First meeting (06-09-2022): The string indexing with compressed pattern problem

Mathias Søndergaard, s174426

September 5, 2022

What has been done initially

Project initiation. The readings I have done include reviewing old material on the LZ77 compression scheme and Suffix trees, as well as constructing these in $O(n)$ time (with constant sized alphabets for suffix trees). For the LZ77 scheme the KKP3 algorithm shown in [2] was reviewed (but not fully understood). Also I have begun reading and understanding [3], now reaching section 4 (so I have just covered the "simple" Phrase trie section). During this time, I also reviewed material on NCA and LCP. This included looking into on how to construct these in $O(n)$ time, however it took me some time to get hold of [1], alas the material has not been fully reviewed.

Implementation / Programming wise, the LZ77 implementation running in $O(n)$ time and space found at https://www.cs.helsinki.fi/group/pads/lz77.html has been tested after updating the code to work on my system. Fortunately, this code includes construction of suffix arrays in $O(n)$ time and space. I have implemented suffix trees running in $O(m)$ time and $O(n)$ space. The phrase trie with $O(n^2)$ space and $O(z + log(n) + occ)$ time has also been implemented. For now I chose a naive construction algorithm for both trees resulting in $O(n^2)$ prepossessing time.

Plans for the next weeks

For the next meeting the goal is to have implemented the Space efficient Phrase Trie found in section 4 of [3] and if time allows, also look into replacing the naive prepossessing algorithms mentioned above with optimal prepossessing algorithms. However this last part requires me to also implement NCA/LCP in an optimal manner, which requires a bit more work.

Questions for this meeting

For now, I have no theory related questions. I would like to discuss my strategy moving forward. The implementation aspect of this project has peaked my interest greatly. But time may become an issue, which is why I have initially chosen non-optimal construction methods for the data structures. This allows me to cover more ground in the beginning, and my plan is to return to the prepossessing later. Is this the correct way to go about things? Or should I implement the data structure in its full power before moving on? Obviously a report also needs to be written - is it required of me to ”invent” new theory on this subject, or is reviewing the theory I use in a concise manner enough? Thus letting the implementation be the novel part of this thesis.

References

[1] D. Harel et al. “Fast Algorithms for finding nearest common ancestors”. In: (1984).

[2] D. KEMPA et al. “Lazy Lempel-Ziv Factorization Algorithms”. In: (2016).

[3] Philip Bille, Inge Li Gørtz, and Teresa Anna Steiner. “String Indexing with Compressed Patterns”. In: (2020).