Pseudorandomized Incremental Algorithms and Derandomization

Presented by: Jianyu Wang

Abstract

We will show that the bounds on the expected running times of most of the randomized incremental algorithm in computational geometry do not change by more than a constant factor when they are made pseudorandom. This reduces the number of random bits used by these algorithms from $\Omega(n \log n)$ to $O(\log n)$. Can we get rid of randomness altogether? We will give some basic idea for this purpose-derandomization.

1 Randomized Incremental Algorithms:

All randomized incremental algorithms in computational geometry have the following abstract form: Constructing a certain geometric partation $H(N)$ induced by a set N of "objects". A randomized incremental algorithm for constructing $H(N)$ has the following general form:

(1)*Construct a random sequence(permutation)* $S_1, S_2, ..., S_n$ *of the objects in* N.

(2)*Add the objects in* N *one at a time according to this sequence. Let* N^i *denote the set of the first* i *added objects. At the ith stage, the algorithm maintains the partition* $H(N^i)$ *formed by the set* N^i *and other necessary data structures. All these structures need to be update efficiently during the addition of a new object.*

It turns out that a large number of randomized algorithms of the above form have a good expected running time, provided the sequence of insertions constructed in the first step is random. In this talk, we will show how these algorithms will perform, if the sequence constructed in the first step is not truly random, but **pseudo-random**.

2 Pseudo-random sources:

A sequence $X_0, X_1, ..., X_{n-1}$ of random variables is called *t-wise independent*, if every t-tuple of distinct variables in the set is independent, but the knowledge of any t variables in the sequence may determine all the others(we assume each random variable X_i will take a value from the set of intergers [n]={0, 1, ..., n-1}).

Polynomial Congruential Generator(PCG):

Choosing t numbers(called seeds) $a_0, a_1, ..., a_{t-1}$ independently from a uniform distribution on [n], it need $t \log n$ bits. We assume n is prime(otherwise we can pad n). For $\forall i \in [n]$, let

$$X_i = a_0 + a_1 i + a_2 i^2 + \cdots + a_{t-1} i^{t-1} \pmod{n}$$

Proposition 1: For each i, X_i is uniformly distributed on $[n]$ and $X_0, X_1, ..., X_{n-1}$ is t-wise independent. Given $z \in [n]$, conditioning on $X_i = z$, each $X_j (j \neq i)$ is still uniformly distributed on $[n]$ and the sequence $X_0, X_1, ..., X_{i-1}, X_{i+1}, ..., X_{n-1}$ is still $(t-1)$-wise independent.

Assume that the set N of objects is given to us in the form of an array of length n, let $N[i]$ denote the object contained in the ith entry of this array. An algorithm to generate a pseudo-random N-sequence of additions is:

First, constructing a t-wise independent sequence $X_0, X_1, ..., X_{n-1}$ of numbers using PCG(or other pseudo-random sources), this need $t[\log n]$ random bits. Then we proceed as follows:

Algorithm(Generation of a pseudo-random N-sequence):

Main phase:

For $i = 0$ to n-1, do:

Add $N[X_i]$, if it is not already added.

Clean-up phase:

Add the remaining objects in N in any order.

Since some objects in N may not be added at the end of the main phase(X_i need not take distinct values), we need clean-up phase.

3 Basic Results:

Assuming we are given an abstract universe N of objects. A configuration σ over N is a pair (D, L), where D and L are disjoint subsets of N. We say that the elements in D, called *triggers*, define σ and that the elements in L, called *stoppers*, conflict with σ. Let $d(\sigma)$(degree of σ) to be the cardinality of $D(\sigma)$, $l(\sigma)$(the conflict size of σ, or the level of σ), to be the cardinality of $L(\sigma)$. For each $i \geq 0$, we define $\Pi^i(N)$ to be the set of configuration in $\Pi(N)$ with level i(i.e. the conflict size is i). Let $\Pi = \Pi(N)$ be any configuration space, $|N| = n$. Let $\bar{u}$ be any N-sequence of additions, N^i be the first i added objects in this sequence, σ be a fixed configuration in Π. We say that σ is active at time i during $\bar{u}$ if it is active over the set N^i(i.e. N^i contains all triggers but no stopper associated with σ). We say that σ is created during $\bar{u}$, if it is active at some time $i \leq n$. If i is the least such integer, we say that σ is created during $\bar{u}$ at time i. We say that σ is killed at time j, if σ is active over N^{j-1} but not over N^j. A created configuration during $\bar{u}$ can be destroyed only once, because $\bar{u}$ contains only additions.

Lemma 1: If the sequence of additions is truly random permutation of N, then the probability that σ is created during $\bar{u}$ is $1/\binom{l(\sigma)+d(\sigma)}{d(\sigma)}$.

What happens if $\bar{u}$ is a pseudo-random sequence of additions generated by the Algorithm?

Lemma 2: Let $\bar{u}$ be a pseudo-random sequence of additions generated by the Algorithm with t seeds. If $t \geq 3d(\sigma) + 2$, the probability that σ is created during $\bar{u}$ is $O(1/\binom{l(\sigma)+d(\sigma)}{d(\sigma)})$. The constant factor depends on $d(\sigma)$.

Remark: Lemma 2 holds even when the random variables X_i's are not generated by PCG, as long as they satisfy the Proposition 1.

The total *structural change* during $\bar{u}$ is defined to be the number of configurations in Π that are created during $\bar{u}$. More precisely, for each configuration $\sigma \in \Pi$, defining a 0-1 indicator function as follows:

$$i(\sigma, \bar{u}) = \begin{cases} 1, & \text{if } \sigma \text{ is created during } \bar{u}, \\ 0, & \text{otherwise.} \end{cases}$$

Then the total structural change during $\bar{u}$ is

$$\sum_{\sigma \in \Pi} i(\sigma, \bar{u})$$

This also bounds the number of configurations that are destroyed during $\bar{u}$. In general, given $\alpha \geq 0$, the αth order conflict change during $\bar{u}$ is

$$\sum_{\sigma \in \Pi} [d(\sigma) + l(\sigma)]^{\alpha} i(\sigma, \bar{u})$$

This also bounds the sum of the αth powers of the conflict sizes of the destroyed configurations.

The $0th$ order conflict change is simply the structural change as defined before. The first order conflict change is the ordinary conflict change.

Let $d = d_{max}(\Pi) = max\{d(\sigma)|\sigma \in \Pi\}$(d is bounded). Let $\bar{u}$ be either a truly random sequence or a pseudo-random sequence of addition that is generated by algorithm using at least (3d+2) seeds. Then the expected αth order conflict change during $\bar{u}$ is bounded, within a constant factor, by

$$\sum_{\sigma\in\Pi} \frac{[d(\sigma)+l(\sigma)]^{\alpha}}{\binom{l(\sigma)+d(\sigma)}{d(\sigma)}} \approx \sum_{\sigma\in\Pi} \frac{1}{[l(\sigma)+d(\sigma)]^{d(\sigma)-\alpha}}$$

Let $d_{min}(\Pi) = min\{d(\sigma)|\sigma \in \Pi\}$, for $\forall\alpha$, $0 \le \alpha \le d_{min}(\Pi)$, define

$$\Theta(\Pi, \alpha) = \sum_{\sigma\in\Pi} \frac{1}{[l(\sigma)+d(\sigma)]^{d(\sigma)-\alpha}}$$

Summanized: Let $\Pi = \Pi(N)$ be any configuration space, $d = d_{max}(\Pi)$ be bounded by a constant. Let $\bar{u}$ be a truly random sequence of additions or a pseudo-random sequence of additions for which Lemma 2 holds(it holds if we use algorithm by using at least (3d+2) seeds). Then for each fixed $\alpha \ge 0$, the expected value of the αth order conflict change over Π is $O(\Theta(\Pi, \alpha))$(If $\bar{u}$ is truly random, this expected value is also $\Omega(\Theta(\Pi, \alpha))$)..

Theorem 1: For any such configuration space Π and any fixed $\alpha \ge 0$, the expected value of the αth order conflict change over a pseudo-random sequence generated as above is bounded, within a constant factor, by the expected value over a truly random sequence.

4 An Example: Trapezodial Decompositions

Let N be a set of segments in the plane and $H(N)$ be the trapezodial decomposition of N. We call a trapezoid f in the plane *feasible* with respect to N, if f occurs in some trapezodial decomposition $H(M)$, where $M \subseteq N$. Let $D(f)$, the triggers associated with f, be the set of segments in N that "define" f, i.e., the segments that are adjacent to f. The stopper set $L(f)$ is the set of remaining segments in N that intersect f. Defining a *racquet* in a trapezoidal decomposition $H(M)$ as a pair $\sigma = (f, e)$, where f is a trapezoid in $H(M)$ and e is a vertical attachment that is adjacent to f. We say f is the face of the racquet, e is called the handle.

Why we need to define racquet? It is very useful when we prove the expected running time remains $O(k + n \log n)$ if using pseudo-randomized algorithm.

A feasible racquet σ over N is a racquet that occurs in a trapezodial decomposition $H(M)$, for some $M \subseteq N$. The trigger set $D(\sigma)$ is the set of segments in N that define σ, i.e., the segments in N that adjacent to the face or the handle of σ, the stopper set $L(\sigma)$ is the set of remaining segments in N that intersect this racquet. The degree of any racquet is clearly bounded(by 6). Thus, given a set of segments N, we get a configuration space $\Sigma(N)$ of feasible trapezoids or racquets over N.

Trivial Configuration Space: A configuration space $\Pi(N)$ is *trivial*, if for every $M \subseteq N$, the number of configuration in $\Pi(N)$ active over M is $O(1)$.

Let $\Sigma(N)$ be the configuration space of trapezoidal decomposition, for a fixed point $p \in R^2$, let $\Sigma_p(N)$ be the subspace of all feasible trapezoids in Σ that contain p. It is clear that Σ_p is trivial. This is becasue for any $M \subseteq N$, exactly one trapezoid in $H(M)$ can contain p.

The expected running time of the randomized incremental algorithm for constructing the trapezoidal decomposition is $O(k + n \log n)$. We will show that its expected running time remains $O(k + n \log n)$, if we use pseudo-randomized incremental algorithm. Here k is the number of intersetions among the segments in N(we assume the number of seeds is larger enough. Since $d_{max}(\Pi)$ is 6, $t \geq 3 \times 6 + 2 = 20$ is enough).

First, we briefly recall the randomized incremental algorithm. The algorithm adds the segments in N, one at a time, in a random order. At the ith stage it maintains the partition $H(N^i)$ formed. It also associates with each trapezoid in $H(N^i)$ a conflict list of the endpoints of the unadded segments in $N \backslash N^i$ that lies within that trapezoid. Conversely, with each endpoint of an unadded segment, there is an associated pointer to the trapezoid containing it. The addition of a segment S in $N \backslash N^i$ can be accomplished in time proportional to

$$\sum_f \text{facelength}(f) + \text{conflictsize}(f)$$

where f ranges over all trapezoids in $H(N^i)$ intersecting S. Here facelength(f) denotes the total number of vertices of $H(N^i)$ lying on the border of f and conflictsize(f) denotes the size of the conflict list associated with f.

Analysis: Ignoring for a while the cost of updating the conflict information. Consider the cost of adding a segment S to N^i. *Notice that the face-length of f is essentially the number of racquets in $H(N^i)$ with face f.* All these racquets are destroyed at the end of the addition of S. Thus the cost of adding S is proportional to the number of racquets in $H(N^i)$ that are destroyed during the addition of S. Hence the total cost of the algorithm, ignoring conflicts, is of the same order as the total number of racquets that are destroyed in the algorithm. This is the same order as the total number of racquets created in the whole algorithm(a created racquet can be destroted only once). Let Σ be the configuration space of feasible racquets, appling Theorem 1 to Σ, it follows that the expected number of racquets that are created or destroyed during a pseudo-random sequence is bounded, within a constant factor, by the expected value over a truly random sequence.

We turn to the cost of updating conflicts. We can amortize this cost during the addition of S by charging $O(1)$ cost to each point that occurs in the conflict list of a destroyed trapezoid. Thus, for a fixed endpoint v of a segment in N, the total cost charged to v in the whole algorithm is of the same order as the total number of trapezoids containing v that got destroyed in the course of the algorithm. Let Σ_v be the trivial configuration space of feasible trapezoids over N containing v. By the above observation, the total cost charged to v is proportional to the total structural change over Σ_v.

Thus the total running time of the algorithm is of the same order as the total structural change of Σ plus the sum of the total structural changes of Σ_v over all choices of the endpoints v. By Theorem 1, the expected values of these total structural changes in the pseudorandom setting is bounded within a constant factor by the expected values in the random setting. Since the total running time in the truly random setting is $O(k+n\log n)$, this prove that the expected running time of the pseudorandomized incremental algorithm for constructing trapezoidal decompositions remains $O(k + n\log n)$.

5 Derandomization:

We give some basic idea of the derandomization technique that can be used to convert several algorithms based on top-down random sampling into deterministic ones. The general search problem is follows:

Given a set N of objects in R^d, construct the induced partition $H(N)$ and a geometric search structure $\bar{H}(N)$ that can be used to answer the queries over $H(N)$ quickly. We always assume $H(N)$ has bounded-degree property.

A search structure based on top-down sampling:

(1) Choose a random subset $R \subseteq N$ of a large enough constant size r.

(2) Build $H(R)$ and a search structure for $H(R)$.

(3) Build conflict lists of all faces of $H(R)$ of relevent dimension.

(4) For each such face $\Delta \in H(R)$, recursively build a search structure for $N(\Delta)$, the set of objects in N in conflict with Δ.

(5) Build an ascent structure, denote by ascent(N, R)(merge step).

Definition: Let $\Pi(N)$ be a configuration space, for $r \le n$, let $\bar{\Pi}(r)$ denote the maximum reduced size of $\Pi(R)$ for any subset $R \subseteq N$ of size r, the function $\bar{\Pi}(r)$ is called the reduced size function of $\Pi(N)$. We say $\Pi(N)$ has a bounded dimension, if there is a constant d such that $\bar{\Pi}(r) = O(r^d)$ for all $r \le n$. In this case, we say that d is the *dimension* of $\Pi(N)$.

There are important classes of configuration spaces that do not have bounded valence, but whose dimension is bounded(the crucial step in the proof of theorem 5.1.2 of the textbook is to show that a configuration space of bounded valence has bounded dimension).

Theorem 2: Let $\Pi(N)$ be any configuration space of bounded dimension d, let $n = |N|$. let R be a random sample of N formed by $r \le n$ radom independent draws with replacement. then with probability $\ge 1/2$, for each active configuration $\sigma \in \Pi^0(R)$, the conflict size of σ(relative to N)$\le c(n/r)\log r$. More generally, fix any $c > d$, for any $t \ge r$, with probability $1 - O(1/(t^{c-d}))$, the conflict size (relative to N) of each configuration in $\Pi^0(R)$ is at most $2c(n/r)\log t$. Here c is a large enough constant.

Range Space: a configuration space $\Pi(N)$ is a *range space*, if the trigger set of every configuration in $\Pi(N)$ is empty.

ϵ-net: Given $\epsilon > 0$, a subset $R \subseteq N$ is called an ϵ-net of the range space $\Pi(N)$, if the conflict size(relative to N) of every range in $\Pi^0(R)$ is at most ϵn.

If $\Pi(N)$ is a range space, we adopt the convention of identifying every range $\sigma \in \Pi(N)$ with its stopper set, i.e., the set of objects in N in conflict with σ(because the trigger set of every range is empty, thus every range can be thought of as just a subset of N).

Theorem 2 implies that any range space $\Pi(N)$ of bounded dimension has a $(1/r)$-net of $O(r \log r)$ size. Moreover, such a net can be found in a randomized fashion trivially: for a large enough constant a, a random sample $R \subseteq N$ of $ar \log r$ size is a $(1/r)$-net with high probability. We give a deterministic algorithm for the same problem as follows:

Theorem 3: Suppose $\Pi(N)$ is a range space of bounded dimension $d \geq 1$, let n be the size of N. Fix a parameter $r > 1$. Then, a $(1/r)$-net for $\Pi(N)$ of $O(r \log r)$ size can be found determinstically in $O(n[r^2 \log r]^d)$ time, assuming that $\Pi(N)$ satisfies a certain enumerability assumption.

The enumerability assumption is that , for any subset $S \subseteq N$, we can enumerate all ranges in $\Pi(S)$ in $O(s\pi(s))$ time, here $s = |S|$, $\pi(s) = |\Pi(S)|$. The enumerability assumption is easy to satisfy in many geometric applications.

References

[1] N. Alon, O. Goldreich, J. Hastad and R. Peralta. Simple contructions of almost k-wise independent random variables. *31st FOCS*, 544-553, 1990.

[2] H. Karloff, P. Raghavan, Randomized algorithms and pseudorandom numbers. *Proceeding of hte 20th annual ACM symp. on Theory of computing*, 310-321, 1988.

[3] J. Matousek, Approximations and optimal geometric divide-and-conquer. *23rd STOC*, 505-511, 1991.

[4] K. Mulmuley, Randomized geometric algorithms and pseudo-random generators. *33rd FOCS*, 90-100, 1992.

[5] V. N. Vapnik, A. Y. Chervonenkis, On the uniform convergence of relative frequeies of events to their probabilities. *Theory Probability Applications*, 16:264-280, 1971.

[6] B. Chazelle, J. Friedman, A determinstic view of random sampling and its use in geometry. *Combinatorica*, 10(3):229-249, 1990.