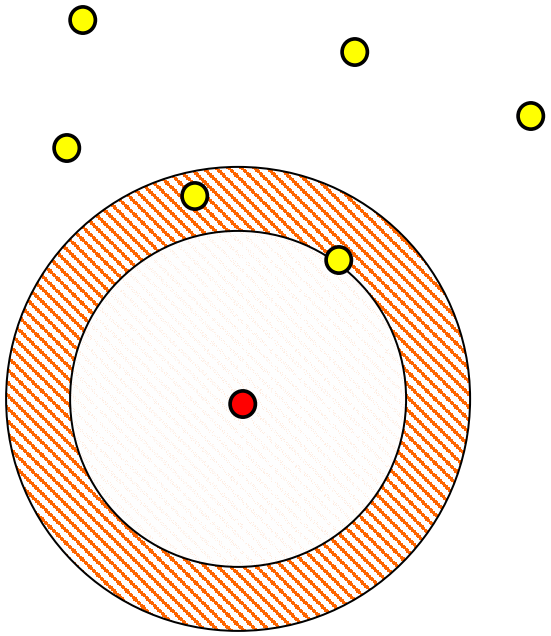
Approximate Nearest Neighbor (Beyond the l_p norms)

Jeff Erickson (UIUC)
Piotr Indyk (MIT)

Nearest Neighbor

- c-Approximate Nearest Neighbor (c-ANN):
  - Given: n points P in a metric $M=(X,D)$, $c=1+\varepsilon$, $\varepsilon > 0$
  - Goal: a data structure that given $q \in X$, reports $p' \in P$ such that

    $$D(q,p') \leq c * \min_{p \in P} D(q,p)$$
- Small space and query time
- Preprocessing and/or update time also an issue



Why Approximate ?

- Exact NN data structures are known. E.g., for $M=(R^2,l_2)$, we can get:
  - Space: $O(n)$
  - Query time: $O(\log n)$
- Unfortunately, they do not scale well. E.g., for $M=(R^d,l_2)$, we get:
  - Space: $O(n^{d/2})$
  - Query time: $O(d^{O(1)} \log n)$
- Problem even more difficult for other metrics:
  - Hausdorff metric
  - Earth Mover Metric
- Linear search with $O(n)$ query time often the fastest exact option

This talk

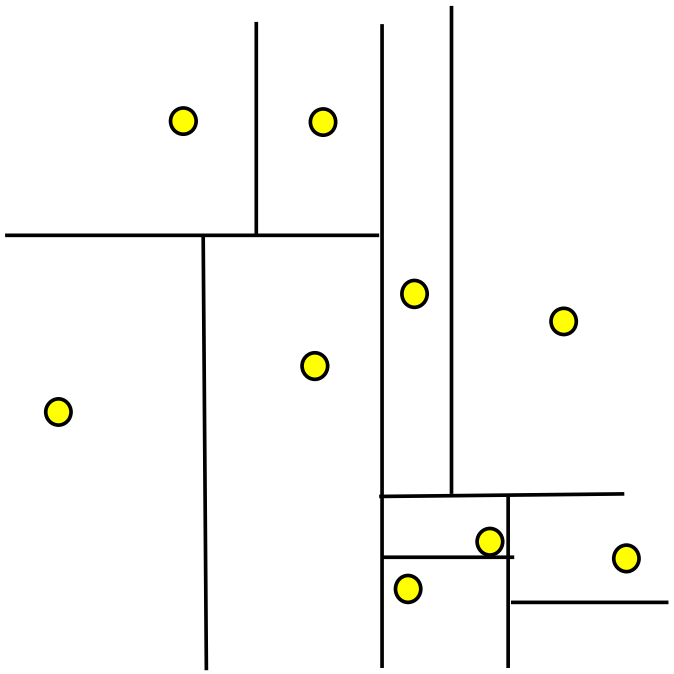
- Mildly exponential space or query time via kd-tree-based approach [AMNSW'98]
- Polynomial space and query time: LSH [IM'98]
- Extension to Hausdorff metric and EMD via *embeddings*

Kd-trees etc.

Kd-trees

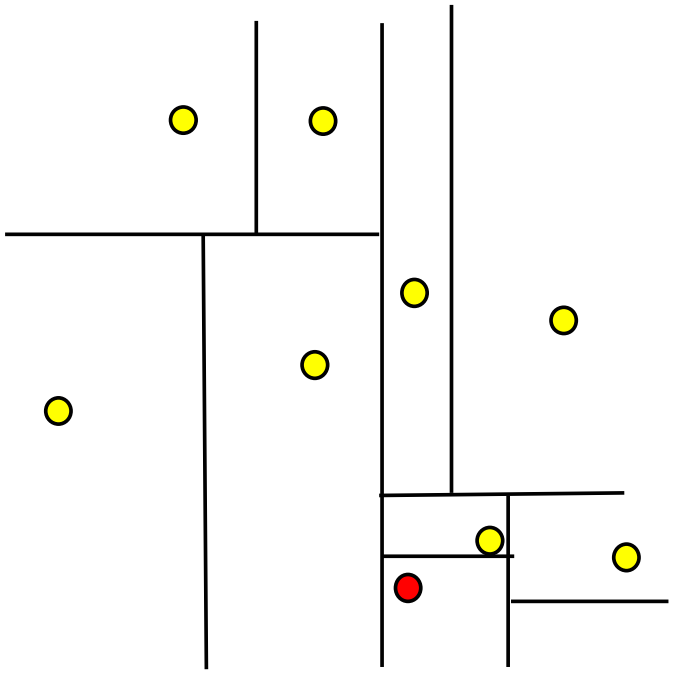
- Not the most efficient solution in theory
- Very efficient in practice
- Algorithm:
  - Choose x or y coordinate (alternate)
  - Choose the median of the coordinate; this defines a horizontal or vertical line
  - Recurse on both sides
- We get a binary tree:
  - Size: O(N)
  - Depth: O(log N)
  - Construction time: O(N log N)

Kd-tree: Example



ANN Query Procedure

- Locate the leaf cell containing q
- Enumerate all leaf cells L in the increasing order of distance r from q. For each L, update p' so that it is the closest point seen so far. Note:
  - r increases
  - D(q,p') decreases
- Stop when D(q,p')<c*r



Mildly exponential time

- If all leaf cells have bounded aspect ratio, we get:
  - Space: $O(n)$
  - Query time: $O_d(\log n/\varepsilon^d)$
- [AMNSW'98] removed the aspect ratio assumption (via better tree construction)
- Implementation available from David Mount's web page:
  - Works very fast even for fairly high dimensions
  - Quite conservative: can set large ε and still get good approximation (no guarantees though)
  - Does not require advance knowledge of ε

LSH

LSH

- A family H of functions $h: X \rightarrow U$ is called (P_1,P_2,R,cR)-sensitive [IM'98],if for any p,q:
  - if $D(p,q) < R$ then $Pr[h(p)=h(q)] > P_1$
  - if $D(p,q) > cR$ then $Pr[h(p)=h(q)] < P_2$
- Given H, we can solve a c-approximate Near Neighbor [IM'98,GIM'99]: if there is $p \in P$ with $D(q,p)<R$, then output $p' \in P$ with $D(q,p')<cR$
  - Query time: $O(d\, n^a \log n)$, $a=\log_{1/P2}(1/P_1)$
  - Space: $O(n^{a+1} + dn)$
  - Constant probability of failure

LSH for Hamming metric [IM'98]

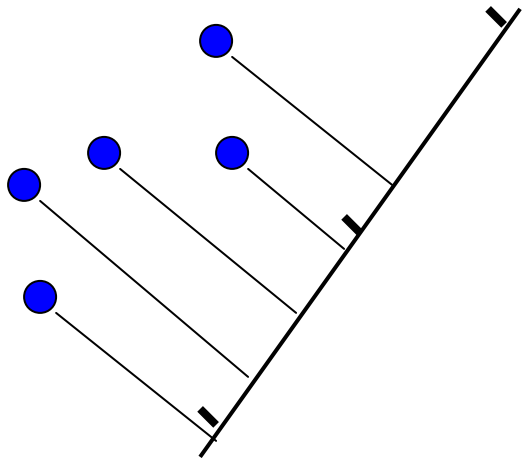
- Functions: $h(p)=p_i$, i.e., the i-th bit of p
- We have

$$\Pr[\, h(p)=h(q)\,] = 1-D(p,q)/d$$

- This gives exponent $a=1/c$
- Can embed l_1^d with coordinates in $\{1\ldots M\}$ into dM-dimensional Hamming space
- Can embed l_2^d into $l_1^{O(d)}$

LSH for l_2 [DIIM'03]

- Function: $h(p)= \lfloor p*X/r \rfloor$
  - $r \approx R$
  - $X=(X_1 \ldots X_d)$, where X_i is chosen from Gaussian distribution $N(0,1)$
- This gives exponent $a=1/c$
- Works directly in l_2
- Can extend to any l_s, $s \in (0,2]$
- Query time depends on #non-zero entries in p



Hashing

- We hash using $g(p)=h_1(p).h_2(p)\ldots h_k(p)$
  (for Hamming metric – sample k bits)

- Preprocessing: prepare indices for $g_1\ldots g_l$, $l=n^a$
- Add p: store p in buckets $g_1(p)\ldots g_l(p)$
- Delete p: remove p from above buckets
- Query: retrieve points from $g_1(q)\ldots g_l(q)$, return the closest one
- Query time: $O(dl)=O(dn^a)$

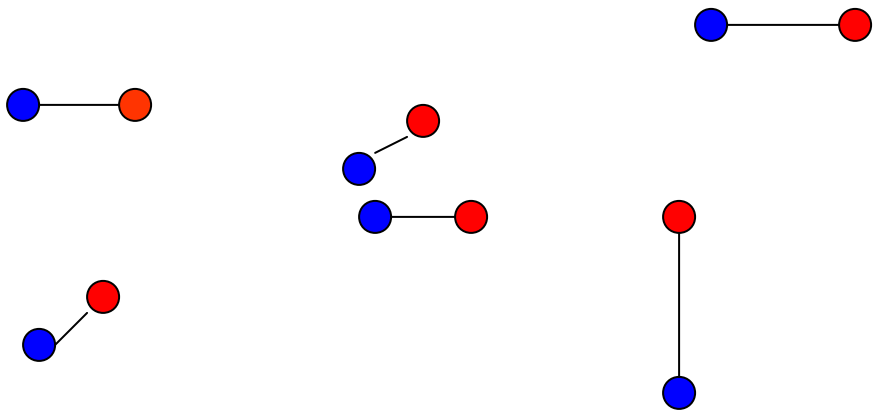
Beyond l_p norms

Hausdorff distance

- Given: two sets of points $P,Q \subseteq \{1...\Delta\}^d$, $|P|=|Q|$
- $DH(P,Q)=\max_{p\in P} \min_{q\in Q} ||p-q||_2$
- $H(P,Q)=\max[\ DH(P,Q),\ DH(Q,P)\ ]$

Earth-Mover Distance

- Given: two (multi)sets of points $P,Q \subseteq \{1...\Delta\}^d$, $|P|=|Q|$
- EMD(P,Q)=min-cost matching of P and Q



Applications

- Measures for dissimilarity between point-sets
- Huttenlocher et al used robust variants of H() for shape comparison

  (Question: any applications of the non-robust version ?)
- Rubner et al used EMD() for comparing
  - color histograms of images
  - texture information of images

NN via Embeddings

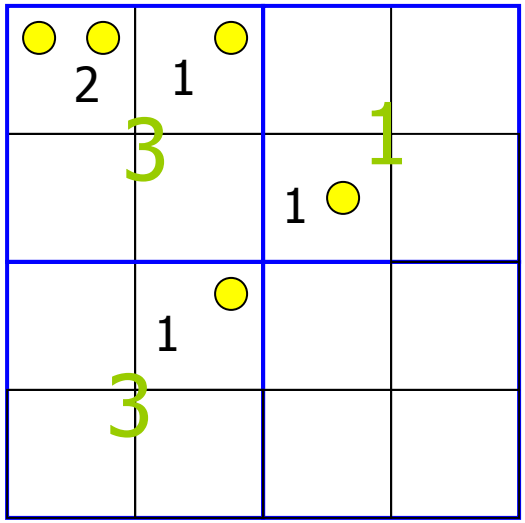
- Approach for NN under metric $M=(X,D)$:
  - Embed M into l_p^d (with distortion c)
  - Use c'-approximate NN under l_p^d
  - This gives cc' -approximate NN under M
- Known results:
  - [FCI'99]: Hausdorff over l_p^d into low-dimensional l_∞
  - [CPSV'00,MS'00,CM'02]: Block-edit distance into l_1
  - [Charikar'02, IT'03]: EMD into l_1

EMD into l_1

- Impose square grids $G_{-1}\ldots G_k$, with side lengths $2^{-1},2^0,2^1, \ldots, 2^k = \Delta$, shifted at random.
- For each square cell c in G_i, let $n^i_P(c)$ be the number of points in $c \cap P$.
- Embedding: P is mapped to

$$f(P) = 2^{-1}n^{-1}_P, \ldots , 2^k n^k_P$$

- Note: the dimension is really high!



Guarantees

- Theorem:
  - $EMD(P,Q) < O(||f(P)-f(Q)||_1)$
  - $E[||f(P)-f(Q)||_1] = O(\log \Delta) EMD(P,Q)$
- Due to:
  - Charikar'02+Kleinberg,Tardos'99+Bartal'96+ Peleg+Goel (personal communication)
  - Indyk,Thaper'02, Varadarajan'02

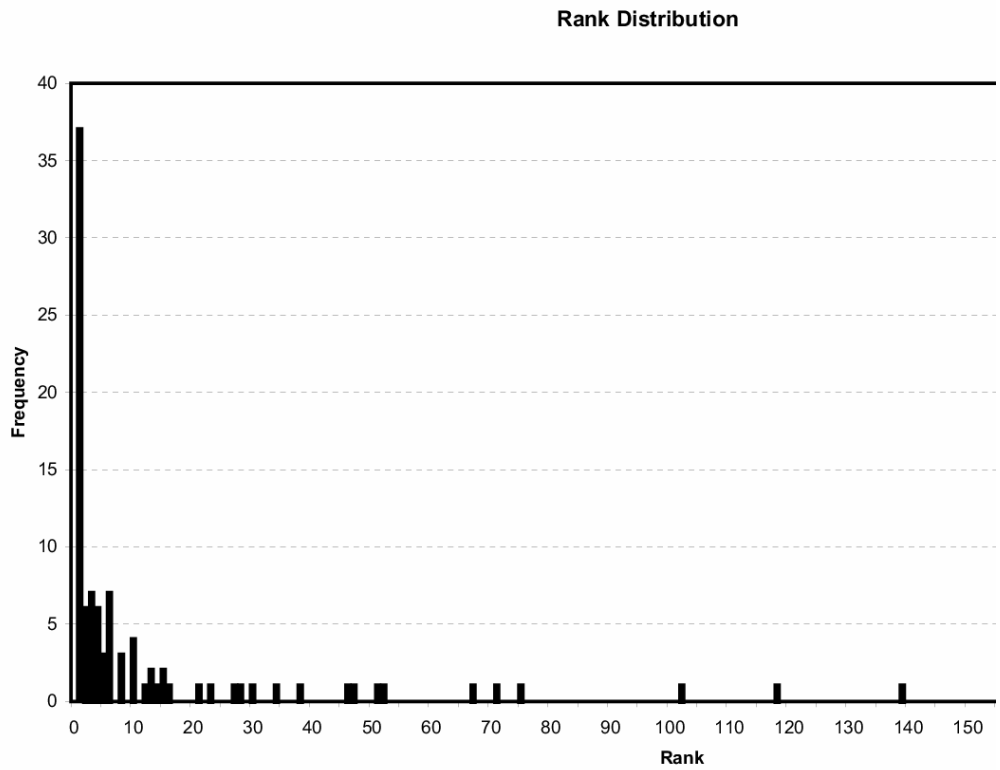
Proof sketch

- $EMD(P,Q) < O(||f(P)-f(Q)||_1)$:
  - Match r-b pairs in same cells of G_0: cost 2 per pair
  - ……G_1 : cost 4 per pair
  - etc.
- $E[||f(P)-f(Q)||_1] < O(\log \Delta)\ EMD(P,Q)$
  - Fix a matching, consider an edge $e=(p,q)$
  - Probability of cutting e by G_i is $\approx length(e)/2^i$
  - If e not cut, p and q are in the same cell so they do not contribute to $||f(P)-f(Q)||_1$

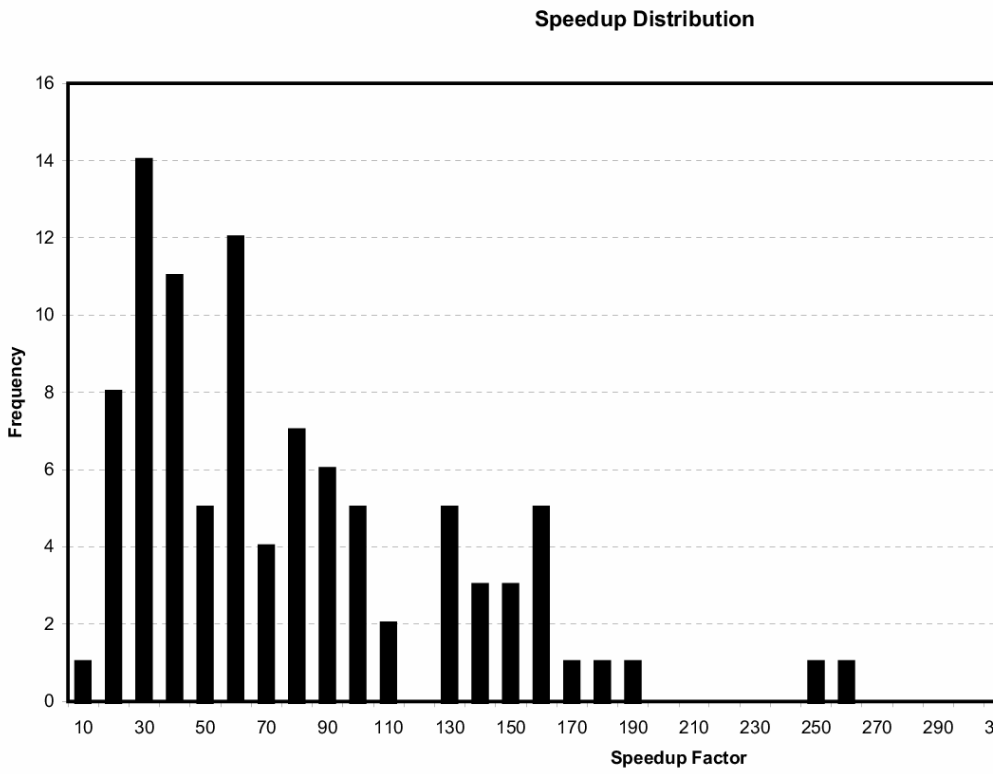
How does this work in practice? [IT'02]

- Data: color histograms of 20,000 Corel-Draw images:
  - Each pixel in an image is a point in 3D color space
  - Image represented by a set of pixels
- 100 queries
- LSH/Embedding parameters:
  - Probability of failure set to 10%
  - Embedding done 5 times per query
  - Approximation factor c set by hand
- Compare our approximate NN to the exact NN (w.r.t. EMD)

NN quality: rank



Speedup


Speedup Distribution
Frequency
16
14
12
10
8
6
4
2
0
10
30
50
70
90
110
130
150
170
190
210
230
250
270
290
Speedup Factor

References

- [AMNSW'98] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching", Journal of the ACM, 45 (1998).
- [Charikar'02] M. Charikar, "Similarity estimation techniques from rounding", Proc. 34th Symposium on Theory of Computing, 2002.
- [DIIM'03] M. Datar, N. Immorlica, P. Indyk and V. Mirrokni, "Locality-sensitive hashing based on p-stable distributions", manuscript, 2003.
- [FCI'99] M. Farach-Colton and P. Indyk, "Approximate Nearest Neighbor Algorithms for Hausdorff Metrics via Embeddings", Proc. *40th Symposium on Foundations of Computer Science*, 1999.
- [GIM'99] A. Gionis and R. Motwani, "Similarity Search in High Dimensions via Hashing", Proc. 25th International Conference on Very Large Databases (VLDB), 1999.
- [IM'98] P. Indyk and R. Motwani, "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality", Proc. 30th Symposium on Theory of Computing, 1998.
- [IT'03] P. Indyk and N. Thaper, "Fast Color Image Retrieval via Embeddings", Proc. 3rd International Workshop on Statistical and Computational Theories of Vision (at ICCV), 2003.