Speeding up compressed matching with SBNDM2

Kerttu Pollari-Malmi  Jussi Rautio  Jorma Tarhio

Aalto University
Finland
Introduction

- Compressed matching problem: string matching in a compressed text without decompression
- Aim: faster searching
Introduction

- Several efficient methods are based on byte pair encoding (BPE).
- We achieved faster searching with encoding of different type.
- Earlier, we have presented a search algorithm based on Boyer-Moore-Horspool.
- Now, we present a search algorithm based on SBNDM2.
Byte pair encoding (BPE)

- **BPE** (Gage 94) replaces recursively the most common byte pair by an unused character code.
  
  \[
  abcabc\ldots \Rightarrow d=ab|dc\ldots \Rightarrow e=dc,d=ab|ee\ldots
  \]

- Manber's BPE: bytes are classified either a **start** or **end** byte of a pair to ensure locally unambiguous decoding.

- BPE achieves moderate compression ratios on text: 45-75% (best methods achieve 20-30%)

- **BPX** (Maruyama et al. 08) is a modification of BPE with better compression ratio.
Our encoding method

- Codeword for a character is a variable-length sequence of $k$-bit base symbols.

  a  b  r  a  c  a  d  a  b  r  a
  00 01 10 00 11 00 00 11 01 00 01 10 00

- Related to Huffman encoding
- de Moura et al. (00) use 8-bit symbols to encode words

- The coding method is called **Stopper Encoding** and denoted by $SE_k$ for $k$-bit base symbols.
Our encoding method (cont.)

- Encoding and decoding are very fast.

- Search algorithm:
  - variation of SBNDM2 (new)
  - variation of Boyer-More-Horspool (presented earlier)

- Comparable compression ratio with fast BPE but searching is faster
Semi-static coding scheme

- Codewords are based on frequencies of characters in the text.
- Two passes
  1. The frequencies of characters are gathered
  2. Actual coding
- The code table is a part of the compressed file.
Stoppers and continuers

Because the length of a codeword varies and SBNDM2 jumps forward, we need a mechanism to recognize where is a border of subsequent codewords.
Two **classes** of base symbols:
- The last base symbol of a codeword is a *stopper*.
- Other base symbols are *continuers*.

Example: $u_1 u_2 u_3 u_4 u_5$

continuers  stopper
Stoppers and continuers (example)

codewords: 00, 01 00, 01 01 00

text: ...00 01 00...
Number of stoppers

- The optimal number depends on the number of different characters and their frequencies.
- Computation is straightforward.
- Example: 14 is optimal for the English Bible with 16 (4-bit) base symbols.
Searching

- The pattern is **encoded in the same way** as the text.
- Search is based on **bytes**.
- An occurrence of the pattern does not necessarily start at the beginning of a byte. To avoid bit manipulation, several patterns are searched at the same time.
SBNDM
Simple Backward Nondeterministic DAWG Matching

- SBNDM is a simplification of BNDM. Both are bit-parallel algorithms, which recognize factors of the pattern.

- Text $T = t_1...t_n$, pattern $P = p_1...p_m$.

- At an alignment of $P$: $t_i...t_{i+m-1}$, scan $T$ from right to left until the suffix $t_k...t_{i+m-1}$ is not a factor of $P$ or an occurrence of $P$ is found ($k = i$).

- Next alignment starts at $t_{k+1}$. 
SBNDM, example

P = banana, T = antanabadbanana...

alignment: antanabadbanana
  a
  na
  ana
  tana
not a factor: d
next alignment: antanabadbanana
not a factor: d
next alignment: antanabadbanana
SBNDM can be made faster by reading two text characters instead of one before checking anything.

Occurrence vectors are precomputed for all 2-grams.

If the encoded pattern is \texttt{618e0} (in hexadecimal), we search for both \texttt{61-8e} and \texttt{18-e0} simultaneously by searching the pattern \texttt{61-8e-18-e0}.
Code splitting

- The high bits of base symbols are concatenated to one file and the low bits to another file:
  \[
  1110 \ 0110 \ 0011 = 110100 \ 101011
  \]

- Motivation: dense accessing is faster than sparse accessing
Code splitting

- Low bits of the pattern are searched in the low bits of the text

- For matches found in low bits
  - verify with high bits
  - check that the preceding base symbol is a stopper
Combining code splitting with stopper encoding

- $SE_{k,h}$: stopper encoding with $k$-bit base symbols and with division to $h$ high bits and $k-h$ low bits
- $SE_k$: stopper encoding without code splitting
- $SE_{8,h}$: plain code splitting without compression

We consider here two versions: $SE_4$, $SE_{8,4}$
Test data

- Part of the fruitfly DNA (5 MB)
- English Bible (extended to 5 MB)
- Finnish Bible (extended to 5 MB)
## Compression ratios

<table>
<thead>
<tr>
<th></th>
<th>English Bible</th>
<th>Finnish Bible</th>
<th>DNA</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPX</td>
<td>28.0 %</td>
<td>32.6 %</td>
<td>27.8 %</td>
</tr>
<tr>
<td>BPE</td>
<td>51.0 %</td>
<td>52.1 %</td>
<td>34.0 %</td>
</tr>
<tr>
<td>SE$_4$</td>
<td>58.8 %</td>
<td>58.2 %</td>
<td>50.0 %</td>
</tr>
</tbody>
</table>

- Compression ratio = compressed size / original size
Tested search algorithms

- TBM: Tuned Boyer-Moore for uncompressed texts
- SBNDM2: for uncompressed texts
- BM-BPE: texts compressed by BPE by Shibata et al. (00)
- KMP-BPX: texts compr. by rec. pairing by Maruyama et al. (08)
- SBNDM2-SE$_4$: SBNDM2 for SE$_4$ encoded texts
- SBNDM2-SE$_{8,4}$: SBNDM2 for SE$_{8,4}$ encoded texts (code splitting, no compression)
- BM-SE$_4$: Boyer-Moore for SE$_4$ encoded texts
- BM-SE$_{8,4}$: Boyer-Moore for SE$_{8,4}$ encoded texts (cs, nc)
Results: DNA

The graph shows the search time (ms) as a function of pattern length for different algorithms. The algorithms compared include TBM, SBNDM2, BM-BPE, KMP-BPX, and SBNDM2-SE_4.
Results: English text

![Graph showing search time vs. pattern length for different algorithms: TBM, SBNM2, BM-BPE, KMP-BPX, SBNM2-SE_4.](image)
Concluding remarks

- Practical solutions for the compressed matching problem

- SBNDM2-SE$_4$ is faster than other tested methods of compressed matching in English and DNA texts for pattern lengths $> 5$.

- SBNDM2-SE$_4$ is faster than SBNDM2 for pattern lengths $\geq 9$ in English text, but slower for shorter patterns.

- SE$_4$ has similar compression ratio to the fast BPE.