Approximate string matching aka fuzzy string searching

Ladislav Šulák (laco.sulak@gmail.com)

Krisztian Benko (kristianbnk@gmail.com)

Brno University of Technology, Faculty of Information Technology

String matching I

- **Method for finding string that match a given pattern approximately, rather than exactly.**
- **Closeness of a match is a number. It is measured by edit distance – number of operations needed to change one string into another.**
- **It is heuristic – more effective than naive approach.**

String matching II

- **It can benefit from Levenstein distance algorithm.**
- **It can be implemented as Bitap algorithm (Baeza-Yates–Gonnet algorithm) which uses bitwise operators and it is therefore extremely fast. It is also used in Unix utility Grep.**
- **Possible real-world usages: spell checker or spam filtering.**

What TODO?

- Prepare data = pairs of (patterns, strings)
- Implement console app with the usage of fuzzy approach in Python for appoximate string matching
- Evaluate results (similarity score, various scenarios in data)
- Compare results with naive approach
- Technical report and documentation

Discussions

Questions?

References

- https://github.com/seatgeek/fuzzywuzzy
- https://github.com/sajari/fuzzy
- https://towardsdatascience.com/symspell-vs-bk-tree-100x-faster-fuzzy-string-search-spell-checking-c4f10d80a078
- http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/
- https://en.wikipedia.org/wiki/Approximate_string_matching
- https://en.wikipedia.org/wiki/Levenshtein_distance
- https://en.wikipedia.org/wiki/Bitap_algorithm