Finding Short Synchronizing Words for Prefix Codes

Andrew Ryzhikov

LIGM, Université Paris-Est, Marne-la-Vallée, France

This is joint work with Marek Szykuła.

Introduction Prefix codes are a simple and powerful class of variable-length codes that are widely used in information compression and transmission. A famous example of prefix codes are Huffman's codes [Huf52]. In general, variable length codes are not resistant to errors, since one deletion, insertion or change of a symbol can desynchronize the decoder causing incorrect decoding of the whole remaining part of the message. However, in a large class of codes called synchronizing codes resynchronization of the decoder is possible in such situations. It is known that almost all maximal finite prefix codes are synchronizing [FJTZ03]. Synchronization of finite prefix codes has been investigated a lot [Ba16, Bis08, BP09, CDSGV92, Sch64, Sch67], see also the book [BPR10] and references therein. For efficiency reasons it is important to use as short words resynchronizing the decoder as possible to decrease synchronization time. However, despite the interest to synchronizing prefix codes, the computational complexity of finding short synchronizing words for them has not been studied so far. We provide a systematic investigation of this topic.

Each recognizable (by a finite automaton) maximal prefix code can be represented by an automaton decoding the star of this code. For a finite code, this automaton can be exponentially smaller than the representation of the code by listing all its words (consider, for example, the code of all words of some fixed length). This can of course happen even if the code is synchronizing. In different applications the first or the second way of representing the code can be useful. In some cases large codes having a short description may be represented by a minimized decoder, while in other applications the code can be described by simply providing the list of all codewords. We study the complexity of problems for both arbitrary and literal decoders of finite prefix codes.

Huffman decoders There is a strong relation between partial automata and prefix codes [BPR10]. A set X of words is called a *prefix code* if no word in X is a prefix of another word. The class of recognizable (by an automaton) prefix codes can be described as follows. Take a strongly connected partial automaton A and pick a state r in it. Then the set of all *first return words* of r (that is, words mapping r to itself such that each non-empty prefix does not map r to itself) is a recognizable prefix code. Moreover, each recognizable prefix code can be obtained this way. A prefix code is called *maximal* if it is not a subset of another prefix code. The class of maximal recognizable prefix codes corresponds to the class of complete automata. If a state r can be picked in an automaton in such a way that the set of all first return words is a finite prefix code, we call the automaton a *partial Huffman decoder*. If such automaton is complete (and thus the finite prefix code is maximal), we call it simply a *Huffman decoder*.

For the mentioned classes of decoders we obtain the following inapproximability result.

Theorem 1. *Unless $P = NP$, the problem of finding a shortest synchronizing words cannot be approximated in polynomial time within a factor of*
(i) $n^{1-\varepsilon}$ for any $\varepsilon > 0$ for n-state binary strongly connected automata;
(ii) $c \log n$ for some $c > 0$ for binary n-states Huffman decoders;
(iii) $n^{\frac{1}{2}-\varepsilon}$ for any $\varepsilon > 0$ for binary n-state partial Huffman decoders.

We remark that for strongly connected automata the bound is optimal because there exists a $O(n)$-approximation algorithm [GH11]. We also conjecture that for binary n-states Huffman decoders the $c \log n$-inapproximability bound is optimal.

Literal decoders The literal automaton of a maximal finite prefix code X is defined as follows. The set of its states is the set of proper prefixes of the words in X and the transitions are naturally defined to concatenate letters to the prefixes (or to map to the empty prefix if the resulting word is in X). The number of states of a literal automaton is polynomially equivalent to the total length of all words in the corresponding finite prefix code. Thus, it is a natural model for the problems where the code is provided by simply enumerating all its codewords.

Theorem 2. *For literal n-state decoders there exist*
(i) a polynomial time $O(\log n)$-approximation algorithm
and
(ii) for any $\varepsilon > 0$ a quasi-polynomial time $(1 + \varepsilon)$-approximation algorithm
for the problem of finding a shortest synchronizing word.

We conjecture that there exists a polynomial time exact algorithm for this problem.

Mortal words A word is called *mortal* for a partial automaton A if its mapping is undefined for all the states of A. The techniques that we develop can be easily adapted to get the same inapproximability for the problem of finding a shortest mortal word. This problem is connected for instance to the famous Restivo's conjecture [Res81].

Theorem 3. *Unless $P = NP$, the problem of finding a shortest mortal word cannot be approximated in polynomial time within a factor of*
(i) $n^{1-\varepsilon}$ for any $\varepsilon > 0$ for n-state binary strongly connected partial automata;
(ii) $c \log n$ for some $c > 0$ for n-state binary partial Huffman decoders.

We also propose a simple polynomial time $O(\log n)$-approximation algorithm for this problem in partial literal decoders.

References

[Ba16] Mikhail V. Berlinkov and Marek Szykuła. Algebraic synchronization criterion and computing reset words. *Information Sciences*, 369:718 – 730, 2016.

[Bis08] Marek Biskup. *Error Resilience in Compressed Data – Selected Topics*. PhD thesis, Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, 2008.

[BP09] Marek Tomasz Biskup and Wojciech Plandowski. Shortest synchronizing strings for huffman codes. *Theoretical Computer Science*, 410(38):3925 – 3941, 2009.

[BPR10] Jean Berstel, Dominique Perrin, and Christophe Reutenauer. *Codes and Automata*. Encyclopedia of Mathematics and its Applications 129. Cambridge University Press, 2010.

[CDSGV92] Renato M. Capocelli, A. A. De Santis, Luisa Gargano, and Ugo Vaccaro. On the construction of statistically synchronizable codes. *IEEE Transactions on Information Theory*, 38(2):407–414, 1992.

[FJTZ03] Christopher F Freiling, Douglas S Jungreis, François Théberge, and Kenneth Zeger. Almost all complete binary prefix codes have a self-synchronizing string. *IEEE Transactions on Information Theory*, 49(9):2219–2225, 2003.

[GH11] Michael Gerbush and Brent Heeringa. *Approximating Minimum Reset Sequences*, pages 154–162. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[Huf52] David A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[Res81] Antonio Restivo. Some remarks on complete subsets of a free monoid. *Quaderni de La ricerca scientifica, CNR Roma*, 109:19–25, 1981.

[Sch64] Marcel-Paul Schützenberger. On the synchronizing properties of certain prefix codes. *Information and Control*, 7(1):23 – 36, 1964.

[Sch67] Marcel-Paul Schützenberger. On synchronizing prefix codes. *Information and Control*, 11(4):396 – 401, 1967.