Conjunctive Filter: Breaking the Entropy Barrier

Daisuke Okanohara\textsuperscript{1, 2}
Yuichi Yoshida\textsuperscript{1, 3}

\textsuperscript{1} Preferred Infrastructure Inc.
\textsuperscript{2} Dept. of Computer Science, Univ. of Tokyo
\textsuperscript{3} School of Informatics, Kyoto University
Breaking the entropy barrier

• We want to store data structures using the space less than its entropy
  – The result should contain some errors

• Bloom Filter [Bloom 1970]
  – Store a set of keys, and given a key it answers whether the key exists or not.
  – It always report that a key exists if the key indeed exists, but false positive are allowed (one-sided error).
Problem:
Associate a key with a set of values

• \( f : X \rightarrow 2^V \)
  – Map from a key to a set of values
  – \( X \): The universe of keys, \(|X| = n\)
  – \( V \): The universe of values, \(|V| = m\)

• Example: Posting List
  – \( X \): words, \( V \): hit document IDs

<table>
<thead>
<tr>
<th></th>
<th>word1</th>
<th>2</th>
<th>7</th>
<th>15</th>
<th>32</th>
<th>33</th>
<th>37</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>word2</td>
<td>1</td>
<td>7</td>
<td>14</td>
<td>16</td>
<td>18</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>word3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>15</td>
<td>37</td>
<td></td>
</tr>
</tbody>
</table>
Entropy barrier storing a map

• For $t$ integers in $[1...m]$, we can store them in $t \log_2 m$ bits

• If we reduce the space further, we must admit too many errors
  – Storing 1 integer in $[1...2^{20}]$ requires 20 bit.
  – If we use only 5 bit, with one-sided error, since we can distinguish $2^5$ cases, each case should contain $2^{20}/2^5 = 32768$ integers

Too many false positives!
Conjunctive queries

- Can’t we reduce the space in the case of a set of values?
  ⇒ We can!

- If we only consider conjunctive queries (or AND query), the space can be reduced further with small errors.

\[
f^\cap(X) := \bigcap_{i=1}^{k} f(x_i)
\]

**Example:**
\[
f(x_1) = \{1, 2, 3, 5\} \quad f(x_2) = \{2, 3, 5\} \quad f(x_3) = \{2, 3, 4\}
\]
\[
X = (x_1, x_2, x_3) \quad f^\cap(X) = \{2, 3\}
\]
(\(\varepsilon, k\))-encoding

- Let \(s\) be a binary string expressing a map \(f\).
- A binary string \(s\) is called \((\varepsilon, k)\)-encodes a map if for \(k\)-conjunctive query, it holds
  1. One-sided error
     \[ v \in f(X) \text{ then } v \in s(X) \]
  2. False positive error rate
     \[
     \frac{|s(X) - f^\cup(X)|}{|V - f^\cup(X)|} < \varepsilon.
     \]
     \(f^\cup(X)\) is the result of a disjunctive query
Key Idea

• Although it is difficult to reduce the size for original queries, it is not for conj. queries.
  – False positives are randomly distributed, and filtered out by conjunction

\[ s(x_1) \cap s(x_3) \cap s(x_7) \]
Conjunctive Filter (1/4)

• Conjunctive filter \((\varepsilon, k)\)-encodes a map in a space-efficient manner.

• For simplicity, we assume here:
  – Only one value is associated with each key.
  – \(k=2\)

• Removing these assumptions is easy.
Conjunctive Filter (2/4)

- To save \((x, f(x))\):
  - Make random binary tree \(T_x\) with \(m\) leaves
    - \(x\) is used as a seed of a hash function.
    - Each leaf corresponds to a value.
  - Save \(v_x\) using the first \(\log_2 m/2\) bits of \(f(x)\)

\[\begin{align*}
\log_2 m & \downarrow \\
m \text{ leaves} & \downarrow \\
7 & 4 & 1 & 5 & 6 & 2 & 8 & 3 \\
\end{align*}\]
Conjunctive Filter (3/4)

- 2-conjunctive query on $x$ and $y$:
- Construct $T_x$ and $T_y$ and go down to $v_x$ and $v_y$:
- Let $S_x$ and $S_y$ be the sub-tree rooted at $v_x$ and $v_y$.

\[
\begin{align*}
\text{Take the intersection of values associated to leaves of } S_x \text{ and } S_y.
\end{align*}
\]

\[
\{6, 2\} \cap \{1, 6\} = \{6\} 
\]
Conjunctive Filter (4/4)

• Theorem: \( E[|S_x \cap S_y|] = O(1) \).

• Fact:
  - \( S_x \) and \( S_y \) are randomly distributed.
  - \(|S_x| = |S_y| = m^{1/2}\)
  - For an element \( v \) not in \( f(x) \) nor \( f(y) \), the probability of \( v \) appearing in \( S_x \cap S_y \) is \( O(1/m) \)
Conjunctive Filter
General case

• For general $k$, we get $(1/m^{1/2}, k)$-encoding consuming $(\log_2 m)/k$ bits per element.
  
  – $1/k$ of the original size.

  – The lower bound to store $(1/m^{1/2}, k)$-encoding is $(\log_2 m)/2k$ bits (following slides)
Lower Bound on Space of Map (k=1)

• #bits per element for naïve map is:

\[
\frac{1}{l} \log_2 \left( \binom{nm}{l} \right) \sim \log_2 \frac{nm - l}{l}
\]

• #bits per element for an \((\varepsilon, 1)\)-encoding of a map is:

\[
\log_2 \frac{nm - l}{l + \varepsilon(nm - l)}
\]

• Proof idea: count how many maps one bit-string can can encode.
Lower Bound on Space of Map \((k>1)\)

• \#bits per element for an \((\varepsilon, k)\)-encoding of a map is:
  \[
  \frac{1}{k} \log_2 \frac{nm - l}{l + \varepsilon(nm - l)}
  \]

Proof Idea: Duplicate each row \(k\) times.

• If we can perform \(k\)-conjunctive query on the duplicated map, we can restore the original map.
• Use the previous bound.
Application: full-text search with long queries

- Problem: Given a query $q$, return the position list of $q$ in the target text $T$.

q = abracadabra

de decompose

$q' = \{abr, aca, dab, ra\}$

Conjunctive query on $q'$

We remove nodes so that each node has at most predefined # positions.
Experiment Setting

• Data: IMDB data set, actors section
  – 1103393 actors (keys, $n$)
  – 1791274 movies (value, $m$)
  – 6493558 relations (who acts in which movies)
• Query : a set of actors
• Result : a set of movies in which all actors act
### Experiment 1

The size of conjunctive filters

<table>
<thead>
<tr>
<th>Naive Encoding</th>
<th>Conj. Filter Size (ratio to naive)</th>
<th>Lower bounds (ratio to naive)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Naive Encoding</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$k=2$</td>
<td>0.73</td>
<td>0.29</td>
</tr>
<tr>
<td>$k=3$</td>
<td>0.53</td>
<td>0.20</td>
</tr>
<tr>
<td>$k=4$</td>
<td>0.44</td>
<td>0.15</td>
</tr>
</tbody>
</table>

The size of conjunctive filter is smaller than raw, but it is still larger than the lower bound.
Experiment 2:

# false positives in 2-conj. queries

Almost results have at most $m^{1/2}$ errors

The result of 3-conjunctive queries is similar
Conclusion and Future Work

- In $k$-conjunctive queries, we can break the entropy barrier
  - when # elements is linear to # keys
- A conjunctive filter achieves an $(\varepsilon,k)$-encode map using $1/k$ of the original size

- Reduce the working space
  - Consider the entropy of value distribution
- Reduce the time complexity
  - Fast intersection
Conjunctive/Disjunctive queries

- **Query**: a set of keys  \( X = (x_1, x_2, \ldots, x_k) \)
- **Conjunctive query**  \( f^\cap(X) := \bigcap_{i=1}^{k} f(x_i) \)
- **Disjunctive query**  \( f^\cup(X) := \bigcup_{i=1}^{k} f(x_i) \)

**Example**

- \( f(x_1) = \{1, 3, 5\} \quad f(x_2) = \{2, 3, 5\} \quad f(x_3) = \{2, 3\} \)
- \( X = (x_1, x_2, x_3) \)

\[ f^\cap(X) = \{3\} \quad f^\cup(X) := \{1,2,3,5\} \]
Experiment 3: comparison of $|f^U(X)|$ and $s(X)$ in 2-conj. query

The result of 3-conjunctive queries is similar
Error Measures

- Let $s$ be a binary string of a map $f$.
- False Positives (1-Precision)

\[
\epsilon_X^+(s) = \frac{|s(X) \setminus f(X)|}{|\mathcal{V} \setminus f(X)|}
\]

- False Negatives (1 – Recall)

\[
\epsilon_X^-(s) = \frac{|f(x) \setminus s(X)|}{|f(X)|}
\]

- Union False Positives

\[
\epsilon_X^U(s) = \frac{|s(X) \setminus f^U(X)|}{|\mathcal{V} \setminus f^U(X)|}
\]