Lodz University of Technology
Institute of Applied Computer Science

Compression and bit-parallelism techniques in selected string matching problems

PhD Thesis

ABSTRACT

Robert Susik

Supervisor: dr hab. Szymon Grabowski, prof. PŁ

February 27, 2018

Abstract

The dissertation focuses on stringology which is a field of algorithmics. The stringology is a term that refers to all problems related to text processing (such as pattern searching, transforming, compression and so on). Searching in the text is an activity which almost all of us do on a daily basis and algorithms for this purpose are the main subject of the research presented in this dissertation. All of the text searching tools use algorithms that can be categorized into two groups:

- Online algorithms – the search is performed without the knowledge about the datasets given prior to the search process.
- Offline algorithms – have the knowledge about the datasets before the searching starts.

The search process can be performed on plain or compressed text (the representation of the text that uses less memory than original one). In the second case the naive approach is to decompress the text and then run a search algorithm on decompressed text, which significantly slows down the process. There are more appropriate algorithms for this case which are specialized in searching in the compressed data (those algorithms usually are designed for a specific data compression method). The mentioned algorithms not only can have similar performance as the ones searching in plain text but in some cases can even be faster. A number of text processing problems can be listed, such as exact pattern matching, exact multiple pattern matching, approximate pattern matching, etc. This dissertation introduces algorithms for the following stringology problems:

- exact single pattern matching,
- exact multiple pattern matching,
- searching in the compressed text,
- approximate pattern matching,
- circular pattern matching,
- text indexing.

The proposed solutions share a common trait, that is, the bit-parallel technique. Most of the designed algorithms are online, but the indexing solution (namely, the semi-index BFSI, a lightweight data structure based on Bloom filter) is a little distinct as it combines the advantages of online and offline solutions being much faster than online ones and builds the index faster than standard offline solutions.

All of the mentioned solutions prove the following theses:

1. Bit-parallel techniques allow to speed-up multiple pattern matching, approximate pattern matching, searching in the compressed text and indexing.
2. Appropriately chosen text compression algorithms allow for faster pattern matching and constructing a text index with favorable practical characteristics.

All algorithms were extensively tested and compared with competing solutions. All source codes are available online and most of the results were published in the following articles:

- **R. Susik**, Sz. Grabowski, and S. Deorowicz. Fast and Simple Circular Pattern Matching. In *Man-Machine Interactions 3*, pages 537–544. Springer, 2014
- **R. Susik**, Sz. Grabowski, and K. Fredriksson. Multiple Pattern Matching Revisited. In *Proceedings of the Prague Stringology Conference*, pages 59–70, 2014
- **R. Susik** and Sz. Grabowski. Engineering the Counting Filter for String Matching Algorithms. In *Algorithms, Networking and Sensing for Data Processing, Mobile Computing and Applications*, pages 125–140. Łódź University of Technology Press, 2016
- Sz. Grabowski, **R. Susik**, and M. Raniszewski. A Bloom Filter based Semi-Index on q-grams. *Software: Practice and Experience*, 47(6):799–811, 2017
- **R. Susik**. Applying a q-gram based Multiple String Matching Algorithm for Approximate Matching. *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska*, 7(3):47–50, 2017

The source codes can be found at the URLs given below:

- `https://github.com/rsusik/mag`
- `https://github.com/rsusik/cf2`
- `https://github.com/rsusik/bfsi`
- `https://github.com/rsusik/magc`
- `https://github.com/rsusik/fscpm`
- `https://github.com/rsusik/maga`
- `https://github.com/rsusik/magetdc`

In order to prove the first thesis the multiple algorithms were designed, extensively tested and compared against competing solutions.

The bit-parallel technique was successfully applied in stringology especially for multiple pattern matching, approximate pattern matching and indexing. The algorithm for multiple pattern matching (MAG) allows to search in both compressed (byte codes) and plain text, achieving better results than its competitors in most analyzed cases. A number of alternative alphabet mapping methods (each having a different characteristic) have been proposed and tested. It was proven that the solution is sublinear in the average case. The application of byte codes allows for text compression, reduces the memory requirement and speeds-up the searching process for long patterns (in comparison to searching in plain text).

The MAG algorithm was successfully applied to circular pattern matching achieving speed of 7–8 GB/s on a standard PC in a serial implementation, which was very competitive to the results reported at that time in the literature. The solution was extended and compared with another solution that was published later and the results were also competitive.

It was also proven that MAG algorithm can be successfully applied to approximate pattern matching problem with k errors/mismatches. This approach allowed to achieve up to 6-fold speed-up (for large number of patterns) in comparison to competitive algorithms.

Additionally, a bit-parallel technique was used for an efficient implementation of the so-called Counting Filter, a filtering algorithm useful in approximate pattern matching. A few variants of this algorithm (CF2) were proposed that use Streaming SIMD Extensions (SSE), q-grams, and skipping technique (avoids repeating steps for positions

where it is certain that the pattern does not occur). The performed tests showed that the algorithm achieves the 4-fold speed-up comparing to a standard implementation known from the literature.

Another and different solution that uses bit-parallel technique is a semi-index (BFSI). The lightweight indexing structure combined high performance with implementation simplicity. The solution uses bit-parallelism to reduce the number of cache misses. The filtering method (using Bloom filters) of text blocks allows to reduce the size of text that needs to be searched in by avoiding searching in the blocks where the pattern do not exists. Experimental results show that the solution is three orders of magnitude faster than a preceding work, by Claude et al. (2012). Additionally, the solution can easily be parallelized and executed in multiple threads, which is a huge advantage taking into account current technology trends.

The main shared trait of all presented algorithms is bit-parallelism, which is mainly constrained by the machine word size. However, the number of bits in machine word is growing (which currently equals 64) what is causing more attention to this technique. Additionally, the modern CPU are equipped with an extension set of instructions that supports 128-bit words (SSE) or even 512-bit ones (for AVX).

In summary, a multiple pattern matching algorithm using bit-parallelism was developed and tested, and also applied to other stringology problems, such as approximate pattern matching and circular pattern matching. In addition to this, searching in the compressed text using byte codes and a semi-index were successfully implemented achieving very good results. All the proposed solutions and experimental results confirm the validity of the posed theses.