Experiments

C++, g++ 4.8.2 with -O3 flags
64bit Ubuntu Server 12.04 LTS version
Intel Xeon E5-2650 2.00GHz processor and 32GB memory.

Comparison with State-the-art

Evaluating Filters for top-k search

Scalability

Threshold-based Similarity Search Algorithm

Search:
1. Locate the right level
2. Generate substrings of query
3. Probe the inverted list, count the number of matched segments
4. Generate candidates
5. Perform verification

\[ \text{level} = \frac{1}{2}(\sqrt{\text{level}^2 - 2}) \]

\[ \text{query} \rightarrow \text{candidates} \rightarrow \text{verify} \]

Testing Candidates

Threshold Filter

Reduce number of substrings:

\[ \min(p_i, j - 1) \]

Remove invalid matching:

Improve Verification:
Multi-Extension method

Greedy Matching
Avoid duplicate search

Batched Pruning
Eliminate consecutive errors within a segment

HS-Tree Index

Iteratively String Partition:

Two disjoint segments, prefix and suffix
Until we reach a level that has segments of length 1
Generate tree nodes and inverted lists

An example of group 7 in HS-Tree index

The HS-Tree Index

Extend Segment Filter

Pass-Join: Segment Filter

Split \( r \) to \( r + 1 \) disjoint segments

Limitation:

Need a threshold before index construction
Can't support top-k similarity search

Threshold-based String Similarity Search: Given a string set \( S \) and a query string \( q \) and threshold \( \tau \), threshold-based string similarity search finds all strings \( s \in S \) that \( \text{ED}(s, q) \leq \tau \).

Top-k String Similarity Search: Given a string set \( S \) and a query string \( q \), top-k string similarity search returns \( k \) strings \( R \in S \) such that \( |R| = k \) and for any string \( r \in R \) and \( r \in S \setminus R \), \( \text{ED}(r, q) > \tau \).

\[ \text{ls} = \log_2(\tau + 1) \]

Location \( i = \log_2(\tau + 1) \)

Level \( i \) has \( 2^i \) segments

\[ 2^i \tau \text{ common segments between } s \text{ and } q \]

Threshold is needed before index construction
Can't support top-k similarity search

Threshold \( \tau \)
Query \( q \)
Data \( s \)