

Sampling from Databases

CompSci 590.04

Instructor: Ashwin Machanavajjhala

Recap

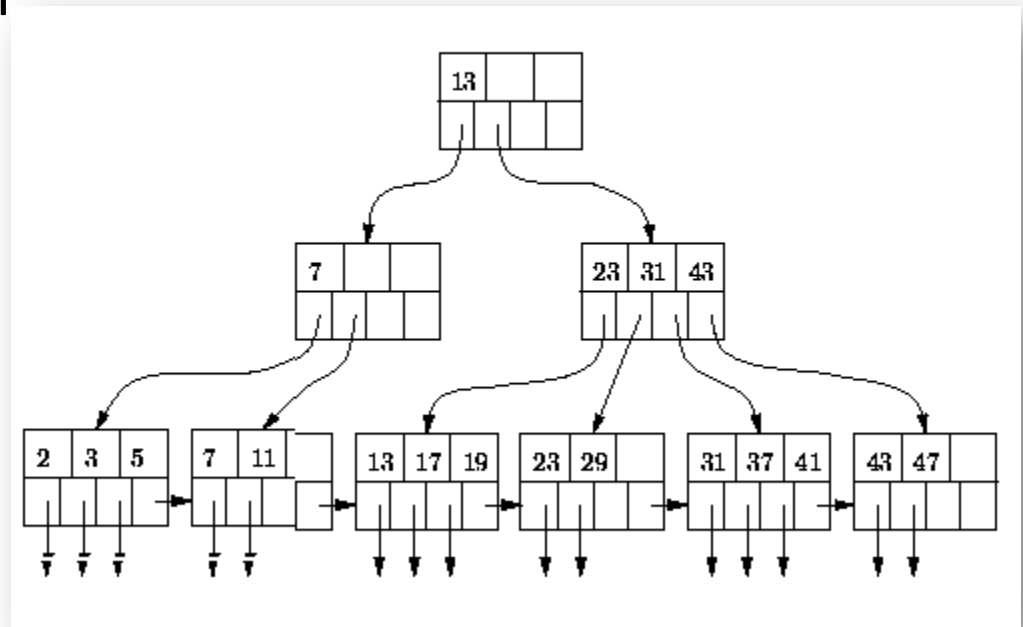
- Given a set of elements, random sampling when number of elements N is known is easy *if you have random access to any arbitrary element*
 - Pick n indexes at random from $1 \dots N$
 - Read the corresponding n elements
- Reservoir Sampling: If N is unknown, or if you are only allowed sequential access to the data
 - Read elements one at a time. Include t^{th} element into a reservoir of size n with probability n/t .
 - Need to access at most $n(1+\ln(N/n))$ elements to get a sample of size n
 - Optimal for any reservoir based algorithm

Today's Class

- In general, sampling from a database where elements are only accessed using indexes.
 - B⁺-Trees
 - Nearest neighbor indexes
- Estimating the number of restaurants in Google Places.

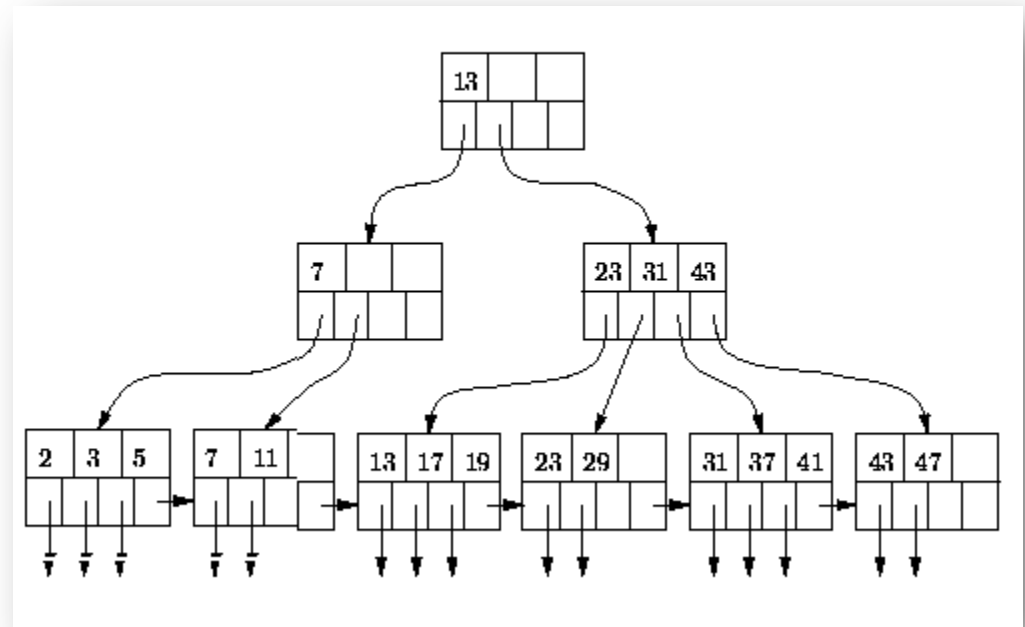
B+ Tree

- Data values only appear in the leaves
- Internal nodes only contain keys
- Each node has between $f_{\max}/2$ and f_{\max} children
 - f_{\max} = maximum fan-out of the tree
- Root has 2 or more children



Problem

- How to pick an element uniformly at random from the B⁺ Tree?

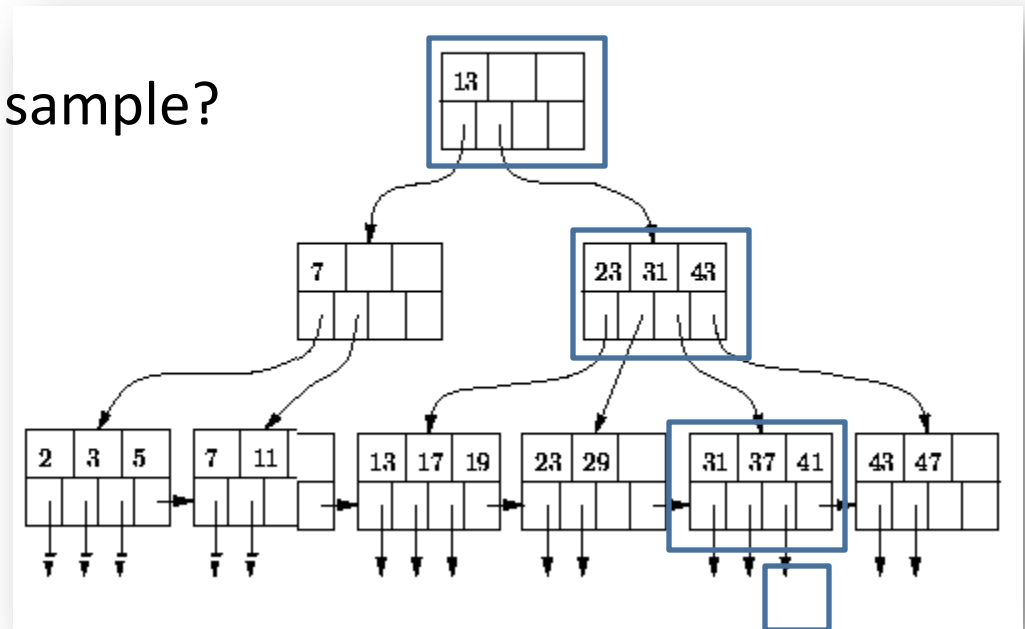


Attempt 1: Random Path

Choose a random path

- Start from the root
- Choose a child uniformly at random
- Uniformly sample from the resulting leaf node

- Will this result in a random sample?



Attempt 1: Random Path

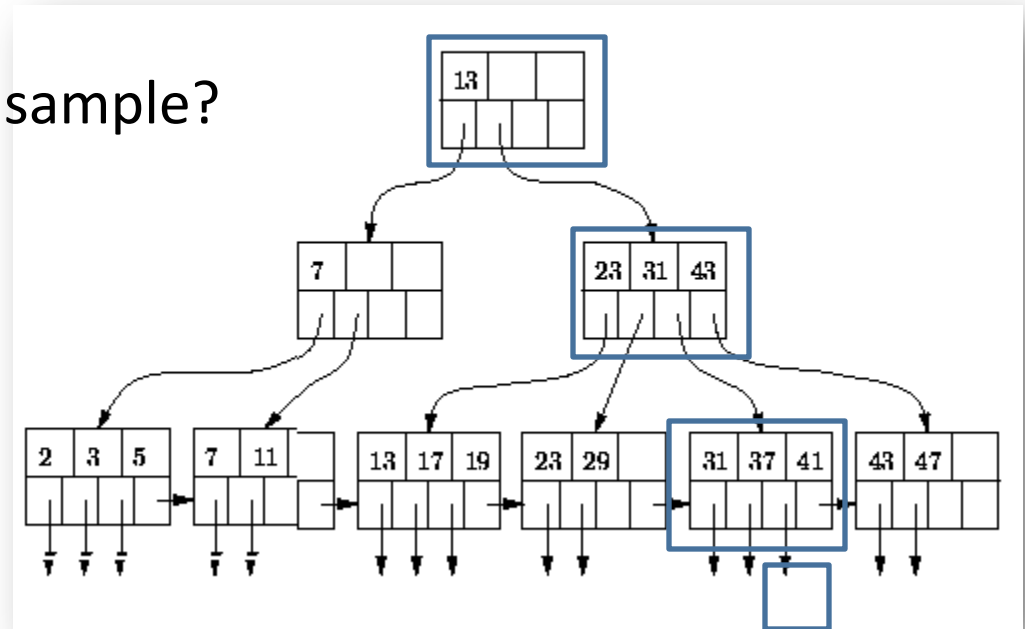
Choose a random path

- Start from the root
- Choose a child uniformly at random
- Uniformly sample from the resulting leaf node

- Will this result in a random sample?

NO.

Elements reachable from internal nodes with low fanout are more likely.



Attempt 2 : Random Path with Rejection

- Attempt 1 will work if all internal nodes have the same fan-out
- Choose a random path
 - Start from the root
 - Choose a child uniformly at random
 - Uniformly sample from the resulting leaf node
- Accept the sample with probability $\prod_{i \in \text{path}} f_i / f_{\max}$

Attempt 2 : Correctness

- Any root to leaf path is picked with probability: $\prod_{i \in \text{path}} f_i / f_{\max}$
- The probability of including a record given the path: $\prod_{i \in \text{path}} 1 / f_i$

Attempt 2 : Correctness

- Any root to leaf path is picked with probability: $\prod_{i \in path} f_i / f_{max}$
- The probability of including a record given the path: $\prod_{i \in path} 1 / f_i$
- The probability of including a record: $\prod_{i \in path} 1 / f_{max} = 1 / f_{max}^h$

Attempt 3 : Early Abort

Idea: Perform acceptance/rejection test at each node.

- Start from the root
- Choose a child uniformly at random
- Continue the traversal with probability: f_i / f_{max}
- At the leaf, pick an element uniformly at random, and accept it with probability : $\frac{\text{\# of elements in leaf}}{\text{max \# elements in leaf}}$

Proof of correctness: same as previous algorithm

Attempt 4: Batch Sampling

- Repeatedly sampling n elements will require accessing the internal nodes many times.

Attempt 4: Batch Sampling

- Repeatedly sampling n elements will require accessing the internal nodes many times.

Perform random walks simultaneously:

- At the root node, assign each of the n samples to one of its children uniformly at random
 - $n \rightarrow (n_1, n_2, \dots, n_k)$
- At each internal node,
 - Divide incoming samples uniformly across children.
- Each leaf node receives s samples. Include each sample with acceptance probability

$$\prod_{i \in \text{path}} f_i / f_{\max}$$

Attempt 4 : Batch Sampling

- Problem: If we start the algorithm with n , we might end up with fewer than n samples (due to rejection)

Attempt 4 : Batch Sampling

- Problem: If we start the algorithm with n , we might end up with fewer than n samples (due to rejection)
- Solution: Start with a larger set
- $n' = n/\beta^{h-1}$, where β is the ratio of average fanout and f_{\max}

Summary of B⁺tree sampling

- Randomly choosing a path weights elements differently
 - Elements in the subtree rooted at nodes with lower fan-out are more likely to be picked than those under higher fan-out internal nodes
- Accept/Reject sampling helps remove this bias.

Nearest Neighbor indexes

Google

Get directions My places

indian restaurant

Azitra Indian Restaurant
 Traditional & Popular Dishes With A Contemporary Flair. Visit Azitra!
www.azitra.us/

Spice & Curry ▼
 2105 North Carolina 54, Durham, NC
 Triangle Village Shopping Center
 (919) 544-7555 · spiceandcurry.net
 Category: Indian Restaurant
 15 39 reviews
 new management · lunch buffet · food quality · chicken tikka masala · south indian
 "I just love this place and their buffet lunch..I always go there for ..."

Tandoor Indian Restaurant ▼
 5410 North Carolina 55 #1, Durham, NC
 (919) 484-2102 · dalesindiancuisine.net
 16 21 reviews \$\$\$
 dinner buffet · goat · brunch · entrees · naan
 "OVERPRICED COLD FOOD!! If you call for delivery they will up their menu ..."

Durham Police Department Crime Mapper

DURHAM CITY OF MEDICINE 1 8 6 9

Durham Police Department
 Durham Sheriff Office
 Durham Interactive Maps

Crime Incidents for the Entire County
 Beginning January 2013 and Ending January 2013

City/County View

ZoomIn ZoomOut Pan

Police Data

- Assault Offenses
- Burglary/Breaking and Entering
- Homicide Offenses
- Larceny/Theft Offenses

Sheriff Data

1. Loaf
 Categories: Bakeries, Breakfast & Brunch
 23 reviews
 111 W Parrish St
 Durham, NC 27701
 (919) 797-1254

resigned to make that effort (if nothing else, it's a good workout for the arms). I'm less enamored of their olive bread and multigrain bread, which were pretty bland in comparison to other bakeries' superior versions

2. Guglhupf Bakery & Patisserie
 Categories: Bakeries, American (New), German
 230 reviews
 2706 Durham-Chapel Hill Blvd
 Durham, NC 27707
 (919) 401-2600

"Bakery/ patisserie/ Durham fave hangout" Time to update my review, as I've been here lots of times since the original. I like this place a lot for breakfast and lunch, or to buy baked goods/ desserts. Pluses

3. Hummingbird Bakery
 Categories: Bakeries, Sandwiches
 12 reviews
 721 Broad St
 Durham, NC 27705

Mo' Map Redo search when map moved

POWERED BY Google
 Map data ©2013 Google - Terms of Use

Problem Statement

Input:

- A database D that can't be accessed directly, and where each element is associated with a geo location.
- A nearest neighbor index (elements in D near $\langle x, y \rangle$)
 - Assumption: index returns k elements closest to the point $\langle x, y \rangle$

Output

- Estimate $\frac{1}{|D|} \sum_{d \in D} f(d)$

Problem Statement

Input:

- A database D that can't be accessed directly, and where each element is associated with a geo location.
- A nearest neighbor index (elements in D near $\langle x, y \rangle$)
 - Assumption: index returns k elements closest to the point $\langle x, y \rangle$

Output

- Estimate $\frac{1}{|D|} \sum_{d \in D} f(d)$

Applications

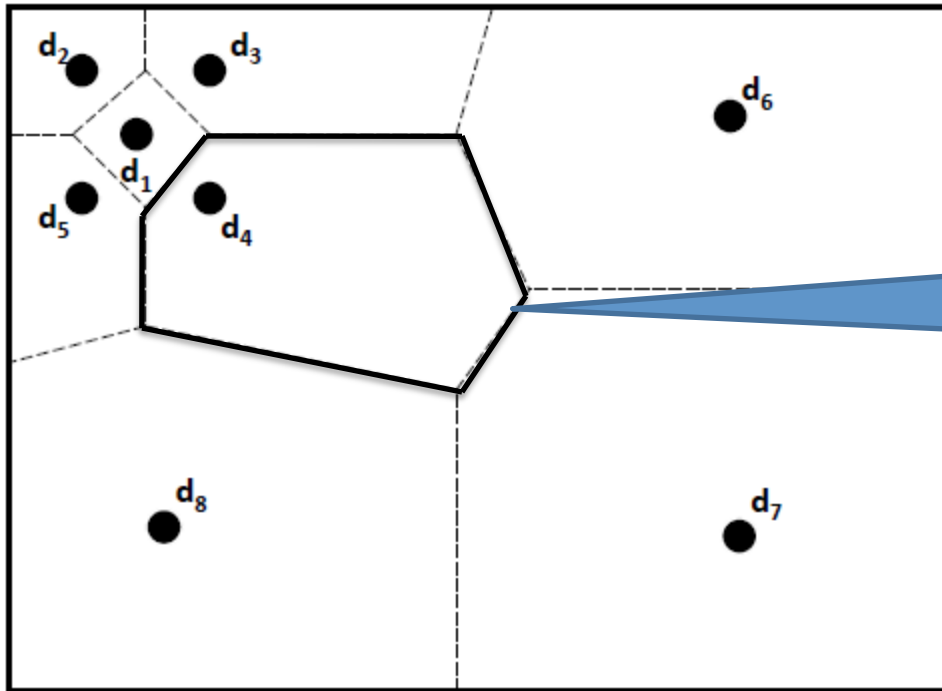
- Estimate the size of a population in a region
- Estimate the size of a competing business' database
- Estimate the prevalence of a disease in a region

Attempt 1: Naïve geo sampling

For $i = 1$ to N

- Pick a random point $p_i = \langle x, y \rangle$
- Find element d_i in D that is closest to p_i
- Return $\hat{f}(D) = \frac{1}{N} \sum_i f(d_i)$

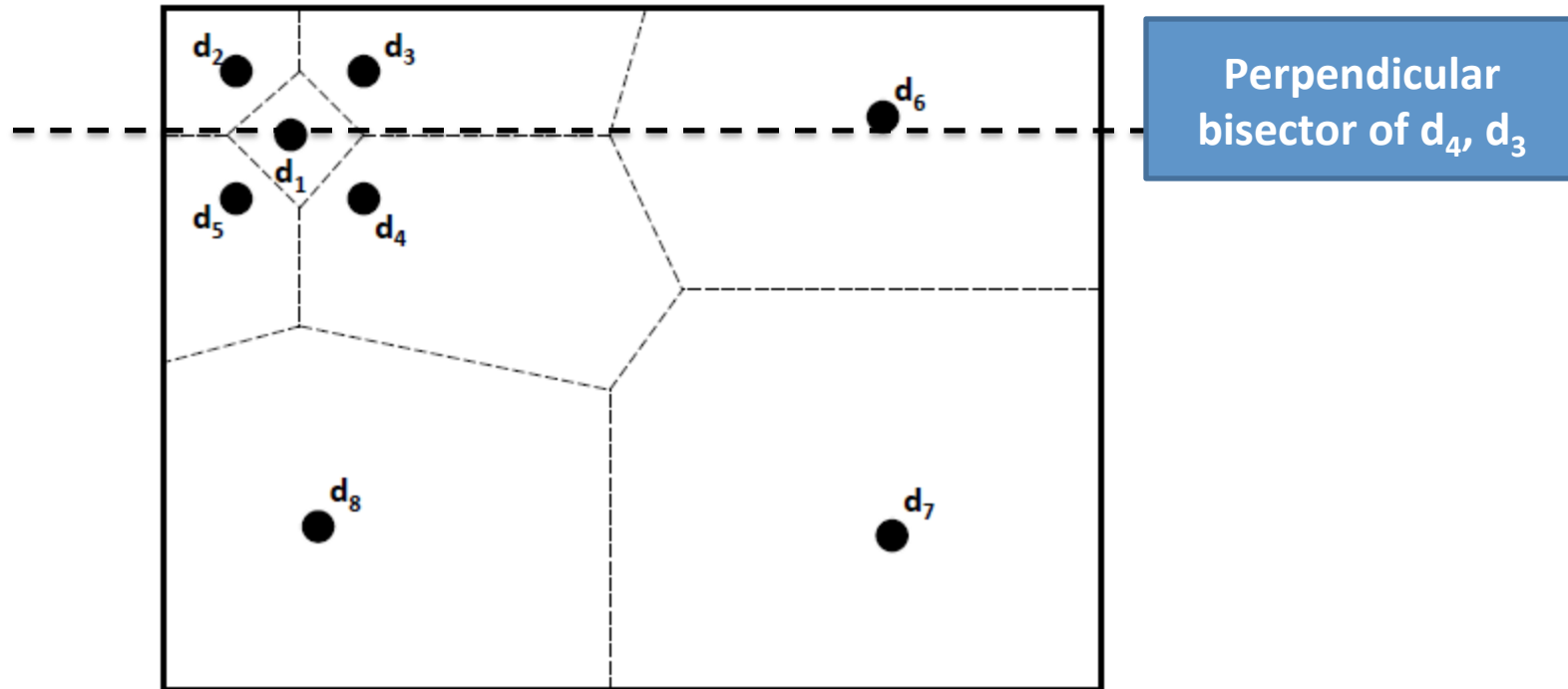
Problem?



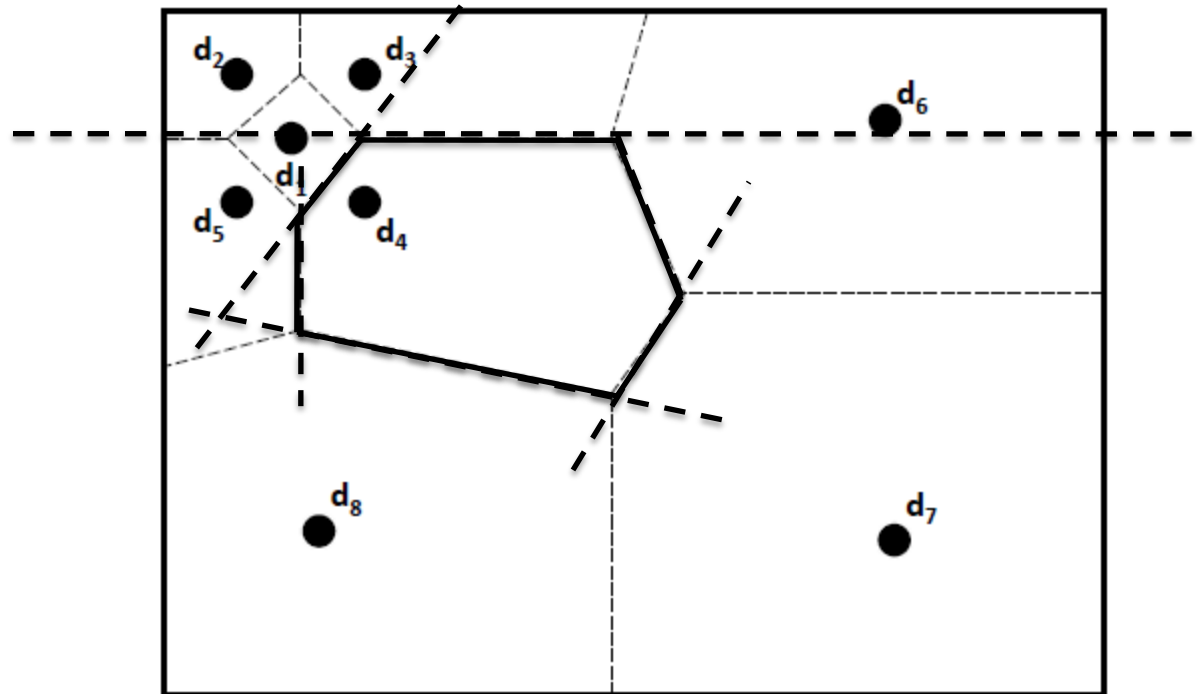
Voronoi Cell:
Points for which d_4 is
the closest element

Elements d_7 and d_8 are much more
likely to be picked than d_1

Voronoi Decomposition

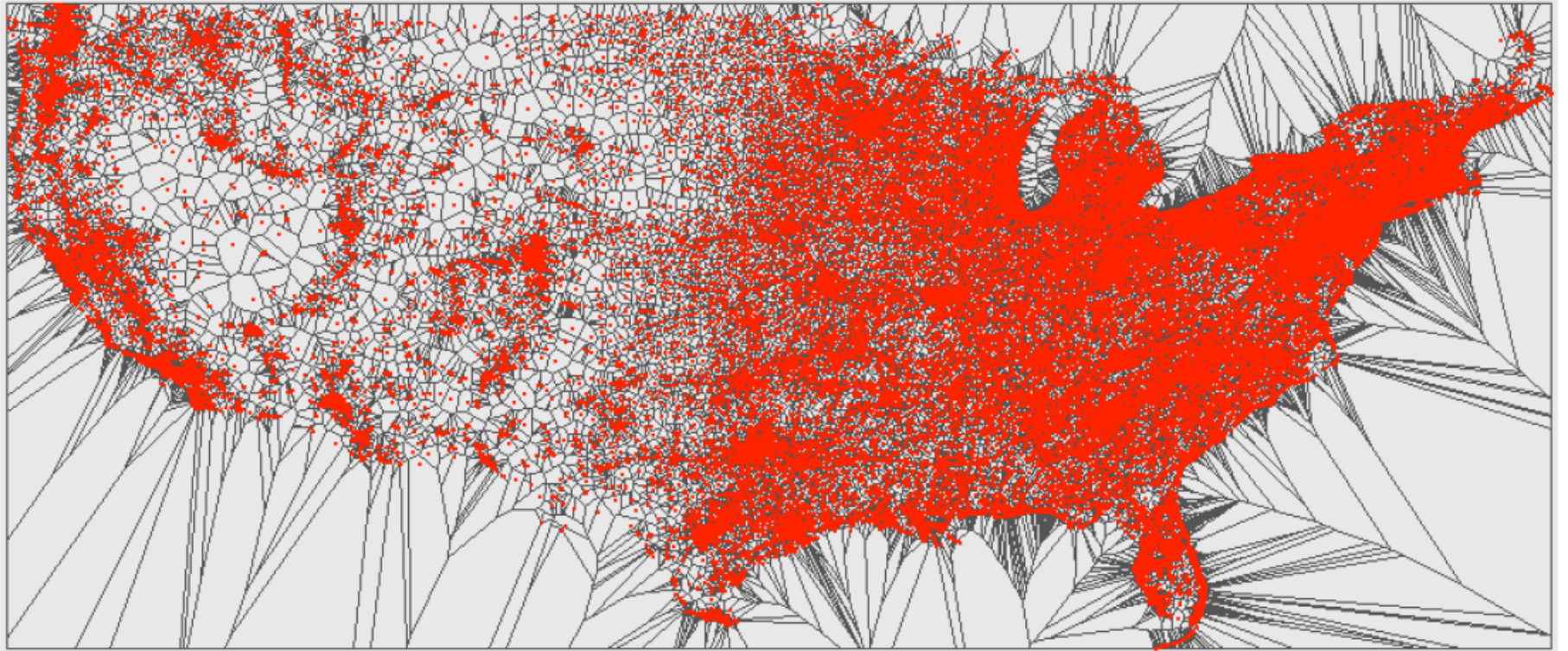


Voronoi Decomposition



$$P[\text{sampling } d_i] = \frac{\text{area}(\text{Vor}(d_i))}{\text{total area}}$$

Voronoi decomposition of Restaurants in US



Attempt 2: Weighted sampling

For $i = 1$ to N

- Pick a random point $p_i = \langle x, y \rangle$
- Find element d_i in D that is closest to p_i

- Return $\hat{f}(D) = \frac{1}{N} \sum_i \left(f(d_i) \cdot \frac{\text{total area}}{\text{area}(\text{Vor}(d_i))} \right)$

Attempt 2: Weighted sampling

For $i = 1$ to N

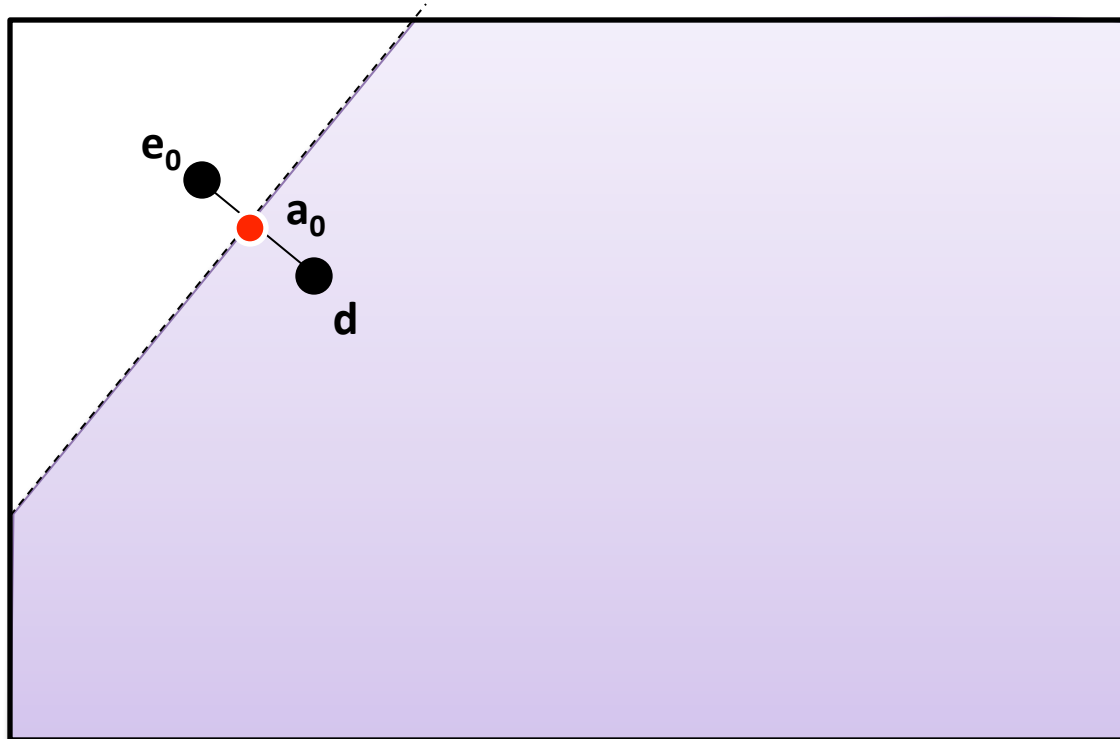
- Pick a random point $p_i = \langle x, y \rangle$
- Find element d_i in D that is closest to p_i
- Return $\hat{f}(D) = \frac{1}{N} \sum_i \left(f(d_i) \cdot \frac{\text{total area}}{\text{area}(\text{Vor}(d_i))} \right)$

Problem:

We need to compute the area of the Voronoi cell.

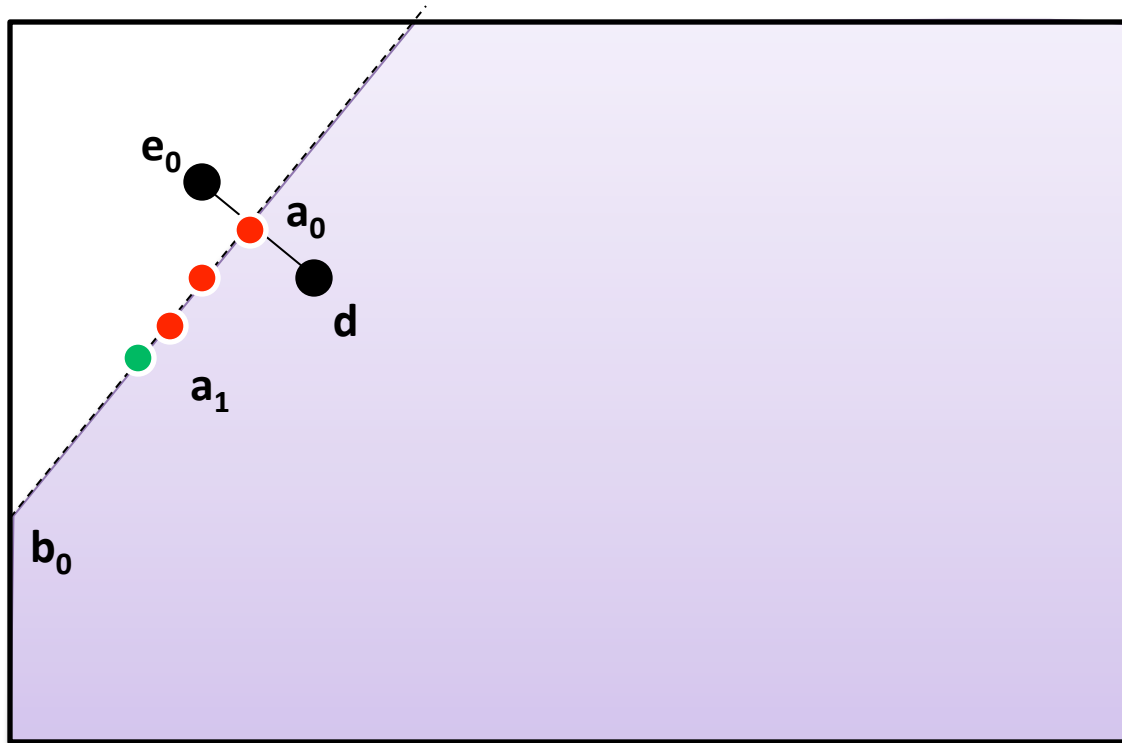
We do not have access to other elements in the database.

Using index to estimate Voronoi cell



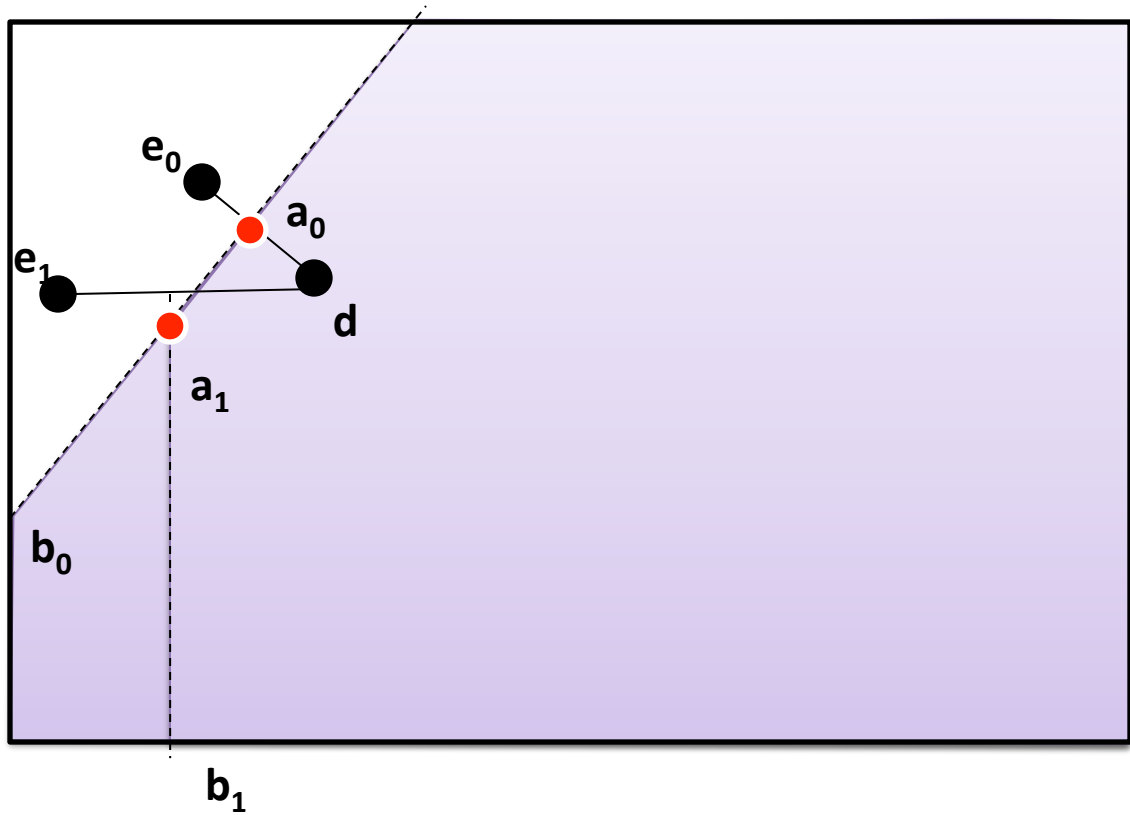
- Find nearest point
- Compute perpendicular bisector
- a_0 is a point *on* the Voronoi cell.

Using index to estimate Voronoi cell



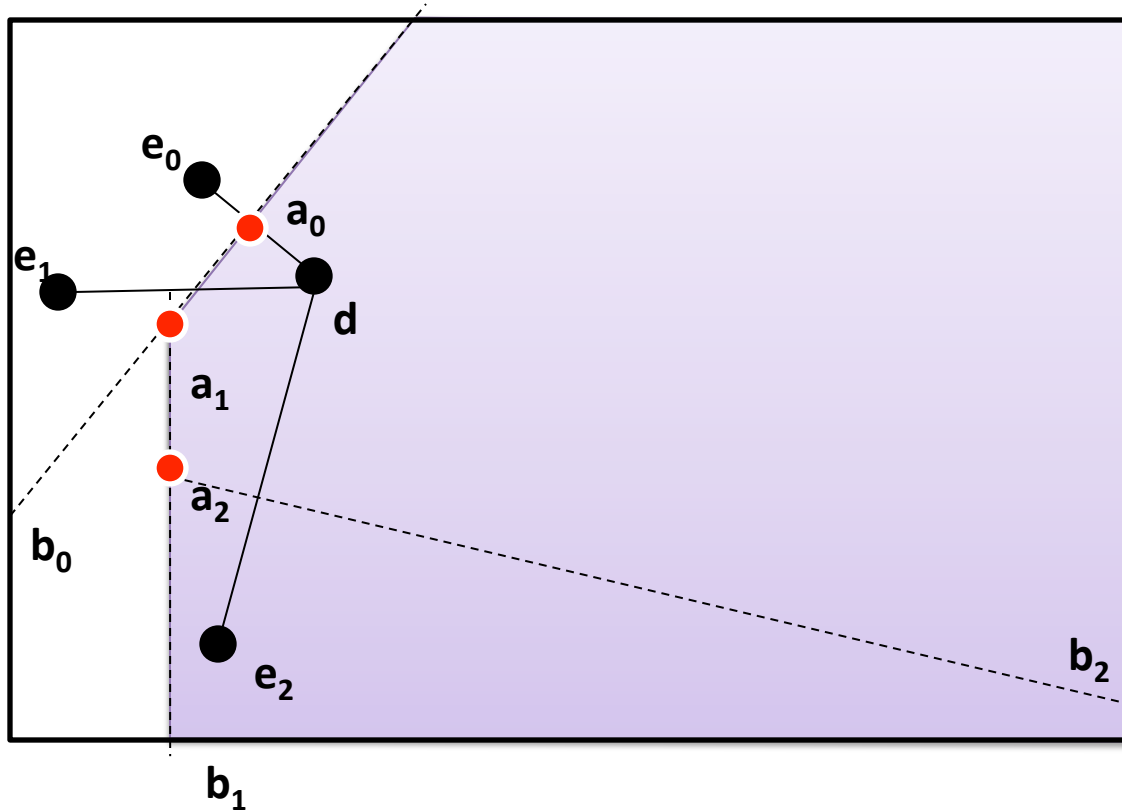
- Find a point on (a_0, b_0) which is just inside the Voronoi cell.
 - Use binary search
 - Recursively check whether mid point is in the Voronoi cell

Using index to estimate Voronoi cell



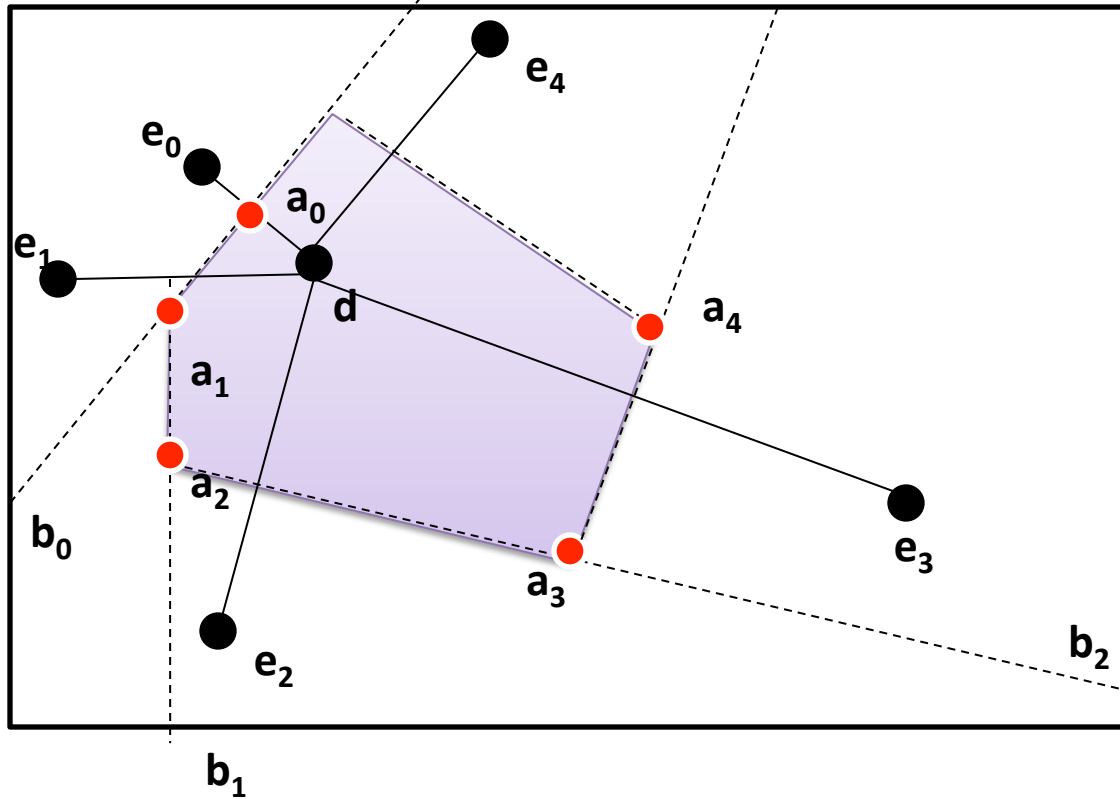
- Find nearest points to a_1
 - a_1 has to be equidistant to one point other than e_0 and d
- Next direction is perpendicular to (e_1, d)

Using index to estimate Voronoi cell



- Find nearest points to a_1
 - a_1 has to be equidistant to one point other than e_0 and d
- Next direction is perpendicular to (e_1, d)
- Find next point ...
- ... and so on ...

Using index to estimate Voronoi cell



- Find nearest points to a_1
 - a_1 has to be equidistant to one point other than e_0 and d
- Next direction is perpendicular to (e_1, d)
- Find next point ...
- ... and so on ...

Number of samples

- Identifying each a_i requires a binary search
 - If L is the max length of (a_i, b_i) ,
then a_{i+1} can be computed with ϵ error in $O(\log(L/\epsilon))$ calls to the index
- Identifying the next direction requires another call to the index
- If number of edges of Voronoi cell = k ,
total number of calls to the index = $O(K \log(L/\epsilon))$
- Average number of edges of a Voronoi cell < 6
 - Assuming general position ...

Summary

- Many web services allow access to databases using nearest neighbor indexes.
- Showed a method to sample uniformly from such databases.
- Next class: Monte Carlo Estimation for #P-hard problems.

References

- F. Olken, “Random Sampling from Databases” , PhD Thesis, U C Berkeley, 1993
- N. Dalvi, R. Kumar, A. Machanavajjhala, V. Rastogi, “Sampling Hidden Objects using Nearest Neighbor Oracles”, KDD 2011