after the first $3\sqrt{n}$ insertions. We want to consider the event that at least $\sqrt{n}$ segments are chosen from $\mathcal{B}$ and no segment is chosen from T. The probability of the former is $1 - 2^{-\sqrt{n}}$ (from Chernoff bounds) whereas the probability of the latter can be easily seen as $(1 - \frac{c(n)}{\sqrt{n}})^{3\sqrt{n}} = \Omega(4^{-3c(n)})$, using $(1-x) \geq \exp -(x+x^2) \geq \frac{1}{4^x}$ $x \in [0, 1/2]$. Since the two events are independent, the probability of their intersection, which will be denoted by the event $\mathcal{E}_1$ is nearly $\Omega(4^{-3c(n)})$ (since $1 - 2^{-\sqrt{n}} \to 1$).

Notice that, if the lowest sampled segment is $s \in \mathcal{U}$ then all the segments below s will intersect the vertical lines through the end-points of the trapezoids defined by the sampled segments in $\mathcal{B}$. Namely, if there are m unsampled segments below s, then the size of the conflict graph is at least $\sqrt{n} \cdot m$. Every time s changes, new edges are created in the conflict graph by the sampled edges in $\mathcal{B}$. Following the choice of the initial $3\sqrt{n}$ segments, let us consider subsequent sampling by the *RIC* where segments from T will be sampled. Within T, let us denote by T' the lowest $\alpha c(n)\sqrt{n}$ segments for some constant $\alpha < 1$ and let T'' denote the remaining segments. Let $\mathcal{E}_2$ denote the event that among the first $\log n$ segments sampled from T, none are from T'. This can be calculated as $(1-\alpha)^{\log n} = \Omega(4^{-\alpha \log n})$.

From our earlier analysis, the lowest sampled segment in T'' changes about $\theta(\log\log n)$ times with probability $\frac{1}{\log n}$ (Corollary 3) - note that this holds regardless of the event $\mathcal{E}_2$. So, in the second phase of *RIC* , at least $\Omega(\alpha\sqrt{n}c(n) \cdot \sqrt{n}\log\log n)$ edges are created with probability $\Omega(\frac{2^{-\alpha \log n}}{\log n})$.

So, by unconditioning the first phase event $\mathcal{E}_1$, we obtain
$\Pr[\mathcal{E}_1 \cap \mathcal{E}_2] = \Pr[\mathcal{E}_2 | \mathcal{E}_1] \cdot \Pr[\mathcal{E}_1] = \Omega(\frac{1}{\log n} n^{-\alpha} \cdot 2^{-6c(n)})$. This yields the following result

▶ **Theorem 13.** *There exists inputs for which the conflict-graph based RIC algorithm for constructing vertical visibility maps (segment intersections with no intersections) encounters* $\Omega(\alpha c(n)\sqrt{n}\log\log n)$ *structural changes with probability* $\Omega(e^{-6c(n)-\alpha})$ *for* $c(n) = o(\sqrt{n})$.

In particular, by choosing $c(n) = \frac{\log n}{13}$, and a small α, the conflict graph based *RIC* algorithm may encounter $\Omega(n \log n \log\log n)$ changes with probability $\Omega(\frac{1}{\sqrt{n}})$ that rules out inverse polynomial bounds for a total work of $O(n \log n)$. Note that $\frac{1}{\sqrt{n}}$ can be easily increased to $\frac{1}{n^\epsilon}$ for any $\epsilon > 0$ by an appropriate choice of $c(n)$.

The reader may compare this result with Corollary 11 to get a sense of the gap between upper and lower bounds of the tail estimates. Coming up with similar examples for the other problems discussed in this paper, like Delaunay Triangulation would be an interesting exercise and shed more light on the behavior of *RIC* .

References

1 P. Chew. The simplest Voronoi diagram algorithm takes linear expected time. In *Manuscript*, 1988.

2 Kenneth L. Clarkson. Applications of Random Sampling in Computational Geometry, II. In *Symposium on Computational Geometry*, pages 1–11. ACM, 1988.

3 Kenneth L. Clarkson, Kurt Mehlhorn, and Raimund Seidel. Four Results on Randomized Incremental Constructions. In *STACS 92, 9th Annual Symposium on Theoretical Aspects of Computer Science, Cachan, France, February 13-15, 1992, Proceedings*, pages 463–474, 1992.

4 Kenneth L. Clarkson and Peter W. Shor. Application of Random Sampling in Computational Geometry, II. *Discrete & Computational Geometry*, 4:387–421, 1989.

5 J.L. Doob. Regularity properties of certain families of chance variables. *Transactions of the American Mathematical Society*, 47 (3):455—-486, 1940.

6 Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press, 2009.

7 W. Feller. *An Introduction to Probability Theory, Vol. 1*. Wiley, New York, NY, third edition, 1968.

8 W. Feller. *An Introduction to Probability Theory and Its Applications, Vol. 2*. Wiley, New York, NY, second edition, 1971.

9 David Freedman. On tail probabilities on martingales. In *The Annals of Probability*, volume 3(1), pages 100–118, 1975.

10 Leonidas J. Guibas, Donald E. Knuth, and Micha Sharir. Randomized Incremental Construction of Delaunay and Voronoi Diagrams. *Algorithmica*, 7(4):381–413, 1992.

11 David Haussler and Emo Welzl. Epsilon-Nets and Simplex Range Queries. In *Proceedings of the Second Annual ACM SIGACT/SIGGRAPH Symposium on Computational Geometry, Yorktown Heights, NY, USA, June 2-4, 1986*, pages 61–71, 1986.

12 S. Khuller and Y. Matias. A Simple Randomized Sieve Algorithm for the Closest-Pair Problem. *Information and Computation*, 118(1):34–37, 1995.

13 C. McDiarmid. Concentration. *Probabilistic Methods for Algorithmic Discrete Mathematics*, 16, Algorithms and Combinatorics:195—-248, 1998.

14 Colin McDiarmid and Ryan Hayward. Large Deviations for Quicksort. *J. Algorithms*, 21(3):476–507, 1996.

15 Kurt Mehlhorn, Micha Sharir, and Emo Welzl. Tail Estimates for the Efficiency of Randomized Incremental Algorithms for Line Segment Intersection. *Comput. Geom.*, 3:235–246, 1993.

16 Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1995.

17 K. Mulmuley. *Computational Geometry: An Introduction Through Randomized Algorithms*. Prentice-Hall, 1994. URL: `https://books.google.com/books?id=rjgZAQAAIAAJ`.

18 Ketan Mulmuley. A Fast Planar Partition Algorithm, I (Extended Abstract). In *29th Annual Symposium on Foundations of Computer Science, White Plains, New York, USA, 24-26 October 1988*, pages 580–589, 1988.

19 Raimund Seidel. A Simple and Fast Incremental Randomized Algorithm for Computing Trapezoidal Decompositions and for Triangulating Polygons. *Comput. Geom.*, 1:51–64, 1991.

20 Raimund Seidel. Small-Dimensional Linear Programming and Convex Hulls Made Easy. *Discrete & Computational Geometry*, 6:423–434, 1991.

21 Raimund Seidel. Backwards Analysis of Randomized Geometric Algorithms. In *In: Pach J. (eds) New Trends in Discrete and Computational Geometry. Algorithms and Combinatorics*, volume 10. Springer, Berlin, Heidelberg, 1993.

A Appendix

We provide a brief description of the notations and definitions that capture the framework of RIC and its analysis in very general setting.

Given a set S of n elements (like points, segments, lines etc.), a configuration σ is defined by at most d objects where d is $O(1)$. The set of objects is denoted by $d(\sigma)$ and the number of configurations is bounded by n^d if there are no more than $O(1)$ configurations associated each subset of d elements (there can be more than one configuration associated with the same $d(\sigma)$ elements.

Let $\ell(\sigma) = S \cap \sigma - d(\sigma)$ be the elements that intersect with σ. With a slight overloading of notation we will also use $\ell(\sigma)$ to denote the set of the intersecting elements with σ also. Let $\Pi^i(S)$ denote the set of configurations σ with $\ell(\sigma) = i$. We use $\Pi(S) = \cup_i \Pi^i(S)$ to denote all configurations. For any subset $R \subset S$, we use $\Pi(R)$ to denote the configurations defined by elements of R and the *conflict list* of any configuration $d(\sigma) \subset R$ as $\sigma \cap S$, i.e., all the elements and not just the elements in R.

A *conflict graph* represents the relation between the configurations in $\Pi^0(R)$ and the corresponding conflict list, which is a bipartite graph with configurations in $\Pi^0(R)$ on one side and the uninserted elements on the other side. Randomized Incremental construction

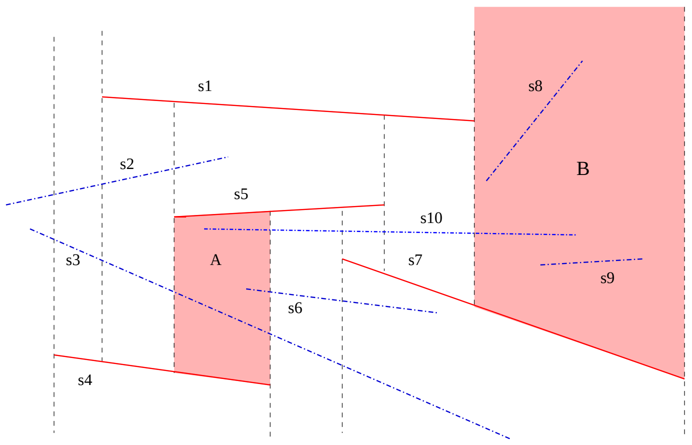

Figure 4 The red segments are sampled segments and blue segments are unsampled. The trapezoids A, B are configurations that belong to $\Pi^0(R)$. Here $d(A) = \{s_4, s_5\}$ $\ell(A) = \{s_3, s_6, s_{10}\}$.

can be thought of as maintaining and update of the conflict graph starting with $R = \phi$ and successively adding a random (uninserted) element $e \in S - R$ into R. This introduces $\sigma \in \Pi^0(R \cup e) - \Pi^0(R)$ requiring appropriate changes in the conflict graph. To illustrate this framework on qiucksort, we define the configurations as intervals defined by a pair of elements $[x_i, x_j]$ where $x_i < x_j$. Initially there is the hypothetical configuration $(-\infty, +\infty)$. As we introduce more pivots, we maintain the ordered set of intervals induced by the elements chosen as pivots. As we introduce a pivot, some interval is split. Eventually we have the sorted set defined by consecutive intervals. When an interval $[x_i, x_j]$ splits because of a pivot element y such that $x_i < y < x_j$, the elements in $\ell([x_i, x_j]) \cap S$ is reassigned to $\ell([x_i, y])$ and $\ell([y, x_j])$ appropriately. The number of comparisons required is roughly $|\ell([x_i, x_j]) \cap S|$ (the cardinality).

The analysis of quicksort in this framework can be done using the technique of *backward analysis* which is very elegant. Let us assign an indicator random variable X_k associated with an element x, such that

$$X_k = \begin{cases} 1 & \text{if } x \text{ is compared for the } k\text{-th pivot} \\ 0 & \text{otherwise} \end{cases}$$

The number of comparisons involving x is given by $\sum_{k=1}^{n} X_k$. Therefore

$$\mathbb{E}[\sum_{k=1}^{n} X_k] = \sum_{k=1}^{n} \mathbb{E}[X_k] = \sum_{k=1}^{n} p_k(x)$$

where $p_k(x)$ is the probability that element x is involved in the partitioning of the k-th pivot insertion.

To compute the probability, we observe that $X_k = 1$ iff the k-th pivot y is one of the two elements that bound the interval containing x after k pivots are chosen randomly. For a fixed choice of k initial pivots, the probability that y is one of the two bounding elements is at most $\frac{2(k-1)!}{k!} = \frac{2}{k}$. The numerator represents the number of permutations with one of the bounding elements being the last pivot. Although this is the probability conditioned on the choice of the first k pivots, clearly unconditioning would also give us the same probability. Therefore the expected number of comparisons involving x is $\sum_{k=1}^{n} \frac{1}{k} = O(\log n)$. Further the total expected number of comparisons is $O(n \log n)$ by summing over all elements.