Large-Scale Similarity Joins With Guarantees

Rasmus Pagh
IT University of Copenhagen

SISAP
October 13, 2015
Talk outline

• In theory…
• The (approximate) similarity join problem
• Techniques for candidate set generation
  - Locality-sensitive hashing
  - Cache-efficiency via recursion
  - CoveringLSH: Achieving 100% recall
• In practice…
In theory...
Confession

• I was trained as an algorithm theorist
  - Worst case assumptions on data
  - Big-O notation
  - Papers full of math, rarely experiments
Confession

- I was trained as an algorithm theorist
  - Worst case assumptions on data
  - Big-O notation
  - Papers full of math, rarely experiments
- “In theory there is no difference between theory and practice… But in practice there is!”
Figure 3: Our map of computer science: The map was constructed by embedding the conference graph into a 2-dimensional Euclidean space. Only top-tier conferences (according to Libra) are shown. Note that the map only represents pairwise distances, there is no notion of orientation, i.e. the axes can be chosen arbitrarily.

Source: Kuhn & Wattenhofer. The Theoretic Center of Computer Science
Figure 3: Our map of computer science: The map was constructed by embedding the conference graph into a 2-dimensional Euclidean space. Only top-tier conferences (according to Libra) are shown. Note that the map only represents pairwise distances, there is no notion of orientation, i.e. the axes can be chosen arbitrarily.

Source: Kuhn & Wattenhofer. The Theoretic Center of Computer Science
Theory with impact

• Almost always algorithms that are easy to describe, implement, and adapt
Theory with impact

- Almost always algorithms that are easy to describe, implement, and adapt
- Analysis often simple enough to be taught to undergraduates
Theory with impact

• Almost always algorithms that are easy to describe, implement, and adapt

• Analysis often simple enough to be taught to undergraduates

• Solving a real problem
NEW YORK, NY, April 9, 2013—ACM (the Association for Computing Machinery) today announced the winners of six prestigious awards [...] Andrei Broder, Moses Charikar, and Piotr Indyk, recipients of the Paris Kanellakis Theory and Practice Award for algorithms that allow for quickly finding similar entries in large databases, known as locality-sensitive hashing (LSH). [...] The Kanellakis Award honors specific theoretical accomplishments that significantly affect the practice of computing.

Main contributions in the years 1997-2002
Now
Now

Soon?
Now

Soon?
Similarity joins
## Similarity join example 1

*(record linkage)*

<table>
<thead>
<tr>
<th>Country</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>USA</td>
<td>IBM</td>
</tr>
<tr>
<td>USA</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Germany</td>
<td>SAP</td>
</tr>
<tr>
<td>China</td>
<td>Baidu</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Token</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mircosoft</td>
<td>1</td>
</tr>
<tr>
<td>SAP SE</td>
<td>2</td>
</tr>
<tr>
<td>I.B.M.</td>
<td>3</td>
</tr>
<tr>
<td>baidu.com</td>
<td>4</td>
</tr>
</tbody>
</table>
### Similarity join example 1
(record linkage)

<table>
<thead>
<tr>
<th>Country</th>
<th>Name</th>
<th>Token</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>USA</td>
<td>IBM</td>
<td>Mircosoft</td>
<td>1</td>
</tr>
<tr>
<td>USA</td>
<td>Microsoft</td>
<td>SAP SE</td>
<td>2</td>
</tr>
<tr>
<td>Germany</td>
<td>SAP</td>
<td>I.B.M.</td>
<td>3</td>
</tr>
<tr>
<td>China</td>
<td>Baidu</td>
<td>baidu.com</td>
<td>4</td>
</tr>
</tbody>
</table>
## Similarity join example 2 (classification)

<table>
<thead>
<tr>
<th>Class</th>
<th>Name</th>
<th>Image</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mammalia</td>
<td>Cat</td>
<td><img src="https://example.com/cat_image" alt="Cat Image" /></td>
<td>1</td>
</tr>
<tr>
<td>Mammalia</td>
<td>Dog</td>
<td><img src="https://example.com/dog_image" alt="Dog Image" /></td>
<td>2</td>
</tr>
<tr>
<td>Reptilia</td>
<td>Snake</td>
<td><img src="https://example.com/snake_image" alt="Snake Image" /></td>
<td>3</td>
</tr>
<tr>
<td>Aves</td>
<td>Parrot</td>
<td><img src="https://example.com/parrot_image" alt="Parrot Image" /></td>
<td>4</td>
</tr>
</tbody>
</table>
## Similarity join example 2 (classification)

<table>
<thead>
<tr>
<th>Class</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mammalia</td>
<td>Cat</td>
</tr>
<tr>
<td>Mammalia</td>
<td>Dog</td>
</tr>
<tr>
<td>Reptilia</td>
<td>Snake</td>
</tr>
<tr>
<td>Aves</td>
<td>Parrot</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Features</th>
<th>Image</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101101</td>
<td><img src="Image" alt="Cat Image" /></td>
<td>1</td>
</tr>
<tr>
<td>1001000</td>
<td><img src="Image" alt="Dog Image" /></td>
<td>2</td>
</tr>
<tr>
<td>1101101</td>
<td><img src="Image" alt="Snake Image" /></td>
<td>3</td>
</tr>
<tr>
<td>1101110</td>
<td><img src="Image" alt="Parrot Image" /></td>
<td>4</td>
</tr>
</tbody>
</table>
### Similarity join example 2 (classification)

<table>
<thead>
<tr>
<th>Class</th>
<th>Name</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mammalia</td>
<td>Cat</td>
<td>1100101</td>
</tr>
<tr>
<td>Mammalia</td>
<td>Dog</td>
<td>1101101</td>
</tr>
<tr>
<td>Reptilia</td>
<td>Snake</td>
<td>1011000</td>
</tr>
<tr>
<td>Aves</td>
<td>Parrot</td>
<td>1010111</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Features</th>
<th>Image</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101101</td>
<td><img src="image1.png" alt="Image" /></td>
<td>1</td>
</tr>
<tr>
<td>1001000</td>
<td><img src="image2.png" alt="Image" /></td>
<td>2</td>
</tr>
<tr>
<td>1101101</td>
<td><img src="image3.png" alt="Image" /></td>
<td>3</td>
</tr>
<tr>
<td>1101110</td>
<td><img src="image4.png" alt="Image" /></td>
<td>4</td>
</tr>
</tbody>
</table>

Images by Bodlina and Marek Szczepanek
Similarity join example 2
(classification)

<table>
<thead>
<tr>
<th>Class</th>
<th>Name</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mammalia</td>
<td>Cat</td>
<td>1100101</td>
</tr>
<tr>
<td>Mammalia</td>
<td>Dog</td>
<td>1101101</td>
</tr>
<tr>
<td>Reptilia</td>
<td>Snake</td>
<td>1011000</td>
</tr>
<tr>
<td>Aves</td>
<td>Parrot</td>
<td>1010111</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Features</th>
<th>Image</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101101</td>
<td><img src="image1.png" alt="Dog Image" /></td>
<td>1</td>
</tr>
<tr>
<td>1001000</td>
<td><img src="image2.png" alt="Snake Image" /></td>
<td>2</td>
</tr>
<tr>
<td>1101101</td>
<td><img src="image3.png" alt="Cat Image" /></td>
<td>3</td>
</tr>
<tr>
<td>1101110</td>
<td><img src="image4.png" alt="Parrot Image" /></td>
<td>4</td>
</tr>
</tbody>
</table>

Images by Bodlina and Marek Szczepanek
### Similarity join example 3
(recommendation)

<table>
<thead>
<tr>
<th>User</th>
<th>Follows</th>
</tr>
</thead>
<tbody>
<tr>
<td>@rasmuspagh1</td>
<td>{@ERC_SSS, @NateSilver538, @NatureNews, @sapinker, @techreview, …}</td>
</tr>
<tr>
<td>@Reza_Zadeh</td>
<td>{@medialab, @techreview, @AndrewYNg, …}</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>@simMachines</td>
<td>{@neiltyson, @NatureNews, @RichardDawkins, …}</td>
</tr>
</tbody>
</table>
Algorithmic problem

• Given a tolerance $r$ compute:

$$Q \otimes_r S = \{(q, x) \in Q \times S \mid \|q - x\| \leq r\}$$
Algorithmic problem

• Given a tolerance $r$ compute:

$$Q \otimes_r S = \{(q, x) \in Q \times S \mid \|q - x\| \leq r\}$$

Distance measure
Algorithmic problem

• Given a tolerance $r$ compute:

$$Q \times_r S = \{(q, x) \in Q \times S \mid ||q - x|| \leq r\}$$

• This talk:

Consider $n$ vectors in $\{0,1\}^d$ and Hamming distance.

$$q = 110\, 0101$$

$$x = 110\, 1101$$

$$||q - x|| = 1$$
Variants

• kNN similarity join

• Batched similarity search
A Faster Subquadratic Algorithm for Finding Outlier Correlations

Matti Karppa, Petteri Kaski, and Jukka Kohonen

LEMP:
Fast Retrieval of Large Entries in a Matrix Product

Christina Teflioudi
Max Planck Institute for Informatics
Saarbrücken, Germany
chtelio@mpi-inf.mpg.de

Rainer Gemulla
University of Mannheim
Mannheim, Germany
rgemull@informatik.uni-mannheim.de

Olga Mykytiuk
Sulzer GmbH
München, Germany
olga.mykytiuk@gmail.com

Finding Correlations in Subquadratic Time, with Applications to Learning Parities and the Closest Pair Problem

Gregory Valiant

Probabilistic Polynomials and Hamming Nearest Neighbors

Batch Hamming Nearest Neighbor Problem

Josh Alman* Ryan Williams†
Similarity join in a picture

\[ R = Q \Join_r S \]
Similarity join in a picture

\[ R = Q \Join_r S \]
Similarity join in a picture

\[ R = Q \bowtie_r S \]
Similarity join in a picture in high dimensional space ($d=80$)
Why is this hard?
Why is this hard?

Because of the CURSE of dimensionality!
Why is this hard?

- [Williams ’04], [Alman & Williams ’15]:
  Hamming similarity search in time $n^{0.99^2}2^{o(d)} \Rightarrow$
  $k$-SAT w. $n$ variables can be solved in time $c^n$, $c < 2$
Why is this hard?

- [Williams ’04], [Alman & Williams ’15]:
  Hamming similarity search in time $n^{0.99} \cdot 2^{o(d)} \Rightarrow$
  $k$-SAT w. $n$ variables can be solved in time $c^n$, $c < 2$

Strong ETH states that this is not possible
Williams’ attempts at disproving SETH have borne considerable fruit. For example, in October he will present a new algorithm for solving the “nearest neighbors” problem. The advance grew out of a failed attempt to disprove SETH.
c-approximate similarity join

\[ C \supseteq Q \bowtie_r S \]
$c$-approximate similarity join

$C \geq Q \bowtie_r S$
\( c \)-approximate similarity join

\[ C \supseteq Q \Join_r S \]

False positive pairs, filter away in time
\[ |Q \Join_{cr} S| \]
Effect of approximation

MNIST data set (unary encoding)

Fraction of all pairs within Hamming distance

\[ r \]
Effect of approximation

MNIST data set (unary encoding)

Fraction of all pairs within Hamming distance

$Q \bowtie_r S$
Effect of approximation

MNIST data set (unary encoding)

Fraction of all pairs within Hamming distance

$Q \bowtie_{cr} S$

$Q \bowtie_r S$
Effect of approximation

MNIST data set (unary encoding)

Additional fraction of pairs that may become candidates

Within Hamming distance
Effect of approximation

MNIST data set (unary encoding)

In rest of the talk:
Assume $|Q \diamond_{cr} S|$ is not much larger than $|Q \diamond_r S|$.

Additional fraction of pairs that may become candidates

Fraction of all pairs within Hamming distance
Candidate set generation
- Locality-sensitive hashing (LSH)
Locality-sensitive hashing

[Indyk & Motwani '98]
Locality-sensitive hashing

Idea: Consider projection onto a random subset of dimensions, each chosen with probability $p$
Locality-sensitive hashing

Idea: Consider projection onto a random subset of dimensions, each chosen with probability $p$. 

[Indyk & Motwani '98]
Locality-sensitive hashing

Idea: Consider projection onto a random subset of dimensions, each chosen with probability $p$

$h(x) = x \land a$
Locality-sensitive hashing

Idea: Consider projection onto a random subset of dimensions, each chosen with probability \( p \)

Candidate match

\[ h(x) = x \wedge a \]
Locality-sensitive hashing

**Idea**: Consider projection onto a random subset of dimensions, each chosen with probability $p$.

Repeat enough times that vectors at distance $r$ produce at least one collision.

$h(x) = x \land a$
Locality-sensitive hashing

Idea: Consider projection onto a random subset of dimensions, each chosen with probability $p$

Candidate match

Repeat enough times that vectors at distance $r$ produce at least one collision

$h_i(x) = x \land a_i$
Probability of collision?

Collision probability for $i$th hash table:

$$\Pr[x \land a_i = q \land a_i] = (1 - p)^{||x-q||} \approx e^{-p ||x-q||}$$

[Indyk & Motwani ’98]
Probability of collision?

Collision probability for $i$th hash table:

$$\Pr[x \land a_i = q \land a_i] = (1 - p)^{||x - q||} \approx e^{-p||x - q||}$$

With $p = \frac{\ln(n)}{cr}$, probability

$\approx 1/n$ at distance $cr$ and $\approx 1/n^{1/c}$ at distance $r$
Probability of collision?

Collision probability for $i$th hash table:

$$\Pr[x \land a_i = q \land a] = (1 - p)^{||x-q||} \approx e^{-p||x-q||}$$

With $p = \frac{\ln(n)}{cr}$, probability

$\approx 1/n$ at distance $cr$ and

$\approx 1/n^{1/c}$ at distance $r$
Probability of collision?

Collision probability for $i$th hash table:

$$\Pr[x \land a_i = q \land a_i] = (1 - p)^{||x-q||} \approx e^{-p||x-q||}$$

$t = n^{1/c}$ repetitions ensure constant success probability

Possibility of false negatives: Saying ‘No’ when ‘Yes’ is required

With $p = \frac{\ln(n)}{cr}$, probability

$\approx \frac{1}{n}$ at distance $cr$ and

$\approx \frac{1}{n^{1/c}}$ at distance $r$
Summary of analysis

Analysis (GIM ’99): Each bit of $a_i$ is 1 with probability $p$.

- Expect $c_p \leq n^2 (1-p)^{cr}$ candidate pairs outside $Q \bowtie_{cr} S$

- Probability of $h(q) = h(x)$ when $d(q, x) = r$ is $(1-p)^r$, so need to hash $t_p = (1-p)^{-r}$ times to achieve constant recall.

- Choose $p$ to balance cost of hashing $t_p$ times and checking $t_p c_p$ candidate pairs not in $Q \bowtie_{cr} S$
Summary of analysis

Analysis (GIM ’99): Each bit of $a_i$ is 1 with probability $p$.

- Expect $c_p \leq n^2 (1-p)^{cr}$ candidate pairs outside $Q \bowtie_{cr} S$
- Probability of $h(q)=h(x)$ when $d(q,x) = r$ is $(1-p)^r$, so need to hash $t_p=(1-p)^{-r}$ times to achieve constant recall.
- Choose $p$ to balance cost of hashing $t_p$ times and checking $t_p c_p$ candidate pairs not in $Q \bowtie_{cr} S$
Summary of analysis

Analysis (GIM ’99): Each bit of $a_i$ is 1 with probability $p$.

• Expect $c_p \leq n^2 (1-p)^{cr}$ candidate pairs outside $Q \bowtie_{cr} S$

• Probability of $h(q)=h(x)$ when $d(q,x) = r$ is $(1-p)^r$, so need to hash $t_p = (1-p)^{-r}$ times to achieve constant recall.

• Choose $p$ to balance cost of hashing $t_p$ times and checking $t_p c_p$ candidate pairs not in $Q \bowtie_{cr} S$
Locality-sensitive hashing

Analysis (GIM ’99): Each bit of $a_i$ is 1 with probability $p$.

- Expect $c_p \leq n^2(1-p)^{cr}$ candidate pairs outside $Q \bowtie_{cr} S$
- Probability of $h(q) = h(x)$ when $d(q,x) = r$ is $(1-p)^r$, so need to hash $t_p = (1-p)^{-r}$ times to achieve constant recall.
- Choose $p$ to balance cost of hashing $t_p$ times and checking $t_p c_p$ candidate pairs not in $Q \bowtie_{cr} S$
Locality-sensitive hashing

Analysis (GIM ’99): Each bit of $a_i$ is 1 with probability $p$.

- Expect $c_p \leq n^2 (1-p)^{cr}$ candidate pairs outside $Q \bowtie_{cr} S$
- Probability of $h(q) = h(x)$ when $d(q, x) = r$ is $(1-p)^r$, so need to hash $t_p = (1-p)^r$ times to achieve constant recall.
- Choose $p$ to balance cost of hashing $t_p$ times and checking $t_p c_p$ candidate pairs not in $Q \bowtie_{cr} S$

Number of operations is $O(dn^{1+1/c})$, expected;
**Newer theory developments**

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani STOC ’98</td>
<td>1/c</td>
<td>Worst case upper bound</td>
</tr>
</tbody>
</table>

* Focus on subquadratic space; lower order terms ignored.
**Newer theory developments**

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani</td>
<td>$1/c$</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>STOC ’98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou</td>
<td>$1/c$</td>
<td>Data indep. LSH, worst case lower bound</td>
</tr>
<tr>
<td>ITCS ’10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Focus on subquadratic space; lower order terms ignored.
## Newer theory developments*

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani</td>
<td>$1/c$</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>STOC ’98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou</td>
<td>$1/c$</td>
<td>Data indep. LSH, worst case lower bound</td>
</tr>
<tr>
<td>ITCS ’10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dubiner</td>
<td>$1/(2c-1)$</td>
<td>Random data upper bound</td>
</tr>
<tr>
<td>Trans. Inf. Theory ’10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Focus on subquadratic space; lower order terms ignored.
Newer theory developments*

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>$1$</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani STOC ’98</td>
<td>$1/c$</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou ITCS ’10</td>
<td>$1/c$</td>
<td>Data indep. LSH, worst case lower bound</td>
</tr>
<tr>
<td>Dubiner Trans. Inf. Theory ’10</td>
<td>$1/(2c-1)$</td>
<td>Random data upper bound</td>
</tr>
<tr>
<td>Andoni &amp; Razenshteyn STOC ’15</td>
<td>$1/(2c-1)$</td>
<td>Data dep. LSH, worst case upper bound</td>
</tr>
</tbody>
</table>

* Focus on subquadratic space; lower order terms ignored.
# Newer theory developments*

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani</td>
<td>1/c</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>STOC ’98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou</td>
<td>1/c</td>
<td>Data indep. LSH, worst case lower bound</td>
</tr>
<tr>
<td>ITCS ’10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dubiner</td>
<td>1/(2c-1)</td>
<td>Random data upper bound</td>
</tr>
<tr>
<td>Trans. Inf. Theory ’10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Andoni &amp; Razenshteyn</td>
<td>1/(2c-1)</td>
<td>Data dep. LSH, worst case upper bound and lower bound</td>
</tr>
<tr>
<td>STOC ‘15</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Focus on subquadratic space; lower order terms ignored.
Newer theory developments*

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani STOC ’98</td>
<td>1/c</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou ITCS ’10</td>
<td>1/c</td>
<td>Data indep. LSH, worst case lower bound</td>
</tr>
<tr>
<td>Dubiner Trans. Inf. Theory ’10</td>
<td>1/(2c-1)</td>
<td>Random data upper bound</td>
</tr>
<tr>
<td>Andoni &amp; Razenshteyn STOC ’15</td>
<td>1/(2c-1)</td>
<td>Data dep. LSH, worst case upper bound  and lower bound</td>
</tr>
<tr>
<td>Kapralov PODS ’15</td>
<td>4/(c+1)</td>
<td>Linear space upper bound</td>
</tr>
</tbody>
</table>

* Focus on subquadratic space; lower order terms ignored.
# Newer theory developments*

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani</td>
<td>$1/c$</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>STOC ’98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou</td>
<td>$1/c$</td>
<td>Data indep. LSH, worst case lower bound</td>
</tr>
<tr>
<td>ITCS ’10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dubiner Trans. Inf. Theory ’10</td>
<td>$1/(2c-1)$</td>
<td>Random data upper bound</td>
</tr>
<tr>
<td>Andoni &amp; Razenshteyn</td>
<td>$1/(2c-1)$</td>
<td>Data dep. LSH, worst case upper bound and lower bound</td>
</tr>
<tr>
<td>STOC ’15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kapralov PODS ’15</td>
<td>$4/(c+1)$</td>
<td>Linear space upper bound</td>
</tr>
<tr>
<td>Valiant FOCS ’12</td>
<td>$\frac{1}{4-\omega} + O\left(\frac{1-1/c}{\log d}\right)$</td>
<td>Batched search [random data]</td>
</tr>
</tbody>
</table>

* Focus on subquadratic space; lower order terms ignored. $\omega < 2.38
## Newer theory developments*

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>$1$</td>
<td></td>
</tr>
</tbody>
</table>
| Indyk & Motwani
STOC ’98 | $1/c$ | Worst case upper bound |
| O’Donnell, Wu & Zhou
ITCS ’10 | $1/c$ | Data indep. LSH, worst case lower bound |
| Dubiner
Trans. Inf. Theory ’10 | $1/(2c-1)$ | Random data upper bound |
| Andoni & Razenshteyn
STOC ’15 | $1/(2c-1)$ | Data dep. LSH, worst case upper bound and lower bound |
| Kapralov
PODS ’15 | $4/(c+1)$ | Linear space upper bound |
| Valiant
FOCS ’12 | $\frac{1}{4-\omega} + O\left(\frac{1-1/c}{\log d}\right)$ | Batched search [random data] |
| Karppa, Kaski & Kohonen
SODA ’16 | $\frac{2\omega-3}{3}$ | Batched search [$c>1$, random data, $\omega > 2.25$] |

* Focus on subquadratic space; lower order terms ignored. $\omega < 2.38$
# Newer theory developments*

<table>
<thead>
<tr>
<th>Reference</th>
<th>Exponent, search time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>Indyk &amp; Motwani</td>
<td>(1/c)</td>
<td>Data indep. LSH, worst case lower bound</td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou</td>
<td>(1/c)</td>
<td>Data indep. LSH, worst case lower bound</td>
</tr>
<tr>
<td>Dubiner</td>
<td>(1/(2c-1))</td>
<td>Random data upper bound</td>
</tr>
<tr>
<td>Andoni &amp; Razenshteyn</td>
<td>(1/(2c-1))</td>
<td>Data dep. LSH, worst case upper bound and lower bound</td>
</tr>
<tr>
<td>Kapralov</td>
<td>(4/(c+1))</td>
<td>Linear space upper bound</td>
</tr>
<tr>
<td>Valiant</td>
<td>(\frac{1}{4-\omega} + O\left(\frac{1-1/c}{\log d}\right))</td>
<td>Batched search [random data]</td>
</tr>
<tr>
<td>Karppa, Kaski &amp; Kohonen</td>
<td>(\frac{2\omega-3}{3})</td>
<td>Batched search ([c&gt;1, \text{random data, } \omega &gt; 2.25])</td>
</tr>
<tr>
<td>Alman &amp; Williams</td>
<td>(1 - \tilde{\Omega}(\log(n)/d))</td>
<td>Batched search, (c=1)</td>
</tr>
</tbody>
</table>

* Focus on subquadratic space; lower order terms ignored.  
\(\omega < 2.38\)
Can LSH work for large data?

• Quote from a database paper: “LSH needs large memory space and long processing time to achieve good performance when searching a massive dataset”.
Can LSH work for large data?

- Quote from a database paper: "LSH needs large memory space and long processing time to achieve good performance when searching a massive dataset".

For similarity join, space is linear: Process one hash function at a time.
Can LSH work for large data?

• Quote from a database paper: "LSH needs large memory space and long processing time to achieve good performance when searching a massive dataset".

• Other issues:
  - LSH parameters are pessimistic; chosen to work for worst-case data set.

For similarity join, space is linear: Process one hash function at a time.
Can LSH work for large data?

• Quote from a database paper: “LSH needs large memory space and long processing time to achieve good performance when searching a massive dataset”.

• Other issues:
  - LSH parameters are pessimistic; chosen to work for worst-case data set.
  - Poor use of internal memory: A simple nested loop join often has better I/O complexity.

For similarity join, space is linear: Process one hash function at a time.
On pessimism

• LSH chosen to ensure few collisions at distance $cr$, even in worst-case scenarios:
On pessimism

- LSH chosen to ensure few collisions at distance $cr$, even in worst-case scenarios:

Price paid: Also small collision probability at distance $r$, so need many repetitions.
Candidate set generation
- Cache-efficiency via recursion
I/O model

- Data is stored on an external storage device, with $B$ vectors / storage block
- Internal memory can hold $M$ vectors
- Count the number of block transfers between internal memory and external storage (I/Os)

Figure courtesy of Lars Arge
Cautious LSH

Joint work with Pham, Silvestri, and Stöckel

Idea: Apply a weak LSH with constant collision probability.
Cautious LSH
Joint work with Pham, Silvestri, and Stöckel

Idea: Apply a weak LSH with constant collision probability.
Cautious LSH
Joint work with Pham, Silvestri, and Stöckel
Cautious LSH
Joint work with Pham, Silvestri, and Stöckel

RECURSIVE SUBPROBLEM $Q_0 \bowtie_r S_0$

RECURSIVE SUBPROBLEM $Q_1 \bowtie_r S_1$
Cautious LSH
Joint work with Pham, Silvestri, and Stöckel

Subproblems of size $M$ or less require no further I/Os
Example, cautious LSH

- Assume weak LSH collision prob. 0.9.
- Prob. that q and x collide i times is $0.9^i$.
- If depth is log $n$, success probability is $\approx n^{-0.152}$.
Example, cautious LSH

- Assume weak LSH collision prob. 0.9.
- Prob. that \( q \) and \( x \) collide \( i \) times is \( 0.9^i \).
- If depth is \( \log n \), success probability is \( \approx n^{-0.152} \).

\[
\begin{align*}
Q_0 \bowtie_r S_0 \\
Q_0 \bowtie_r S_0 & \\
Q_0 \bowtie_r S_0 & \\
Q_0 \bowtie_r S_0 & \\
Q_0 \bowtie_r S_0 & \\
Q_1 \bowtie_r S_1 \\
Q_1 \bowtie_r S_1 & \\
Q_1 \bowtie_r S_1 & \\
Q_1 \bowtie_r S_1 & \\
Q_1 \bowtie_r S_1 & \\
Q_1 \bowtie_r S_1 & \\
\end{align*}
\]
Example 2, cautious LSH

• Assume weak LSH collision prob. 0.5.
• Prob. that $q$ and $x$ collide $i$ times is $0.5^i$. 
Example 2, cautious LSH

- Assume weak LSH collision prob. 0.5.
- Prob. that $q$ and $x$ collide $i$ times is $0.5^i$.
- Idea: Recurse *twice* at each node.
Example 2, cautious LSH

- Assume weak LSH collision prob. 0.5.
- Prob. that $q$ and $x$ collide $i$ times is $0.5^i$.
- Idea: Recurse *twice* at each node.

# subproblems containing $q$ and $x$:
1 at each level, expected!
Aside:
Don’t be fooled by great expectations

WolframAlpha output:
100 Teratons / 66*10^6 years

48.05 kg/s (kilograms per second)

≈ 0.3 x rate of trash production by New York City (≈ 1 x 10^7 kg/day)
Aside:
Don’t be fooled by great expectations

What is the expected TNT equivalent of asteroid impacts during this talk?

WolframAlpha

100 Teratons / 66*10^6 years

48.05 kg/s (kilograms per second)

≈ 0.3 \times \text{rate of trash production by New York City} (≈ 1 \times 10^7 \text{ kg/day})
Branching processes

- **Abstract setting:**
  - One pair \((q,x)\) at root problem.
  - Generate \(t\) subproblems such that \((q,x)\) is “reproduced” in each with probability \(\geq 1/t\).
Branching processes

- **Abstract setting:**
  - One pair \((q,x)\) at root problem.
  - Generate \(t\) subproblems such that \((q,x)\) is “reproduced” in each with probability \(\geq 1/t\).

- What is the probability that \((q,x)\) is *extinct* at recursive level \(i\)?
Branching processes

• **Abstract setting:**
  - One pair \((q, x)\) at root problem.
  - Generate \(t\) subproblems such that \((q, x)\) is “reproduced” in each with probability \(\geq 1/t\).

• What is the probability that \((q, x)\) is *extinct* at recursive level \(i\)?
  - From theory of branching processes: \(\Omega(1/\sqrt{i})\)
I/O complexity, sketch

recursion
tree
I/O complexity, sketch

Size of all subproblems

\[ n \]

recursion
tree
I/O complexity, sketch

Size of all subproblems

n
nt

recursion
tree
I/O complexity, sketch

Size of all subproblems

\[ n \]
\[ nt \]
\[ nt^2 \]
\[ nt^3 \]

\[ \vdots \]

recursion tree
I/O complexity, sketch

Size of all subproblems

\[ n \]
\[ nt \]
\[ nt^2 \]
\[ nt^3 \]

... 

recursion tree

Expected average

# points at distance > cr.
I/O complexity, sketch

Size of all subproblems:
- $n$
- $nt$
- $nt^2$
- $nt^3$
- ...

recursion tree

Expected average

#points at distance > cr.

$n$

$n/ t^c$
I/O complexity, sketch

Size of all subproblems

\[ n \]
\[ nt \]
\[ nt^2 \]
\[ nt^3 \]
\[ \vdots \]

Expected average

\[ n \]
\[ n/t^c \]
\[ n/t^{2c} \]
\[ n/t^{3c} \]
\[ \vdots \]

#points at distance > cr.

recursion

tree
I/O complexity, sketch

Size of all subproblems:

- \( n \)
- \( nt \)
- \( nt^2 \)
- \( nt^3 \)

Expected average number of points at distance > \( cr. \):

- \( n \)
- \( n/t^c \)
- \( n/t^{2c} \)
- \( n/t^{3c} \)

In-memory computation:

Recursion tree:
I/O complexity, sketch

Size of all subproblems

$\log_t c \left( \frac{n}{M} \right)$

In-memory computation

# points at distance > cr.
I/O complexity, sketch

Size of all subproblems

- $n$
- $nt$
- $nt^2$
- $nt^3$

In-memory computation

Recursion tree

Pessimistic

Size of subproblems

- $n$
- $n/t^c$
- $n/t^{2c}$
- $n/t^{3c}$

#points at distance $> cr.$

Expected average

In-memory computation

$n/M$
I/O complexity

- Simplified:

\[ O \left( \left( \frac{n}{M} \right)^{1/c} \left( \frac{n}{B} + \frac{|Q \bowtie_r S|}{MB} \right) \right) \] I/Os

Assuming \(|Q \bowtie_{cr} S|\) is not much larger than \(|Q \bowtie_r S|\)
I/O complexity

- Simplified:

\[
O \left( \left( \frac{n}{M} \right)^{1/c} \left( \frac{n}{B} + \frac{|Q \bowtie_{cr} S'|}{MB} \right) \right) \text{ I/Os}
\]

Assuming \( |Q \bowtie_{cr} S'| \) is not much larger than \( |Q \bowtie_r S'| \)
I/O complexity

- Simplified:

\[ O \left( \left( \frac{n}{M} \right)^{1/c} \left( \frac{n}{B} + \frac{|Q \bowtie_{cr} S|}{MB} \right) \right) \]

Cost of reading input

Cost of generating output

Assuming \( |Q \bowtie_{cr} S| \) is not much larger than \( |Q \bowtie_{r} S| \)
I/O complexity

- Simplified:

\[
O\left(\left(\frac{n}{M}\right)^{1/c}\left(\frac{n}{B} + \frac{|Q \bowtie r S|}{MB}\right)\right)\]  

I/Os

Assuming $|Q \bowtie_{cr} S|$ is not much larger than $|Q \bowtie_r S|$
I/O complexity

- Simplified:

\[ \tilde{O} \left( \left( \frac{n}{M} \right)^{1/c} \left( \frac{n}{B} + \frac{|Q \otimes_r S|}{MB} \right) \right) \text{ I/Os} \]

- In general:

\[ O \left( \left( \frac{n}{M} \right)^{1/c} \left( \frac{n}{B} + \frac{|Q \otimes_r S|}{MB} \right) \right) \text{ I/Os} \]
Candidate set generation

- CoveringLSH: Achieving 100% recall
Efficient Exact Set-Similarity Joins

Arvind Arasu  
Microsoft Research  
One Microsoft Way  
Redmond, WA 98052  
arvinda@microsoft.com

Venkatesh Ganti  
Microsoft Research  
One Microsoft Way  
Redmond, WA 98052  
vganti@microsoft.com

Raghav Kaushik  
Microsoft Research  
One Microsoft Way  
Redmond, WA 98052  
skaushik@microsoft.com

ABSTRACT

Given two input collections of sets, a set-similarity join (SSJoin) identifies all pairs of sets, one from each collection, that have high similarity. Recent work has identified SSJoin as a useful primitive operator in data cleaning. In this paper, we present a number of algorithms [6, 22] do not provide any such guarantee. Previous work [8, 15] has proposed probabilistic algorithms based on the idea of locality-sensitive hashing (LSH) [13] that have similar guarantees. However, these algorithms are approximate since they can miss some output set pairs; in contrast, all of our algorithms are exact, and never produce a wrong output. We believe our algorithms are the first exact ones with such performance guarantees.

Data conflation, for example, the same address could be encoded using different strings in different records in the collection. Multiple representations arise due to a variety of reasons such as misspellings caused by typographic errors and different formatting conventions used by different

<table>
<thead>
<tr>
<th>City</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Los Angeles</td>
<td>CA</td>
</tr>
<tr>
<td>Palo Alto</td>
<td>CA</td>
</tr>
<tr>
<td>San Diego</td>
<td>CA</td>
</tr>
<tr>
<td>Santa Barbara</td>
<td>CA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>City</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Los Angeles</td>
<td>California</td>
</tr>
<tr>
<td>San Diego</td>
<td>California</td>
</tr>
<tr>
<td>Santa Barbara</td>
<td>California</td>
</tr>
</tbody>
</table>

Informally, SSJoin is defined as follows: Given two input collections, one collection contains source data, and the other contains target data. SSJoin is defined as the set of all pairs of sets, one from each collection, such that the similarity between the sets is above a specified threshold.
Approximation

Two kinds of approximation:

• Approximate distances

• Allow false positives and negatives (precision and recall below 100%)
Approximation

Two kinds of approximation:

• Approximate distances – Inherent to LSH
• Allow false positives and negatives (precision and recall below 100%)
Approximation

Two kinds of approximation:

• Approximate distances

• Allow false positives and negatives (precision and recall below 100%)

Inherent to LSH

Total recall possible!
Basic correlated LSH partitioning

[Arasu, Ganti & Kaushik '06]
Basic correlated LSH
partitioning

$\{\text{[Arasu, Ganti & Kaushik '06]}\}$

$h_1$
Basic correlated LSH
partitioning

$h_2$
Basic correlated LSH partitioning

\[ h_3 \]
Basic correlated LSH partitioning

\[ h_4 \]
Basic correlated LSH partitioning

For Hamming distance $\leq 3$, a collision is guaranteed!
Basic correlated LSH partitioning

For Hamming distance \( \leq 3 \), a collision is guaranteed!

[Arasu et al. '06]: To bound probability of collision for distance \( > 3 \) randomly permute the dimensions

[Arasu, Ganti & Kaushik '06]
Basic correlated LSH

equation

\[
h_{12}
\]
Basic correlated LSH

**enumeration**

\[ h_{13} \]

[Arasu, Ganti & Kaushik ’06]
Basic correlated LSH

<table>
<thead>
<tr>
<th>$h_{14}$</th>
</tr>
</thead>
</table>

[Arasu, Ganti & Kaushik '06]
Basic correlated LSH

equation

For Hamming distance $\leq 2$, a collision is guaranteed in $h_{12}$, $h_{13}$, $h_{14}$, $h_{23}$, $h_{24}$, $h_{34}$

$h_{14}$
Result in LSH framework

• The bit sampling LSH achieves:

\[ \Pr[h(q) = h(x) \mid \|x - q\| = cr] \leq \Pr[h(q) = h(x) \mid \|x - q\| = r]^c \]
Result in LSH framework

• The bit sampling LSH achieves:

\[
\Pr[h(q) = h(x) \mid \|x - q\| = cr] \leq \Pr[h(q) = h(x) \mid \|x - q\| = r]^c
\]

• Bound on \textit{PartEnum} construction with partitioning + enumeration + permutation:

\[
\Pr[h(q) = h(x) \mid \|x - q\| = cr] \leq \Pr[h(q) = h(x) \mid \|x - q\| = r]^{0.36c}
\]
Result in LSH framework

• The bit sampling LSH achieves:

\[
\Pr[h(q) = h(x) \mid \|x - q\| = cr] \leq \Pr[h(q) = h(x) \mid \|x - q\| = r]^c
\]

• Bound on *PartEnum* construction with partitioning + enumeration + permutation:

\[
\Pr[h(q) = h(x) \mid \|x - q\| = cr] \leq \Pr[h(q) = h(x) \mid \|x - q\| = r]^{0.36^c}
\]
Result in LSH framework

- The bit sampling LSH achieves:

\[ \Pr[h(q) = h(x) \mid \|x - q\| = cr] \leq \Pr[h(q) = h(x) \mid \|x - q\| = r]^c \]

- Bound on \textit{PartEnum} construction with partitioning + enumeration + permutation:

\[ \Pr[h(q) = h(x) \mid \|x - q\| = cr] \leq \Pr[h(q) = h(x) \mid \|x - q\| = r]^{0.36c} \]

Probabilistic argument suggests that it is possible to do much better. But how?

Need \(\approx 3\times\) larger \(c\)
Mathematical question

• A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;  
  - for every set of \(r\) columns there exists a row with 0s in all of them.
Mathematical question

- A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.
Mathematical question

• A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.
Mathematical question

• A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.

• Question:
  How few rows can such a matrix have?
Mathematical question

• A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.

• **Question:**
  How few rows can such a matrix have?
Mathematical question

• A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.

• Question:
How few rows can such a matrix have?

\[
\begin{pmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 \\
\end{pmatrix}
\]

\(p = 4/7\)
\(r = 2\)
Mathematical question

• A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.

• **Question:**
  How few rows can such a matrix have?

\[
\begin{pmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 \\
\end{pmatrix}
\]

\(p = 4/7\)
\(r = 2\)

Small collision probability
Collision guarantee
Number of hash functions
Mathematical question

• A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.

• Question:
  How few rows can such a matrix have?

\[
\begin{pmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 \\
\end{pmatrix}
\]

\[p = \frac{4}{7}, \quad r = 2\]
Mathematical question

- A \((p,r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.

- Question:
  How few rows can such a matrix have?

\[
\begin{pmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1
\end{pmatrix}
\]

\(p = 4/7\)
\(r = 2\)
Mathematical question

• A \((p, r)\)-covering matrix of dim. \(d\) satisfies:
  - Has \((1-p)d\) zeros and \(pd\) ones in each row;
  - for every set of \(r\) columns there exists a row with 0s in all of them.

• Question:
  How few rows can such a matrix have?

Related to covering problems in extremal combinatorics, but good constructions have been known only for \(p = O(1/d)\)
CoveringLSH

• Next: Answer for $d = 2^{r+1} - 1$, $pd = 2^r + 1 \approx d/2$
### CoveringLSH

- **Next**: Answer for \( d = 2^{r+1}-1 \), \( pd = 2^r + 1 \approx d / 2 \)

<table>
<thead>
<tr>
<th>Index (binary)</th>
<th>(001)</th>
<th>(010)</th>
<th>(011)</th>
<th>(100)</th>
<th>(101)</th>
<th>(110)</th>
<th>(111)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(001)</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(010)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>(011)</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>(100)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>(101)</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>(110)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(111)</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
CoveringLSH

- **Next:** Answer for $d = 2^{r+1}-1$, $pd = 2^r+1 \approx d/2$

**Idea:** Entry is dot product (mod 2) of row/column ID vectors (Hadamard code)
CoveringLSH

• **Next:** Answer for $d = 2^{r+1} - 1$, $pd = 2^r + 1 \approx d / 2$

**Idea:** Entry is dot product (mod 2) of row / column ID vectors (Hadamard code)

\[
\begin{bmatrix}
001 & 010 & 011 & 100 & 101 & 110 & 111 \\
001 & 010 & 011 & 100 & 101 & 110 & 111 \\
001 & 010 & 011 & 100 & 101 & 110 & 111 \\
001 & 010 & 011 & 100 & 101 & 110 & 111 \\
001 & 010 & 011 & 100 & 101 & 110 & 111 \\
001 & 010 & 011 & 100 & 101 & 110 & 111 \\
\end{bmatrix}
\]

\[(0,1,1) \cdot (0,1,0) = 1\]
Covering LSH

• **Next:** Answer for $d = 2^{r+1} - 1$, $pd = 2^r + 1 \approx d / 2$

**Idea:** Entry is dot product (mod 2) of row / column ID vectors (Hadamard code)

\[
\begin{align*}
\text{Index (binary)}: & \\
001 & \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \\
010 & \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \\
011 & \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \\
100 & \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \\
101 & \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{pmatrix} \\
110 & \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} \\
111 & \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}
\end{align*}
\]

\[(0,1,1) \cdot (0,1,1) = 0\]
CoveringLSH

- **Next**: Answer for $d = 2^{r+1} - 1$, $pd = 2^{r+1} \approx d / 2$

**Idea**: Entry is dot product (mod 2) of row / column ID vectors (Hadamard code)

\[
\begin{array}{cccccccc}
001 & 010 & 011 & 100 & 101 & 110 & 111 \\
001 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
010 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
011 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
100 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
101 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
110 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\
111 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
\end{array}
\]
Covering LSH

- **Next**: Answer for $d = 2^{r+1} - 1$, $pd = 2^{r+1} \approx d/2$

**Idea**: Entry is dot product (mod 2) of row/column ID vectors (Hadamard code)

![Hadamard matrix example](image-url)
CoveringLSH

- **Next:** Answer for $d = 2^{r+1}-1$, $pd = 2^r + 1 \approx d/2$

**Idea:** Entry is dot product (mod 2) of row/column ID vectors (Hadamard code)

**Lemma:**
For every set of $r$ vectors in $\{0,1\}^{r+1}$ there exists a nonzero vector that is orthogonal to all
CoveringLSH

- **Next:** Answer for \( d = 2^{r+1} - 1, \ p d = 2^{r+1} \approx d/2 \)

**Idea:** Entry is dot product (mod 2) of row / column ID vectors (Hadamard code)

**Lemma:**
For every set of \( r \) vectors in \( \{0,1\}^{r+1} \) there exists a nonzero vector that is orthogonal to all
Optimality?

• Could there be a smaller \((4/7,2)\)-covering?

• We need to “cover” \(\binom{7}{2} = 21\) sets of two columns
Optimality?

- Could there be a smaller \((4/7,2)\)-covering?

- We need to “cover” \(\binom{7}{2} = 21\) sets of two columns

- Each vector can cover \(\binom{3}{2} = 3\) sets of two columns

\[
\begin{pmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1
\end{pmatrix}
\]
Optimality?

• Could there be a smaller \((4/7,2)\)-covering?

• We need to “cover” \(\binom{7}{2} = 21\) sets of two columns.

• Each vector can cover \(\binom{3}{2} = 3\) sets of two columns.

\[
\begin{pmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 \\
\end{pmatrix}
\]

7 vectors is optimal!
Optimality?

- Could there be a smaller (4/7,2)-covering?
- We need to “cover” \( \binom{7}{2} = 21 \) sets of two columns
- Each vector can cover \( \binom{3}{2} = 3 \) sets of two columns
Optimality?

- Could there be a smaller (1/2,\(r\))-covering?
- We need to cover \(\binom{d}{r}\) sets of \(r\) columns.
Optimality?

- Could there be a smaller \((1/2, r)\)-covering?
- We need to cover \(\binom{d}{r}\) sets of \(r\) columns
- Each vector can cover \(\binom{d/2}{r}\) sets of \(r\) columns
Optimality?

- Could there be a smaller \((1/2,r)\)-covering?
- We need to cover \(\binom{d}{r}\) sets of \(r\) columns
- Each vector can cover \(\binom{d/2}{r}\) sets of \(r\) columns

\[
\frac{\binom{d}{r}}{\binom{d/2}{r}} > 2^r
\]

Within factor 2 of optimal!
Use in similarity search

• Combine with random permutation trick: Each bit sampled w. prob. 1/2 in each hash value.
Use in similarity search

• Combine with random permutation trick: Each bit sampled w. prob. 1/2 in each hash value.

• “Sweet spot” is for $r = \log(n) / c$:
  - $2^{r+1} = 2n^{1/c}$ hash functions
  - Collision probability $1/n$ at distance $cr = \log(n)$, so number of “far” collisions insignificant
Use in similarity search

• Combine with random permutation trick: Each bit sampled w. prob. 1/2 in each hash value.

• “Sweet spot” is for $r = \log(n)/c$:
  - $2^{r+1} = 2n^{1/c}$ hash functions
  - Collision probability $1/n$ at distance $cr = \log(n)$, so number of “far” collisions insignificant

• Matches bound of Indyk and Motwani.
Smaller radius?

• Can map vectors from \( \{0,1\}^d \) to \( \{0,1\}^{td} \), increasing all distances by an integer factor \( t \).

\[
q^t = 1100101110010111001011100101 \\
x^t = 1101101110110111011011101101
\]

• Try to “hit” sweet spot \( tr = \log(n)/c \)

- Details: arXiv:1507.03225 [cs.DS]
Example

$r=8, c=3$

- Linear search
- Exhaustive search in Hamming ball
- Classical LSH, error prob. $1/n$
- Classical LSH, error prob. 1%
- CoveringLSH (1 partition)
Larger radius?

- Partition to reduce to $pd < d/2$ sampled bits:
  - Iterate over $1/(2p)$ parts
Larger radius?

• Partition to reduce to $pd < d/2$ sampled bits:
  - Iterate over $1/(2p)$ parts

  Part 1 | Part 2 | ⋯ | Part $1/(2p)$

  $\begin{array}{cccccccccccccccccccc}
  1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  \end{array}$

• Distance in some part will be $\leq 2pr$
  - Use CoveringLSH with radius $2pr$ on each part
  - Details: arXiv:1507.03225 [cs.DS]
Example

\[ r=256, \ c=2 \]

Time

Linear search
Classical LSH, error prob. \( \frac{1}{n} \)
Classical LSH, error prob. 1%
CoveringLSH (multi-partition)

n (size of set)
Euclidean space - shifted grids

\[(d+1)r\]
Euclidean space - shifted grids

\[(d+1)r\]
Euclidean space - shifted grids

\[(d+1)r\]
Euclidean space - shifted grids

In same cell $\Rightarrow$

$d+1$-approximate near neighbor
Euclidean space - shifted grids

\((d+1)r\) 

In same cell \( \Rightarrow \)
d+1-approximate near neighbor

Time \(d+1\).
Approx. factor \(d+1\).
JL dimension reduction

Euclidean vector $x$

random linear mapping

Projection

Length concentrated around $\|x\|_2$
JL dimension reduction

Euclidean vector $x$

random linear mapping

Projection

Length concentrated around $\|x\|_2$

Lengths can increase $\Rightarrow$ false negatives possible
Correlated dimension reduction

Euclidean vector $x$

random rotation

Rotated vector $\pi(x)$

partitioning, scaling by $t$

Projection 1  Projection 2  ...  Projection $t$
Correlated dimension reduction

- Euclidean vector $x$
  - random rotation
  - Rotated vector $\pi(x)$
    - Length concentrated around $||x||_2$
    - partitioning, scaling by $t$
      - Projection 1
      - Projection 2
      - $\cdots$
      - Projection $t$
Correlated dimension reduction

Euclidean vector $x$

random rotation

Rotated vector $\pi(x)$

Length concentrated around $||x||_2$

partitioning, scaling by $t$

Projection 1  |  Projection 2  |  …  |  Projection $t$

Some projection will have length at most $||x||_2$
Correlated Euclidean LSH
(joint work with Matthew Skala)

- $dn\tilde{O}(1/c)$ hash functions
- Collision guaranteed within distance $r$
- Collision probability $1/n$ at distance $cr$
In practice...
In practice...

• What are the good use cases for similarity join?

• When is 100% recall of particular value?
In practice...

- What are the good use cases for similarity join?
- When is 100% recall of particular value?
Linking new theory to practice?

Match performance of classical (or data dep.) LSH without false neg.? (SODA ’16)
Linking new theory to practice?

- Match performance of classical (or data dep.) LSH without false neg.? (SODA ’16)

- Indexing: When is sublinear query time possible with linear space? (PODS ’15)
Linking new theory to practice?

- Match performance of classical (or data dep.) LSH without false neg.? (SODA ’16)
- Indexing: When is sublinear query time possible with linear space? (PODS ’15)
- Making new LSH constructions truly practical? (NIPS ’15)
Linking new theory to practice?

Match performance of classical (or data dep.) LSH without false neg.? (SODA ’16)

Indexing: When is sublinear query time possible with linear space? (PODS ’15)

Making new LSH constructions truly practical? (NIPS ’15)

Data dep. LSH that works in theory and in practice? (STOC ’15)
Thank you

To people with whom I have discussed material of this talk: Annalisa De Bonis, Francesco Silvestri, Ilya Razenshteyn, Johan von Tangen Sivertsen, Matthew Skala, Ninh Pham, Riko Jacob, Thomas Dybdahl Ahle, Tobias Christiani, Ugo Vaccaro, and more.

For economic support:
Thank you for your attention!