A Linear Size Index for Approximate String Matching

CPM 2006

Siu-Lung Tam

Joint work with:
Ho-Leung Chan  Tak-Wah Lam
Wing-Kin Sung  Swee-Seong Wong

2006-07-05 11:10 +0200
Problem

Index a text $S[1..n]$ over a constant-size alphabet $\Sigma$ w.r.t. an integer constant $k$.

For each pattern $P[1..m]$, search for all $S[i..n]$ such that

$\exists \ j \geq i \ \text{dist}(P, S[i..j]) \leq k$.

- We focus on Hamming distance in this talk, although edit distance works similarly.
Indexing for Approximate String Matching

Problem

Index a text $S[1..n]$ over a constant-size alphabet $\Sigma$ w.r.t. an integer constant $k$.
For each pattern $P[1..m]$, search for all $S[i..n]$ such that
$\exists j \geq i \ dist(P, S[i..j]) \leq k$.

- We focus on Hamming distance in this talk, although edit distance works similarly.
- Performance is measured by both
  (i) index size, and
  (ii) searching time, which can depend on
    - text size $n$,
    - alphabet size $|\Sigma|$,
    - number of errors $k$,
    - pattern length $m$,
    - number of matches $occ$
When $m$, $|\Sigma|$ and $k$ are small ...
When $m$, $|\Sigma|$ and $k$ are small ...

- Searching $P[1..m]$ for 1 error = Searching $m(|\Sigma| - 1)$
- "1-modified patterns"
Brute Force Searching

- When $m$, $|\Sigma|$ and $k$ are small ...
- Searching $P[1..m]$ for 1 error = Searching $m(|\Sigma| - 1)$ “1-modified patterns”
- Suppose it takes $O(f(m,n) + occ')$ time to search each 1-modified pattern for exact match.
  - Nowadays $f(m,n)$ is $O(\log n)$ or even $O(\log \log n)$. 
Brute Force Searching

- When $m$, $|\Sigma|$ and $k$ are small ...

- Searching $P[1..m]$ for 1 error = Searching $m(|\Sigma| - 1)$ “1-modified patterns”

- Suppose it takes $O(f(m, n) + occ')$ time to search each 1-modified pattern for exact match.
  - Nowadays $f(m, n)$ is $O(\log n)$ or even $O(\log \log n)$.

- $O(m|\Sigma| f(m, n) + occ)$ time to search $P$ for 1 error.
- $O(m^k|\Sigma|^k f(m, n) + occ)$ time to search $P$ for $k$ errors.
<table>
<thead>
<tr>
<th>Space</th>
<th>Time</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(n)$</td>
<td>$O(m^2 + \text{occ})$</td>
<td>Cobbs [CPM95]</td>
</tr>
</tbody>
</table>
### 1-Error Solutions

<table>
<thead>
<tr>
<th>Space</th>
<th>Time</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(n)$</td>
<td>$O(m^2 + occ)$</td>
<td>Cobbs [CPM95]</td>
</tr>
<tr>
<td>$O(n \log^2 n)$</td>
<td>$O(m \log n \log \log n + occ)$</td>
<td>Amir et al. [WADS99]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m \log \log n + occ)$</td>
<td>Bauchsbaum et al. [ESA00]</td>
</tr>
</tbody>
</table>
### 1-Error Solutions

<table>
<thead>
<tr>
<th>Space</th>
<th>Time</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(n)$</td>
<td>$O(m^2 + occ)$</td>
<td>Cobbs [CPM95]</td>
</tr>
<tr>
<td>$O(n \log^2 n)$</td>
<td>$O(m \log n \log \log n + occ)$</td>
<td>Amir et al. [WADS99]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m \log \log n + occ)$</td>
<td>Bauchsbaum et al. [ESA00]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m + \log n \log \log n + occ)$</td>
<td>Cole et al. [STOC04]</td>
</tr>
<tr>
<td>Space</td>
<td>Time</td>
<td>Authors</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(m^2 + \text{occ})$</td>
<td>Cobbs [CPM95]</td>
</tr>
<tr>
<td>$O(n \log^2 n)$</td>
<td>$O(m \log n \log\log n + \text{occ})$</td>
<td>Amir et al. [WADS99]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m \log \log n + \text{occ})$</td>
<td>Bauchschbaum et al. [ESA00]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m + \log n \log\log n + \text{occ})$</td>
<td>Cole et al. [STOC04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(m \log n + \text{occ})$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O((m \log n + \text{occ}) \log n)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
</tbody>
</table>
## 1-Error Solutions

<table>
<thead>
<tr>
<th>Space</th>
<th>Time</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(n)$</td>
<td>$O(m^2 + occ)$</td>
<td>Cobbs [CPM95]</td>
</tr>
<tr>
<td>$O(n \log^2 n)$</td>
<td>$O(m \log n \log \log n + occ)$</td>
<td>Amir et al. [WADS99]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m \log \log n + occ)$</td>
<td>Bauchsbaum et al. [ESA00]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m + \log n \log \log n + occ)$</td>
<td>Cole et al. [STOC04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(m \log n + occ)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O((m \log n + occ) \log n)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(m \log \log n + occ)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O((m \log \log n + occ) \log^\epsilon n)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
<tr>
<td>Space</td>
<td>Time</td>
<td>Authors</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(m^2 + \text{occ})$</td>
<td>Cobbs [CPM95]</td>
</tr>
<tr>
<td>$O(n \log^2 n)$</td>
<td>$O(m \log n \log \log n + \text{occ})$</td>
<td>Amir et al. [WADS99]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m \log \log n + \text{occ})$</td>
<td>Bauchsbaum et al. [ESA00]</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>$O(m + \log n \log \log n + \text{occ})$</td>
<td>Cole et al. [STOC04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(m \log n + \text{occ})$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O((m \log n + \text{occ}) \log n)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(m \log \log n + \text{occ})$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O((m \log \log n + \text{occ}) \log^e n)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(m + \log^3 n \log \log n + \text{occ})$</td>
<td>Our Result</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O((m + \log^4 n \log \log n + \text{occ}) \log^e n)$</td>
<td>Our Result</td>
</tr>
</tbody>
</table>
### k-Error Solutions in Literature

<table>
<thead>
<tr>
<th>Space</th>
<th>Time</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(n)$</td>
<td>$O((cm)^k \log n + occ)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O(((cm)^k \log n + occ) \log n)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O((cm)^k \log \log n + occ)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O(((cm)^k \log \log n + occ) \log^\epsilon n)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
</tbody>
</table>
### $k$-Error Solutions in Literature

<table>
<thead>
<tr>
<th>Space</th>
<th>Time</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O\left(\frac{c^k}{k!} n \log^k n\right)$</td>
<td>$O\left(m + \frac{c^k}{k!} \log^k n \log \log n + \text{occ}\right)$</td>
<td>Cole et al. [STOC04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O\left((cm)^k \log n + \text{occ}\right)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O\left(((cm)^k \log n + \text{occ}) \log n\right)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O\left((cm)^k \log \log n + \text{occ}\right)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O\left(((cm)^k \log \log n + \text{occ}) \log^\epsilon n\right)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
</tbody>
</table>
### $k$-Error Solutions in Literature

<table>
<thead>
<tr>
<th>Space</th>
<th>Time</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O\left(\frac{c^k}{k!} n \log^k n\right)$</td>
<td>$O\left(m + \frac{c^k}{k!} n \log^k n \log \log n + occ\right)$</td>
<td>Cole et al. [STOC04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O\left((cm)^k \log n + occ\right)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O\left(((cm)^k \log n + occ) \log n\right)$</td>
<td>Huynh et al. [CPM04]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O\left((cm)^k \log \log n + occ\right)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O\left(((cm)^k \log \log n + occ) \log^\epsilon n\right)$</td>
<td>Lam et al. [ISAAC05]</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O\left(m + (c \log n)^k(k+1) \log \log n + occ\right)$</td>
<td>Our Result</td>
</tr>
<tr>
<td>$O(n)$ bits</td>
<td>$O\left((m + (c \log n)^k(k+2) \log \log n + occ) \log^\epsilon n\right)$</td>
<td>Our Result</td>
</tr>
</tbody>
</table>
We combine

- “$k$-Errata Trees” by Cole, Gottlieb and Lewenstein [STOC04]
- “Tree Cross Product” by Buchsbaum, Goodrich and Westbrook [ESA00], and

If the pattern is “long-enough”
- this index allows us to locate $k$-error matches efficiently.

Otherwise
- use the best brute-force searching, by Lam, Sung and Wong [ISAAC05].
Let $\beta \leq |P|$ be an integer, later chosen as $\Theta(\log^{k+1} n)$.

We mark in text $S$, a checkpoint at every $\beta$ characters.

- e.g. $S[\beta]$, $S[2\beta]$, $S[3\beta]$, ... are checkpoints.

\[
\begin{array}{cccccccc}
\boxtimes & \boxtimes & \boxtimes & S & \boxtimes & \ldots
\end{array}
\]
Let $\beta \leq |P|$ be an integer, later chosen as $\Theta(\log^{k+1} n)$.

We mark in text $S$, a checkpoint at every $\beta$ characters
- e.g. $S[\beta]$, $S[2\beta]$, $S[3\beta]$, ... are checkpoints.

**Lemma**

Consider a substring $S[i..j]$ that matches $P$ with exactly $k$ errors. $S[i..j]$ contains at least one checkpoint.

In particular, $S[i..i+\beta-1]$ contains exactly one checkpoint.
Index $S$ for $k$-error searching only around checkpoints.

Let $\text{TAIL} = \{S[a..n] \mid a = \beta, 2\beta, \ldots\}$.
and $\text{HEAD} = \{S[1..b] \mid b + 1 = \beta, 2\beta, \ldots\}$. 
Index $S$ for $k$-error searching only around checkpoints.

Let $\text{TAIL} = \{S[a..n] \mid a = \beta, 2\beta, \ldots\}$. and $\text{HEAD} = \{S[1..b] \mid b + 1 = \beta, 2\beta, \ldots\}$. 

```
X X X S X X ... 
```

```
P 
```
Index $S$ for $k$-error searching only around checkpoints.

Let $\text{T AIL} = \{S[a..n] \mid a = \beta, 2\beta, \ldots\}$.
and $\text{HEAD} = \{S[1..b] \mid b + 1 = \beta, 2\beta, \ldots\}$.

Lemma

Consider a substring $S[i..j]$ that matches $P$ with $k$ errors. There exist non-negative integers $k_1 + k_2 = k$, such that some string in $\text{T AIL}$ has a prefix matching $P$ with $k_1$ errors, and some string in $\text{HEAD}$ has a suffix matching $P$ with $k_2$ errors.

Note that $|\text{T AIL}| = |\text{HEAD}| = O(n/\beta)$
Problem

Index all strings in \textit{TAIL} w.r.t. an integer $\ell$. Given a query $Q$, search for all strings with $Q'$ as a prefix, where $\text{dist}(Q, Q') = \ell$.

Lemma

The STOC04 index
- is a forest of compact tries where each leaf is a suffix of $S$;
- supports searching in $O(\log^\ell n \log \log n)$ time, if $Q$ is preprocessed using $O(|Q|)$ time on suffix tree of $S$;
- represents the strings found as the disjoint union over descendant leaves of $O(\log^\ell n)$ nodes in the compact tries;
- contains each suffix at most $O(\log^\ell n)$ times;
- takes $O(|TAIL| \log^\ell n)$ space.
Algorithm 1 Find all $k$-error matches of $P$ in $S$, for $|P| \geq \beta$

1: for $i$ in $[1, \beta]$ \{Conceptually cut $P$ into $P[1..i-1]$ and $P[i..m]$\}
do

7: end for
Algorithm 2 Find all $k$-error matches of $P$ in $S$, for $|P| \geq \beta$

1: for $i$ in $[1, \beta]$ {Conceptually cut $P$ into $P[1..i-1]$ and $P[i..m]$} do
2: for $k_2$ in $[0, k]$ {Search for $k_2$ and $k_1 = k - k_2$ errors respectively} do
6: end for
7: end for
Algorithm 3 Find all \( k \)-error matches of \( P \) in \( S \), for \( |P| \geq \beta \)

1: for \( i \) in \([1, \beta]\) \{Conceptually cut \( P \) into \( P[1..i-1] \) and \( P[i..m] \}\) 
   do
2:   for \( k_2 \) in \([0, k]\) \{Search for \( k_2 \) and \( k_1 = k - k_2 \) errors respectively\} 
   do
3:     Find all \( S[a..n] \in TAIL \) with a prefix matching \( P[i..m] \) with \( k_1 \) errors.
4:     Find all \( S[1..b] \in HEAD \) with a suffix matching \( P[1..i-1] \) with \( k_2 \) errors.
6:   end for
7: end for
Algorithm 4 Find all $k$-error matches of $P$ in $S$, for $|P| \geq \beta$

1: for $i$ in $[1, \beta]$ \{Conceptually cut $P$ into $P[1..i-1]$ and $P[i..m]$\}

2: for $k_2$ in $[0, k]$ \{Search for $k_2$ and $k_1 = k - k_2$ errors respectively\}

3: Find all $S[a..n] \in$ TAIL with a prefix matching $P[i..m]$ with $k_1$ errors.

4: Find all $S[1..b] \in$ HEAD with a suffix matching $P[1..i-1]$ with $k_2$ errors.

5: Find all “connecting pairs”: $S[a..n]$ and $S[1..b]$ where $a = b + 1$.

6: end for

7: end for
**Problem**

Index a set $I \subseteq V(T_1) \times V(T_2)$. Given a query $(u, w)$, search for all $(x, y) \in I$ where $x \in \text{descendants}(u)$ and $y \in \text{descendants}(w)$.

**Lemma**

The ESA00 index

- supports searching in $O(\log \log n + \text{occ})$ time;
- takes $O(|I| \log |V|)$ space, where $V = V(T_1) \cup V(T_2)$.
Index a set $I \subseteq V(T_1) \times V(T_2)$. Given a query $(u, w)$, search for all $(x, y) \in I$ where $x \in \text{descendants}(u)$ and $y \in \text{descendants}(w)$.

The ESA00 index

- supports searching in $O(\log \log n + \text{occ})$ time;
- takes $O(|I| \log |V|)$ space, where $V = V(T_1) \cup V(T_2)$.

A pair of leaves is in $I$ if they represent $S[a..n]$ and $S[1..b]$ respectively where $a = b + 1$.

Each $(a, b)$ has $O(\log^{k_1} n)$ and $O(\log^{k_2} n)$ leaves respectively. $|I| = O(\frac{n}{\beta} \log^{k_1} n \log^{k_2} n) = O(\frac{n}{\beta} \log^k n)$. 

H.L. Chan, T.W. Lam, W.K. Sung, S.L. Tam, S.S. Wong

A Linear Size Index for Approximate String Matching
Searching Complexity

- By setting $\beta = \Theta(\log^{k+1} n)$, we obtained an $O(n)$ space index.

- $|P| \geq \beta$:
  - It takes $O(\log^k n \log \log n + occ')$ time to search STOC04 and ESA00 indexes.
  - It takes $O(m)$ time to preprocess $P$ to enable the above searching.
  - Searching time is $O(m + \beta \log^k n \log \log n + occ) = O(m + \log^{2k+1} n \log \log n + occ)$.

- $|P| < \beta$:
  - Searching time is $O((cm)^k \log \log n + occ) = O((c \log n)^{k(k+1)} \log \log n + occ)$. 
If we pick $\beta = \Theta(\log^{k+2} n)$ and use a compressed suffix tree to replace the suffix tree in [STOC04], we obtain an index using $O(n)$ bits.

Space-time tradeoff is possible by choosing a smaller $\beta$ and using the brute-force searching technique for some of the $k$ errors.

Extension to edit distance is possible but each $k$-error match maybe reported multiple times.
Future Directions

- Make a better trade-off — the gap in time complexity between long and short patterns
- Improve “output complexity” of [STOC04]
- Improve space complexity of [ESA00]
- Improve searching complexity for short patterns
- Our recent ESA 2006 paper uses $O(n)$ space to allow $k$-error searching in $O((cm)^{k-1} \log n \log \log n + occ)$ time.

  Using this result, searching time is reduced to $O(m + \log^k n \log \log n + occ)$. 

H.L.Chan, T.W.Lam, W.K.Sung, S.L.Tam, S.S.Wong
A Linear Size Index for Approximate String Matching