Sampling from Databases

CompSci 590.04
Instructor: AshwinMachanavajjhala

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 … 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

Duke UNIVERSITY

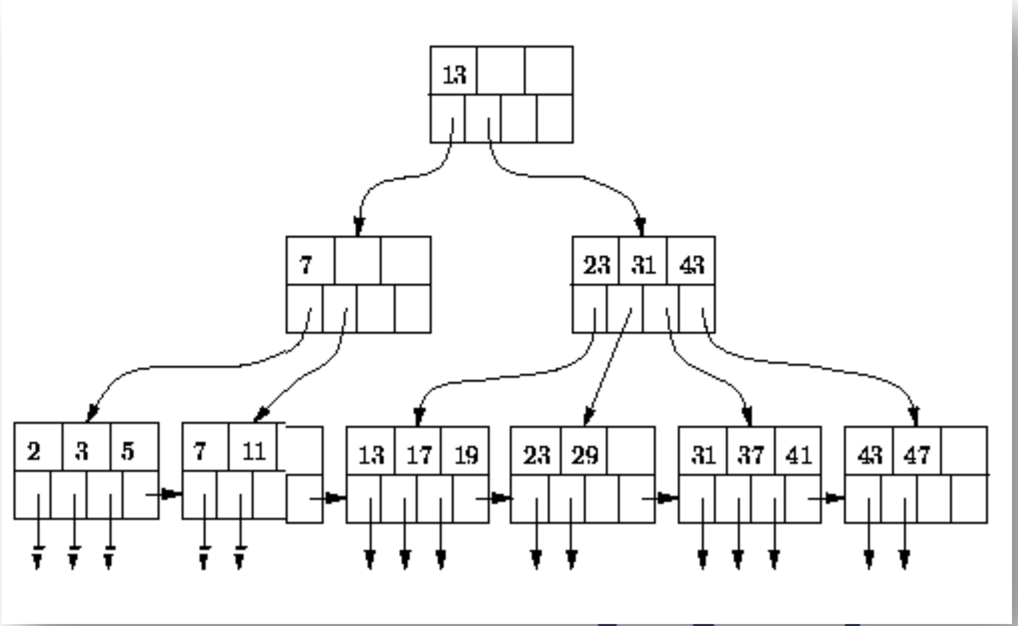
Today's Class

- 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.

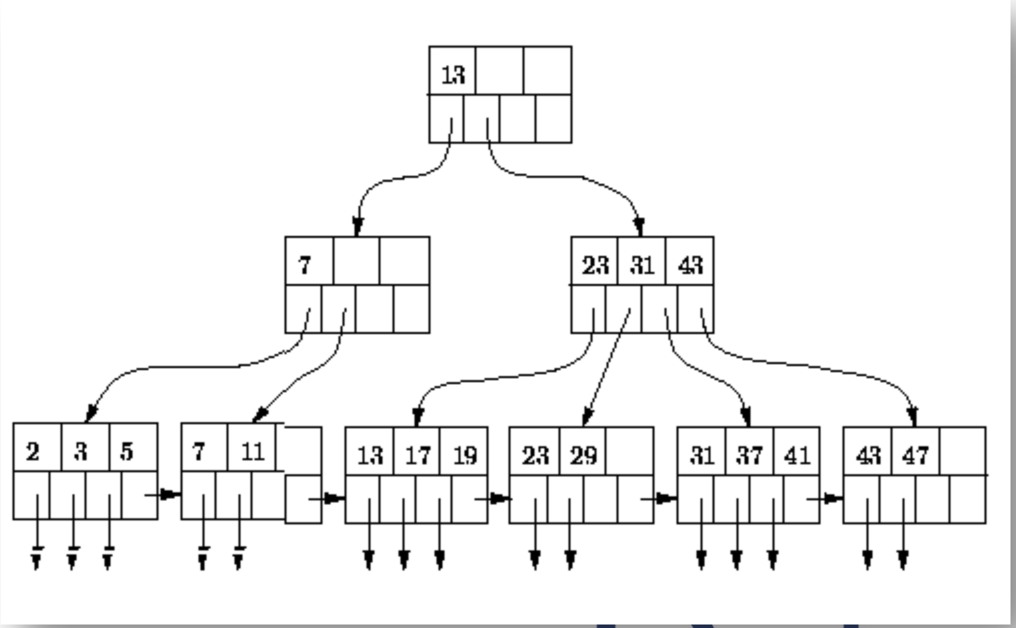
B+ Tree

- 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?

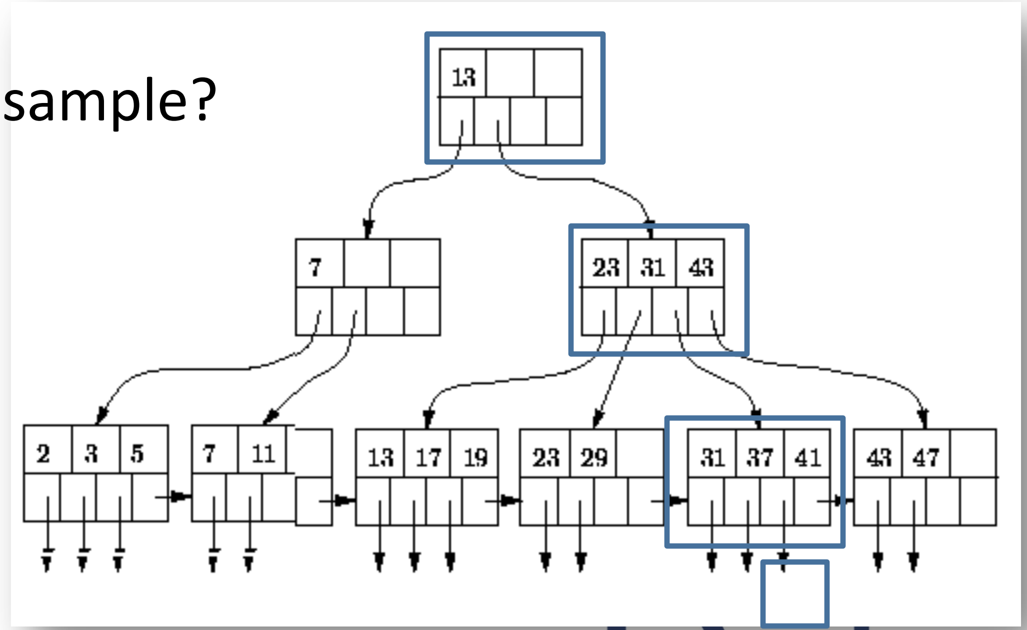


Attempt 1: Random Path

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?



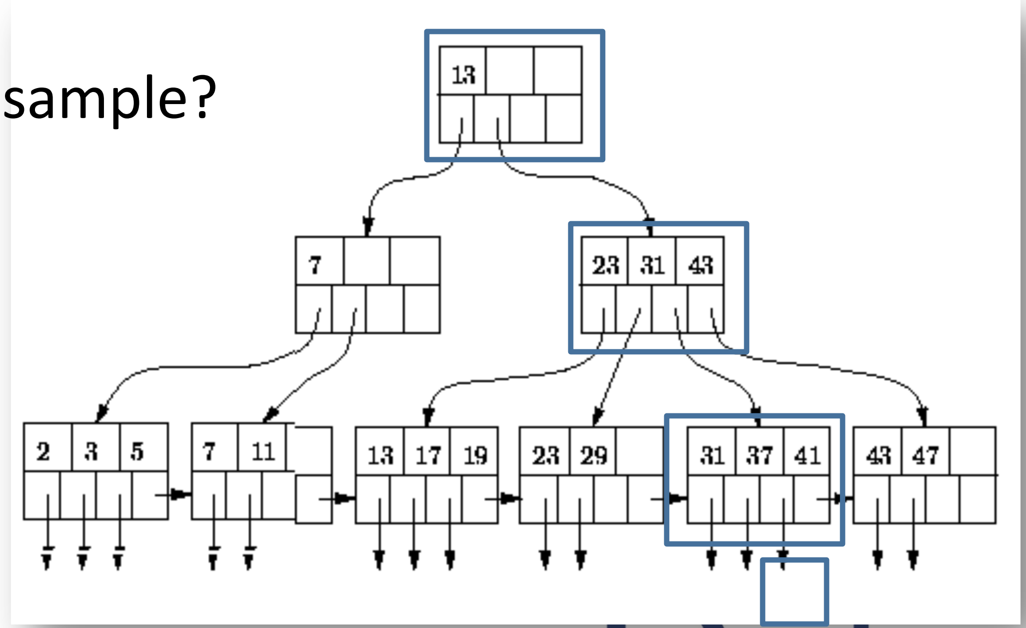
Attempt 1: Random Path

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

Duke UNIVERSITY

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: $\prod_{i \in path} {}^{1}/_{f_i}$

Duke UNIVERSITY

Attempt 2 : Correctness

- Any root to leaf path is picked with probability: $$\prod_{i \in path} {}^{f_i}/_{f_{max}}$$

$$\times$$

- The probability of including a record given the path: $$\prod_{i \in path} {}^{1}/_{f_i}$$

- The probability of including a record: $$\prod_{i \in path} {}^{1}/_{f_{max}} = {}^{1}/_{f_{max}^{h}}$$

Duke UNIVERSITY

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 : $\frac{\#\ of\ elements\ in\ leaf}{max\ \#\ elements\ in\ leaf}$

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
  - $n \rightarrow (n_1, n_2, \ldots, n_k)$
- At each internal node,
  - Divide incoming samples uniformly across children.
- Each leaf node receives *s* samples. Include each sample with acceptance probability

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

Duke UNIVERSITY

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

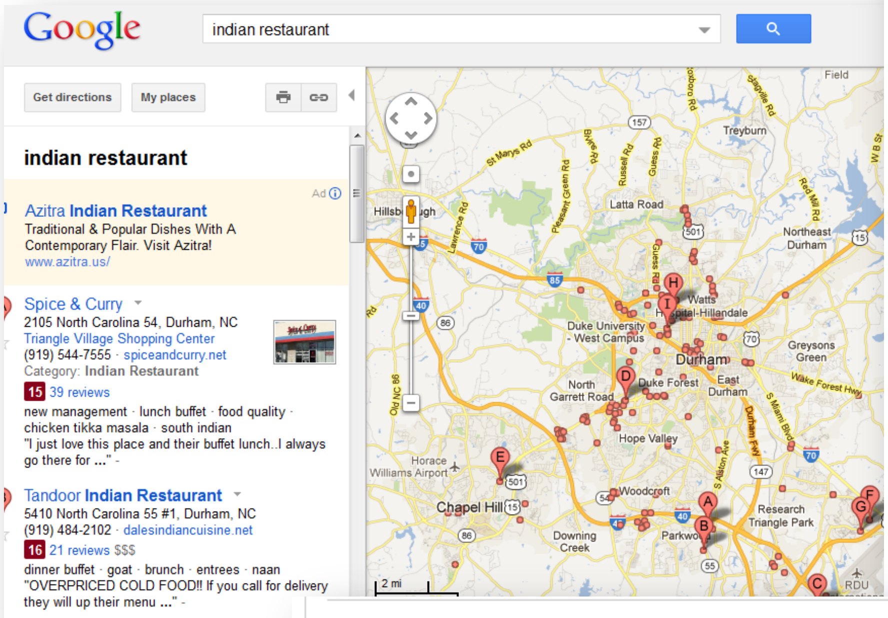
- $n' = n/\beta^{h-1}$, where β is the ratio of average fanout and f_{max}

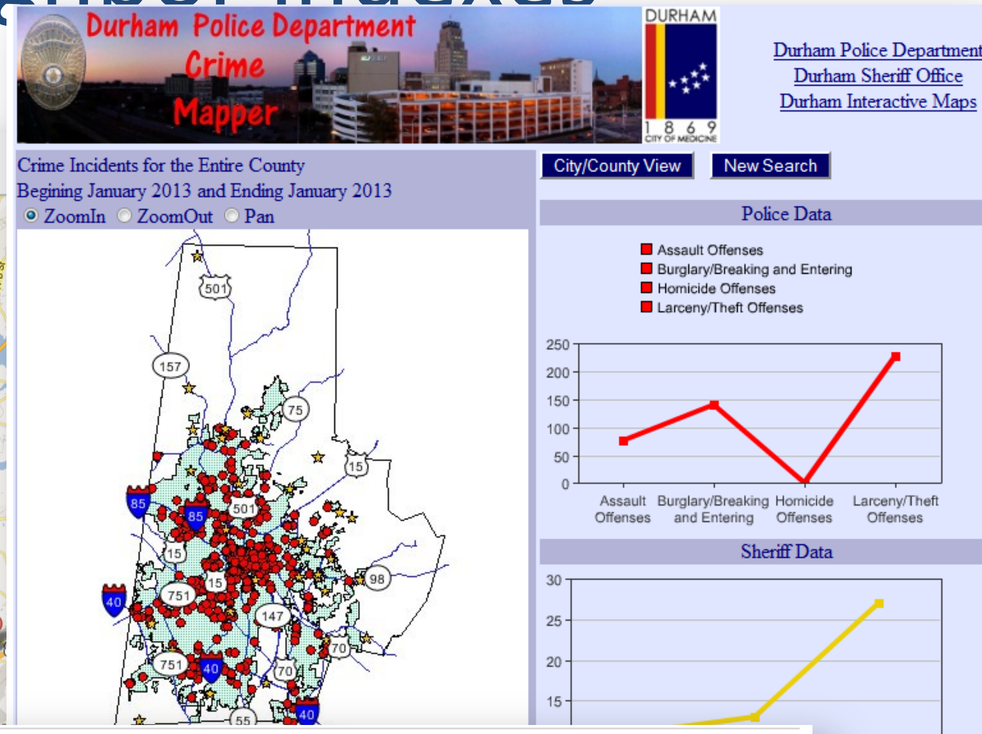
Summary of B^{+}tree sampling

- 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





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 <x, y>)
  - Assumption: index returns k elements closest to the point <x,y>

Output

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

Duke UNIVERSITY

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 <x, y>)
  - Assumption: index returns k elements closest to the point <x,y>

Output

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

Applications

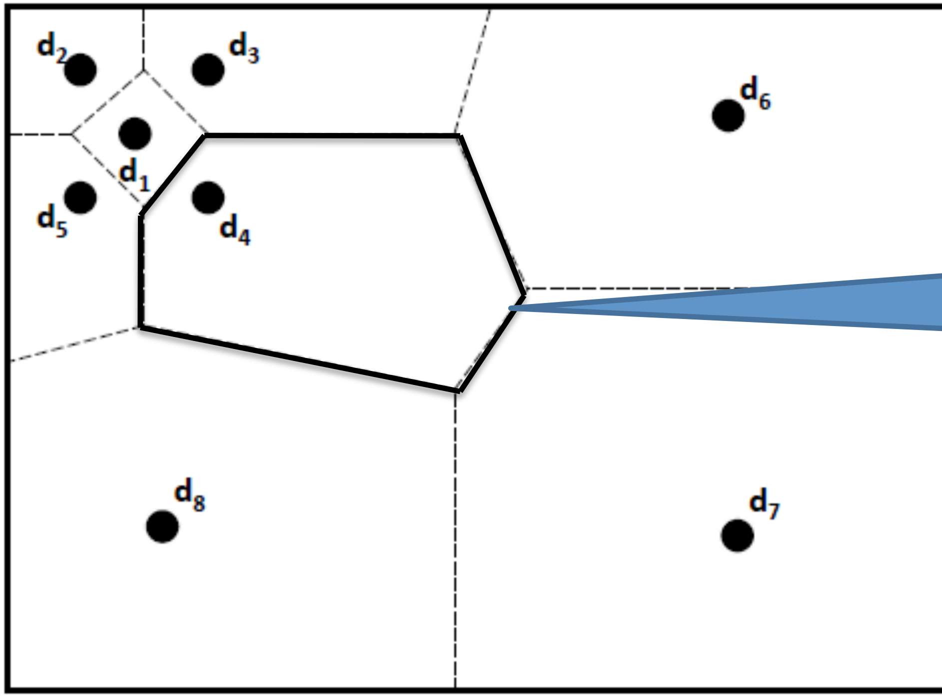
- Estimate the size of a population in a region
- Estimate the size of a competing business' database
- Estimate the prevalence of a disease in a region

Attempt 1: Naïve geo sampling

For i = 1 to N

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

Problem?

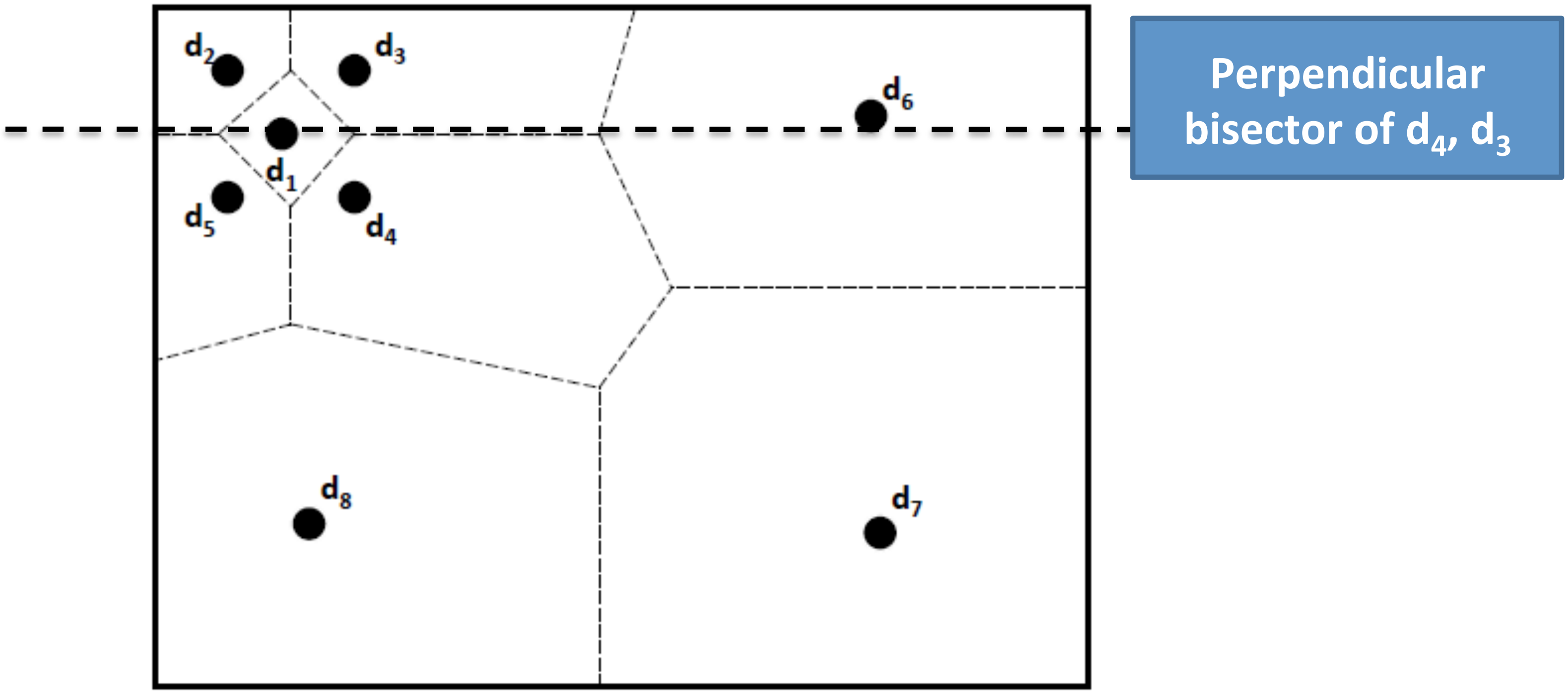


Voronoi Cell: Points for which d_4 is the closest element

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

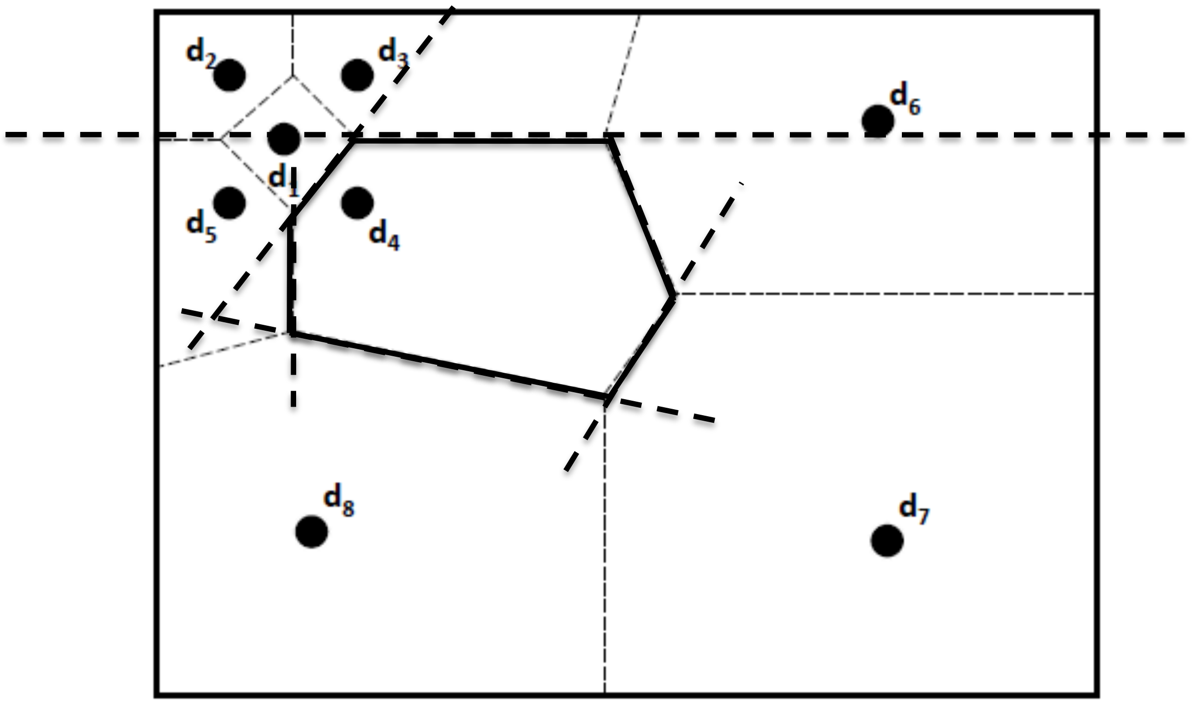
Duke UNIVERSITY

Voronoi Decomposition



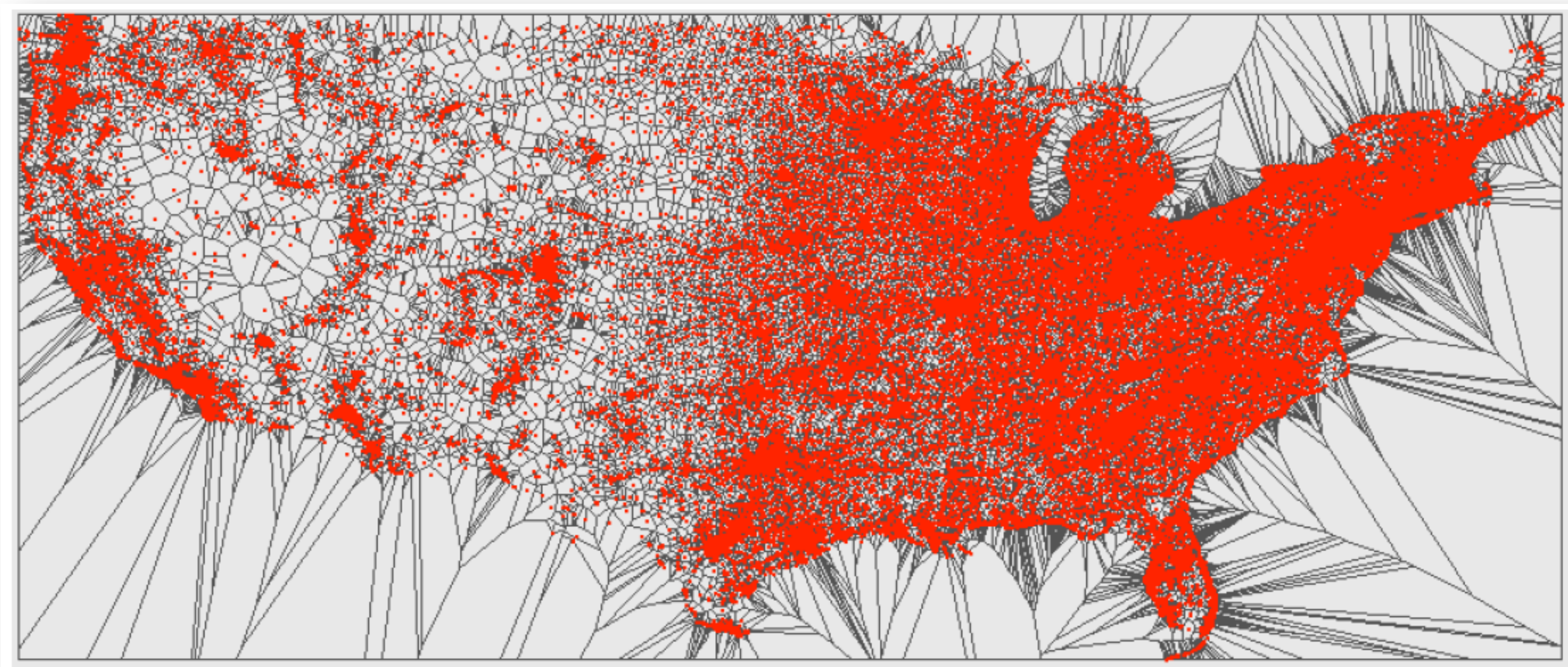
Duke UNIVERSITY

Voronoi Decomposition



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

Voronoi decomposition of Restaurants in US



Attempt 2: Weighted sampling

For i = 1 to N

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

Attempt 2: Weighted sampling

For i = 1 to N

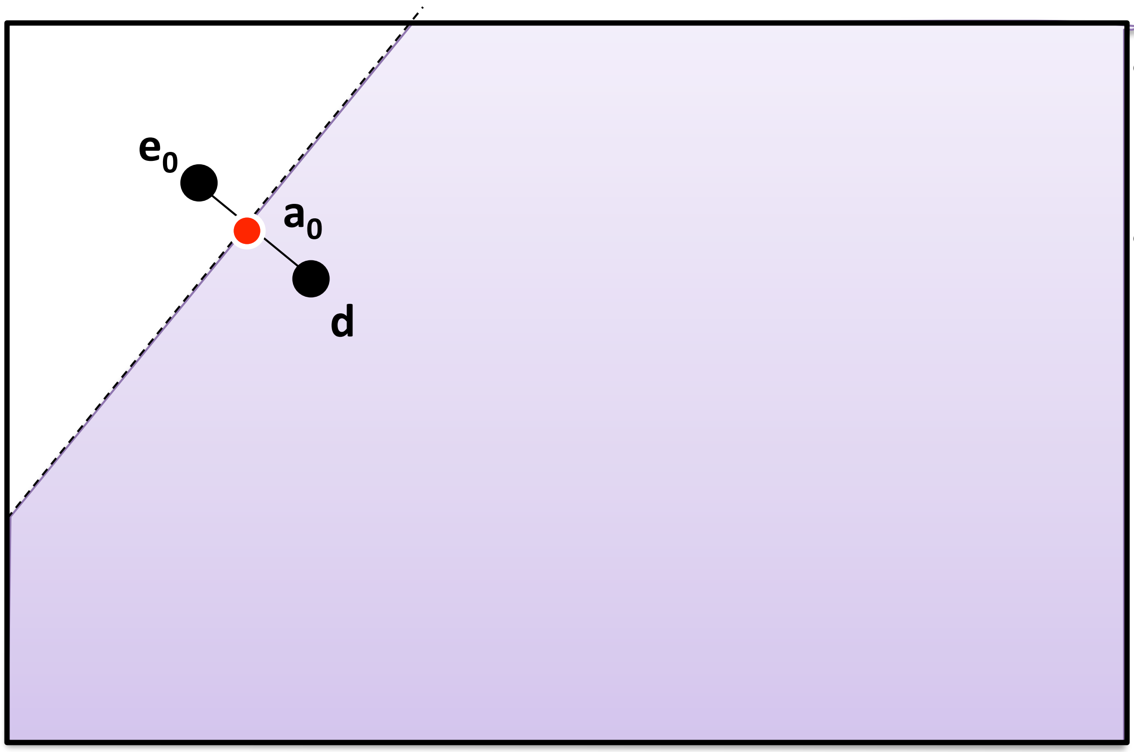
- Pick a random point p_i = <x,y>
- Find element d_i in D that is closes to p_i
- Return $\hat{f}(D) = \frac{1}{N}\sum_i \left(f(d_i) \cdot \frac{total\ area}{area(Vor(d_i))} \right)$

Problem:

We need to compute the area of the Voronoi cell.

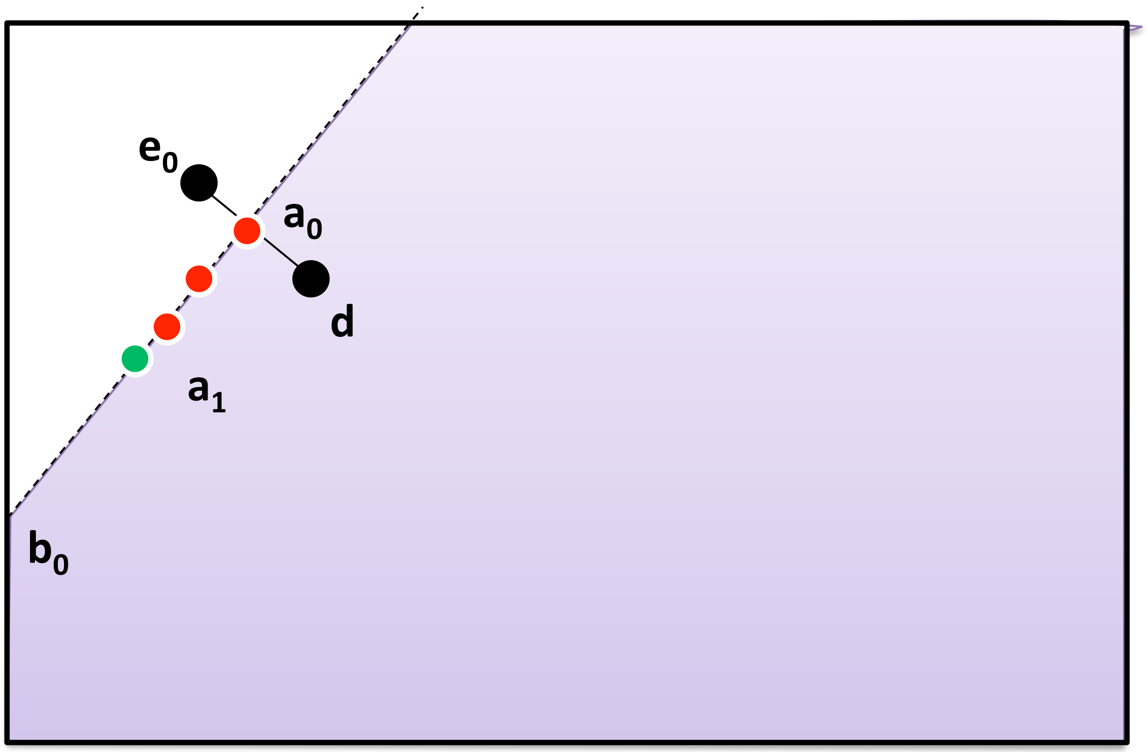
We do not have access to other elements in the database.

Using index to estimate Voronoi cell



- Find nearest point
- Compute perpendicular bisector
- a0 is a point *on* the Voronoi cell.

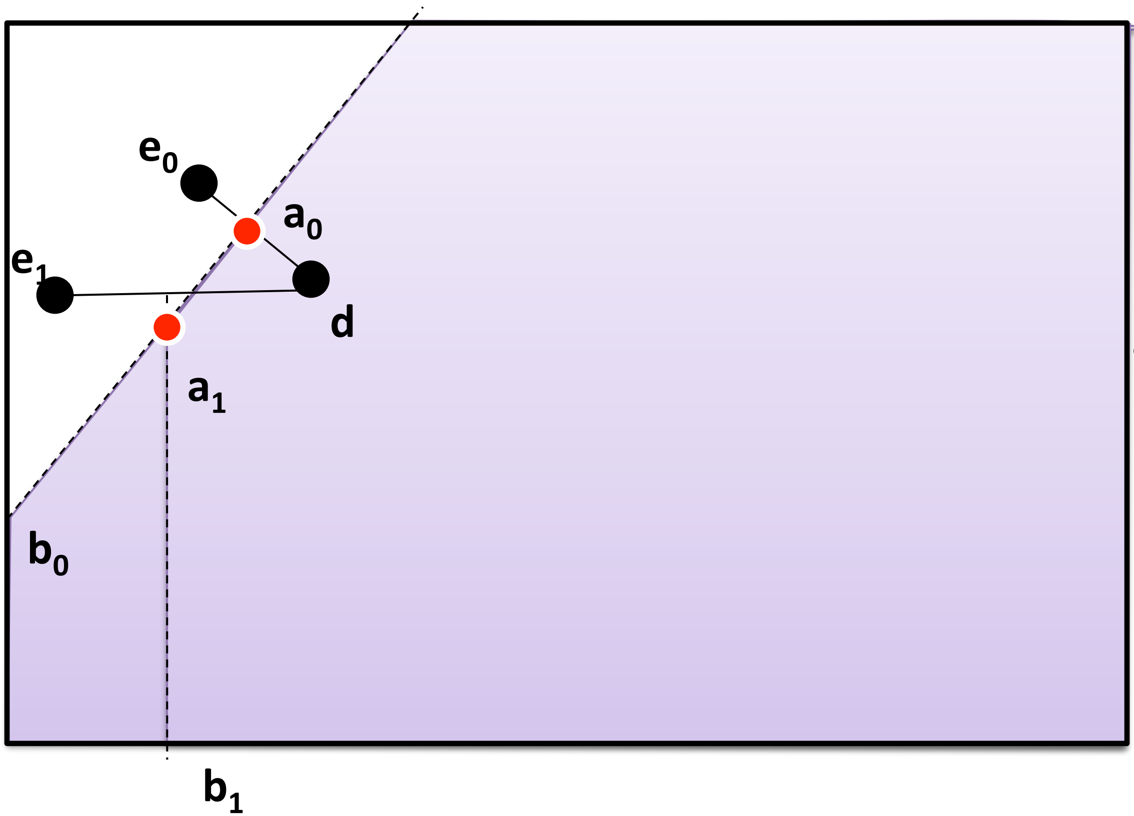
Using index to estimate Voronoi cell



- 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

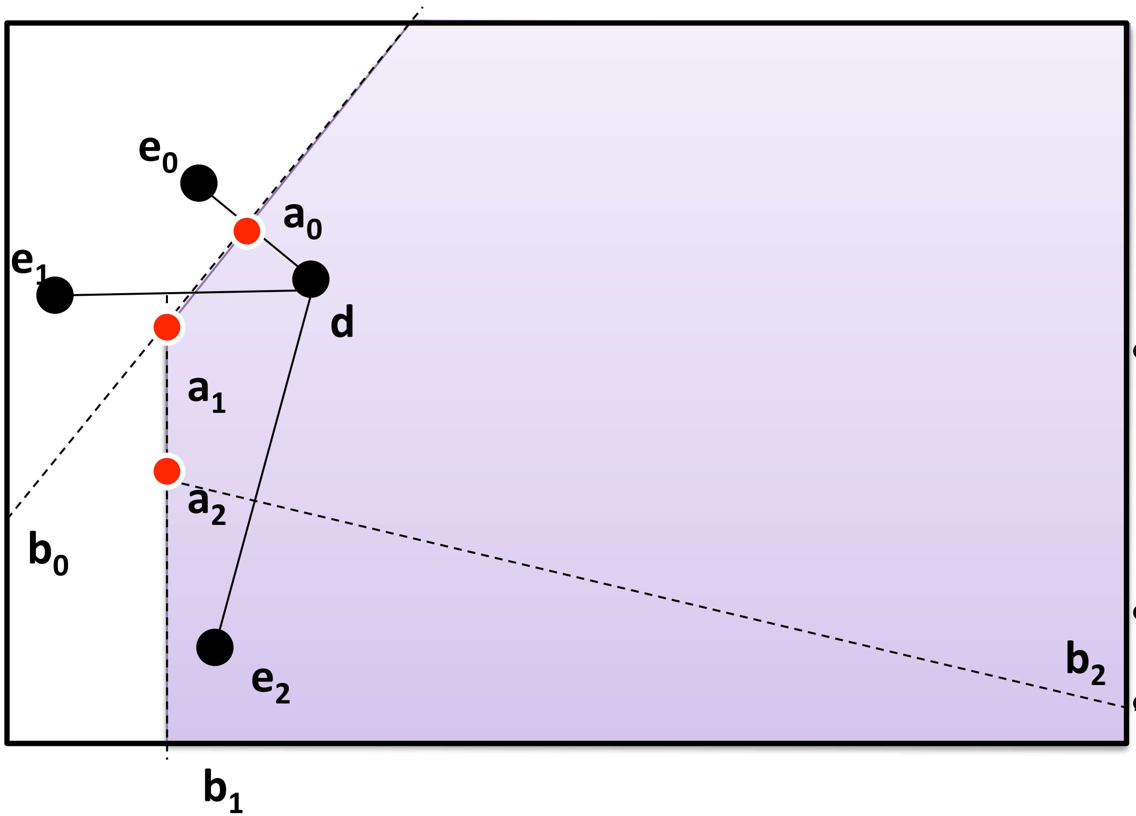
Duke UNIVERSITY

Using index to estimate Voronoi cell



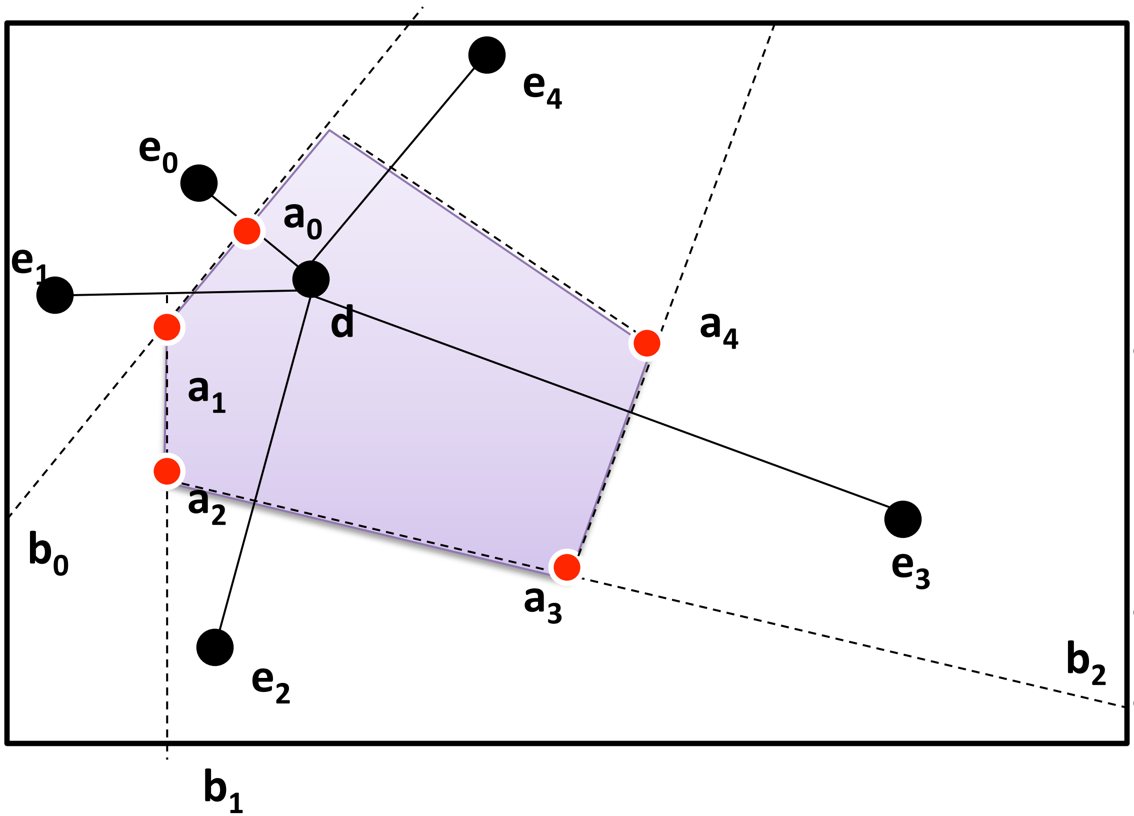
- Find nearest points to a_1
  - a_1 has to be equidistant to one point other than e_0 and d
- Next direction is perpendicular to (e_1,d)

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 (ai, bi),
    then a_{i+1} can be computed with ε error in O(log (L/ε)) 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/ε))
- 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

Duke UNIVERSITY