PI: **dr hab. Jakub Radoszewski**
Project title: „Quest for Optimality in Variants of Text Indexing and Related Problems"

# Project Description for the General Public

This research project is devoted to searching for efficient solutions to text indexing problems. The basic scenario in text indexing is as follows. We are given a large collection of data that is stored digitally in a text format. Here by a text we mean a sequence of characters or numbers rather than a text written in a natural language; this allows to apply our methods to any input data in a text format, like a sequence of data reported by a certain measuring device or a DNA sequence. Our goal is to create an index, that is, a data structure that allows, in the simplest case, to answer queries if a given pattern has an occurrence being a fragment of one of the texts in the collection. For us it is of utmost importance that the index uses little memory in addition to what is required to store the data collection, that it can be built in reasonable time and that the time of answering pattern matching queries is minimized.

Many efficient solutions were developed for the above classic version of the problem. In particular, there are programming libraries implementing indexes that are very fast and small at the same time, and computer science students are usually taught basic data structures for text indexing. In practice, however, the classic version of the problem is not enough, for several reasons: the data may contain errors or corruptions; sometimes only a subset of the occurrences that satisfy certain additional requirements are sought; and one may be interested in fragments that do not match the specified pattern exactly, but are just similar to the pattern. Existing indexes serve these purposes with varying success, but in a majority of cases either they are not optimal, or it is not known if they can be improved. The goal of the project is refining these indexes. We will also consider algorithmic problems on texts that are related to text indexing and that allow to look at this problem from a broader perspective.

One of the problems that will be considered in this project is constructing a so-called approximate index. Such an index allows to efficiently check if a given collection contains a fragment that differs from the query pattern at up to $k$ positions, for a specified parameter $k$. For example, the pattern "positrons" occurs in the previous sentence with $k = 1$ mismatch (as the word "positions"). The size and running time of existing approximate indexes grow exponentially with regard to the parameter $k$. In 2019 it was shown that for a sufficiently large parameter $k$ and under certain additional assumptions, this dependency cannot be avoided. However, for small $k$ this problem is so important from a practical perspective that this kind of an answer is not enough. In particular, there exist more efficient approximate indexes in the basic case that $k = 1$. In 2021 the PI worked in a team on a problem of computing a longest common substring with at most $k$ mismatches of two strings, for a small value of parameter $k$. The classic variant of this problem (for $k = 0$) has the same solution as the problem of constructing an index. In turn, in the approximate case we proposed a solution to the longest common substring problem that works (a little) faster than what one would expect by applying an approximate index. This result begs the question if, for a small $k$, a more efficient approximate index can be constructed and how significant improvement in the two considered problems can be reached. An attempt to answer this question is one of the particular goals of the project.

The outcome of the project will be publications describing the best of the indexes proposed by us. Additionally we might be able to prove that better indexes (most likely) cannot be constructed. When assessing an index we consider the so-called worst case complexity, that is, we would like the index to have certain parameters regardless of the particular text that it is constructed for. In some cases we might write programs implementing our indexes, but in principle it will be more important to verify if a given theoretical barrier can be broken.