1 Time-space Trade-offs

Below we present a table of LSH algorithms using different space and time.

<table>
<thead>
<tr>
<th>Type</th>
<th>Space</th>
<th>Time</th>
<th>Comment</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low space and High query time</td>
<td>$\approx n$</td>
<td>$n^\sigma$</td>
<td>$\sigma = 2.09/c$</td>
<td>[Ind’01,Pan’06]</td>
</tr>
<tr>
<td>High query time</td>
<td>$\approx n$</td>
<td>$n^\sigma$</td>
<td>$\sigma = O(1/c^2)$</td>
<td>[AI’06]</td>
</tr>
<tr>
<td>Medium space and Medium query time</td>
<td>$n^{1+\rho}$</td>
<td>$n^\rho$</td>
<td>$\rho = 1/c$</td>
<td>[IM’98,DIIM’04]</td>
</tr>
<tr>
<td></td>
<td>$n^{1+\rho}$</td>
<td>$n^\rho$</td>
<td>$\rho = 1/c^2$</td>
<td>[AI’06]</td>
</tr>
<tr>
<td></td>
<td>$n^{1+\rho}$</td>
<td>$n^\rho$</td>
<td>$\rho \geq 1/c^2$</td>
<td>[MNP’06,OWZ’11]</td>
</tr>
<tr>
<td>High space and Low query time</td>
<td>$n^{\frac{d}{\epsilon^2}}$</td>
<td>$O(d\log n)$</td>
<td>(1 mem lookup)</td>
<td>$c = 1 + \epsilon$</td>
</tr>
<tr>
<td></td>
<td>$n^{\alpha(1/c^2)}$</td>
<td>$\omega(1)$ memory lookup</td>
<td>[PTW’08,PTW’10]</td>
<td></td>
</tr>
</tbody>
</table>

2 Near-linear Space for $\{0,1\}^d$

[Indyk’01,PanIgraphy’06]

- General idea: Sample a few bucket in the same hash table.

- Setting:
  - Close: $r = \frac{d}{2c}$ [Note that from last lecture $P_1 = 1 - \frac{r}{d} = 1 - \frac{1}{2c}$]
  - Far: $cr = \frac{d}{2}$ [Note that from last lecture $P_2 = 1 - \frac{cr}{d} = \frac{1}{2}$]

- Algorithm:
  - Use on hash table with $k = \frac{\log n}{\log 1/P_2} = \alpha \ln n$
    [Note that since $P_2 = 1/2$ here $\alpha$ is a constant]
  - On query q:
    * Compute $w = g(p) \in \{0,1\}^k$
    * Define $w'$ such that starting from $w$, flip each $w_j$ with probability $1 - P_1$
    * Lookup bucket $g(w')$ and compute distance to all points there
    * Repeat $R = n^\sigma$ times, stop if found an approximate near neighbor
Theorem. For $\sigma = \Theta(\log \frac{c}{c})$, we have

- $\Pr[\text{find an approximate near neighbor}] \geq 0.1$
- Expected runtime: $O(n^\sigma)$

Proof. Let $p^*$ be the near neighbor, then we know that $\|q - p^*\| \leq r$. Define $w = g(q)$, $t = \|w - g(p^*)\|_1$.

Claim 1. $\Pr[t \leq \frac{k}{c}] \geq \frac{1}{2}$

Proof. Note that $E[t] = \frac{r^d}{c} = \frac{k}{2c}$. Hence by Markov Inequality, $\Pr[t \leq \frac{k}{c}] \geq 1 - \frac{k}{2c} / \frac{k}{c} = \frac{1}{2}$ \hfill $\square$

Claim 2. $\Pr[w' = g(p)|\|q - p\|_1 \geq \frac{d}{2}] \leq \frac{1}{n}$

Proof.\[
\Pr[\text{Collision}] \leq (P_1 P_2 + (1 - P_1)(1 - P_2))^k \leq (P_2 (P_1 + 1 - 1) + (1 - P_1)(1 - P_2))^k \leq (P_2 + (1 - P_1)(1 - 2P_2))^k \leq P_2^k = 1/n \hfill \square
\]

Claim 3. $\Pr[w' = g(p^*)|\text{Claim 1}] \geq n^{-\sigma}$

Proof.\[
\Pr[w' = g(p^*)|\text{Claim 1}] = (1 - P_1)^t P_1^{k-t} \geq (1 - (1 - \frac{1}{2c}))^{k/c}(1 - \frac{1}{2c})^{k(1-1/c)} \geq (\frac{1}{2c})^{\frac{k}{c}} e^{-\frac{k}{2c}} \geq n^{-\frac{\Theta(1) \lg c - \alpha}{n - \frac{2c}{2c}}} \geq n^{-\sigma} \hfill \square
\]

Since if $w' = g(p^*)$ for at least one $w'$, we are guaranteed to output either $p^*$ or an approximate near neighbor, we are done by Claim 3. \hfill $\square$
3 Beyond LSH

Below we give a contrast of LSH algorithms and other algorithm.

In Hamming Space

<table>
<thead>
<tr>
<th>Type</th>
<th>Space</th>
<th>Time</th>
<th>Comment</th>
<th>$c = 2$</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSH</td>
<td>$n^{1+\rho}$</td>
<td>$n^\rho$</td>
<td>$\rho = 1/c$</td>
<td>$\rho = 1/2$</td>
<td>[IM’98]</td>
</tr>
<tr>
<td>Non-LSH</td>
<td>$n^{1+\rho}$</td>
<td>$n^\rho$</td>
<td>$\rho \geq 1/c$</td>
<td>$\rho = 1/3$</td>
<td>[MNP’06,OWZ’11]</td>
</tr>
</tbody>
</table>

In Euclidean Space

<table>
<thead>
<tr>
<th>Type</th>
<th>Space</th>
<th>Time</th>
<th>Comment</th>
<th>$c = 2$</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSH</td>
<td>$n^{1+\rho}$</td>
<td>$n^\rho$</td>
<td>$\rho \approx 1/c^2$</td>
<td>$\rho = 1/4$</td>
<td>[AI’06]</td>
</tr>
<tr>
<td>Non-LSH</td>
<td>$n^{1+\rho}$</td>
<td>$n^\rho$</td>
<td>$\rho \approx 1/2^{c^2-1}$</td>
<td>$\rho = 1/7$</td>
<td>[AINR’14,AR’15]</td>
</tr>
</tbody>
</table>

4 Data-dependent hashing

[A.-Indyk-Nguyen-Razenshteyn’14,A.-Razenshteyn’15]

- General idea: Using a random has function, which is chosen after seeing the given dataset
- Feature: Efficiently computable

- Components:
  - Nice geometric structure (has better data partition)
  - Reduction to such structure (depends on the data)

- Nice geometric structure:
  - Like a random dataset on a sphere s.t. random points at distance $\approx cr$
  - Query: At angle 45’ from near-neighbor

- Alg 1: Hyperplanes[Charikar’02]
  - We sample unit $r$ uniformly, hash $p$ into $sgn < r, p >$,
    $Pr[h(p) = h(q)] = 1 - \alpha/\pi$, where $\alpha$ is the angle between $p$ and $q$
  - $P_1 = 3/4, P_2 = 1/2$
  - $\rho \approx 0.42$

  - Sample $T$ i.i.d. standard $d$-dimensional Gaussians $g_2, g_2, \ldots, g_T$.
  - Hash $p$ into $h(p) = \arg\max_{1 \leq i \leq T} < p, g_i >$
  - Note that it is simply Hyperplane LSH when $T = 2$
• Hyperplane VS Voronoi
  – Hyperplane with \( k = 6 \) hyperplanes, which means we partition space into \( 2^6 = 64 \) pieces
  – Voronoi with \( T = 2^k = 64 \) vectors. \( \rho = 0.18 \)

– In Hyperplane algorithm we partition into grids while in Voronoi we partition into sphere

5 Nearest Neighbor Search: Conclusion

• Approach 1: Via sketches
• Approach 2: Locality Sensitive Hashing
  – Use Random Space Partitions
  – Algorithm with Better Space Bound
  – Use Data-dependent hashing