Processing massive string data and beyond: algorithms and conditional lower bounds

Paweł Gawrychowski

This project concerns the so-called algorithms on strings, where a string is just a linear sequence of characters from some alphabet. Algorithms on strings are usually concerned with one of the three basic questions: pattern matching, indexing, and calculating similarity between two strings. In pattern matching, the goal is to find an occurrence of one string in another. In indexing, the goal is to build, for a given string, a data structure that later allows to find occurrences of any given string efficiently. Finally, in calculating similarity the goal is to check how similar are two different strings. All of those questions come in multiple shapes and sizes. For example, we might allow for some noise in the input, and choose an appropriate definition of occurring and being similar.

The motivation for this area comes largely from bioinformatics, where a string could be a nucleotide sequence consisting of characters from $\{A, C, G, T\}$, or a chromosome consisting of genes identified by numbers (with every gene possibly having a direction). While algorithms on strings are widely used in bioinformatics, the former is an established area in its own right, with two conferences: CPM (Annual Symposium on Combinatorial Pattern Matching) and SPIRE (Symposium on String Processing and Information Retrieval) taking place every year since the early 90s, and multiple papers appearing annually in major conferences on algorithms such as SODA (ACM-SIAM Symposium on Discrete Algorithms), ICALP (International Colloquium on Automata, Languages and Programming), ESA (European Symposium on Algorithms), and some papers appearing in the top conferences on theoretical computer science: STOC (Symposium on the Theory of Computing) and FOCS (IEEE Annual Symposium on Foundations of Computer Science).

This proposal concerns theoretical aspects of algorithms on strings, and aims to further our understanding of some major open questions in this area motivated by possible applications in bioinformatics. I hope to make progress on those questions by combining insights from combinatorics on words, algorithms and data structures in the Word RAM model, and finally the field of fine-grained complexity. The Word RAM model provides an elegant formalism for designing algorithms and (in some cases) interesting lower bounds, and on the other hand, captures the power of modern computers. The emerging field of fine-grained complexity tries to classify polynomial problems according to their complexities, providing reductions that connect them to each other and provide an explanation for why, in some cases, no efficient algorithm has been found so far.

I would like to focus on five directions: the first three are each guided by a major open question, while the fourth and the fifth are more open-ended. The first direction is pattern matching in compressed strings, a topic that I am very familiar with, having extensively worked on it in my PhD thesis. The second direction is succinct and encoding data structures with applications to text indexing, a topic that I am quite familiar with, having published some papers on the related encoding data structures. The third direction is complexity of streaming pattern matching, a topic on which I have published a few papers, and spent some time thinking about the most fundamental question in the area. The fourth direction are different notions of string similarity. A few years ago I have published a paper on edit distance with moves, very recently I have been working on sorting oriented permutations by reversals, and I would like to explore possible follow-ups of both problems. Finally, the fifth direction is pattern matching in generalisations of strings. I published a few papers on problems equivalent to counting cycles in sparse graphs, and a paper on pattern matching in elastic-degenerate strings, and would like to find more generalisations.

For all above research directions, my goal is to design new efficient algorithms, and (especially for the last direction) find some explanation for why such algorithms are unlikely to exist for some problems. The results would be published in leading conferences in the field.