Indexing extra-large data for long patterns using small space

**Supervisor:** Solon Pissis (CWI) and Michelle Sweering (CWI)

**Keywords:** Algorithms, data structures, string algorithms, indexing, pattern matching

Text indexing is a classic problem in computer science. It consists in constructing a compact index over a given text for answering subsequent pattern matching queries. From early days, and in contrast to the traditional data structure literature, where the focus is on space-query time trade-offs, the main focus in text indexing has been on the construction time. This focus can be explained by the myriad applications of text indexing in bioinformatics and elsewhere. That was until the breakthrough result of Farach [FOCS 1997], who showed that suffix trees can be constructed in linear time. After that, more and more attention had been given to reducing the space of the index via compression techniques [Grossi and Vitter, SIAM J. Comput. 2005; Ferragina and Manzini, J. ACM 2005]. Nowadays, as the data volume grows rapidly, construction space is as well becoming crucial [Belazzougui et al., ACM Trans. Algorithms 2020]. This completes the four absolute measures anyone should pay attention to when designing or implementing a text index. Unfortunately, however, most (if not all) widely-used indexes are not optimized for all four measures simultaneously, as it is difficult to have the best of all four worlds. A new approach to text indexing assumes a lower bound on the length of the pattern matching queries and exploits it by first sampling the text with locally-consistent anchors (i.e., carefully selecting some positions on the text), and then indexing only the suffixes starting at these anchors (positions). Loukides and Pissis [ESA 2021] have recently shown that this paradigm is a very effective towards meeting the best of all four worlds. In this project, we plan to investigate trade-offs between construction time and construction space. We will re-visit the sparse suffix sorting problem, which lies at the heart of indexing with locally-consistent anchors, and try to improve it for this special regime. This re-visit will hopefully result in a new index construction, which meets the best of all four worlds. We are looking for someone with a background in algorithms and strong programming skills (e.g. C++).