Scaling Up Similarity Joins Using A Cost-Based Distributed-Parallel Framework

Fabian Fier and Johann-Christoph Freytag

Key Contributions

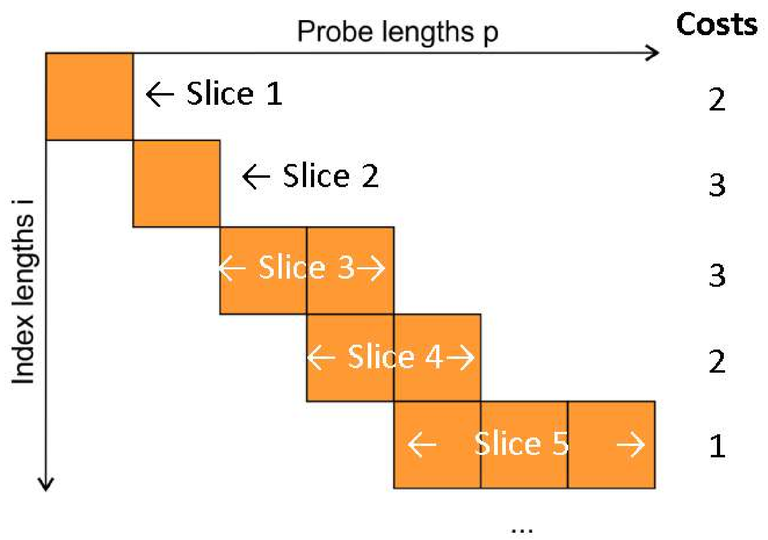
- A cost-based heuristic and further mechanisms to break down the Set Similarity Join (SSJ) into independent slices,
- a RAM usage estimation to avoid swapping,
- high scalability to hundreds of GB of input data.

Set Similarity Join

- Input. Given a collection of sets R formed over the universe U of tokens (set elements), and a similarity function between two sets, sim : P(U) × P(U) -> [0, 1].
- Output. The SSJ computes all pairs of sets (s, r) $\in$ R ×R whose similarity exceeds a user-defined threshold t, 0 < t < 1, i.e., all pairs (s, r) with sim(s, r) > t.

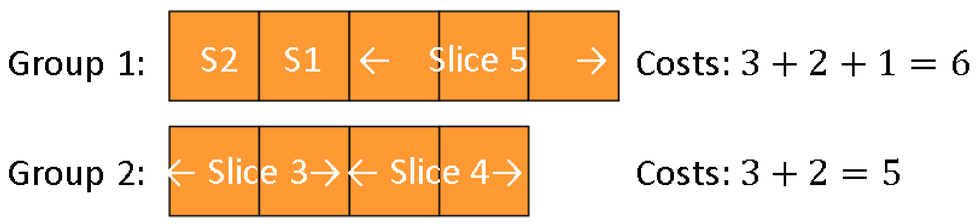
Cost-based heuristic

- Avoid recomputation: only matching lengths,
- divide join computation into independent slices
- estimate compute costs per slice using length statistics,
- assign slices to n groups such that costs are distributed evenly (greedy heuristic).

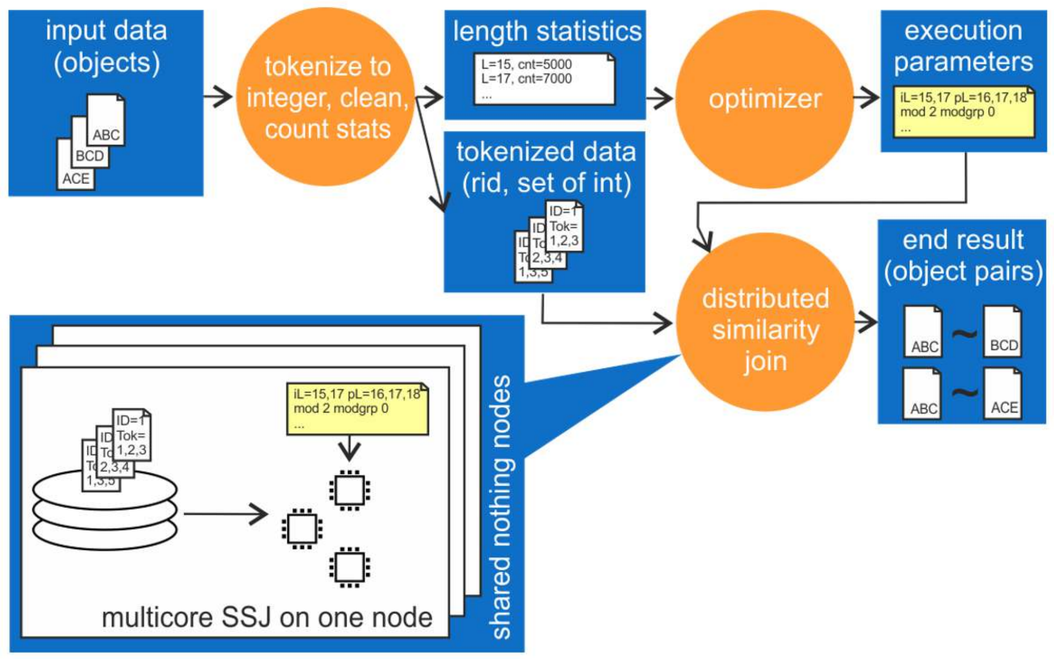


Example join matrix of matching lengths.

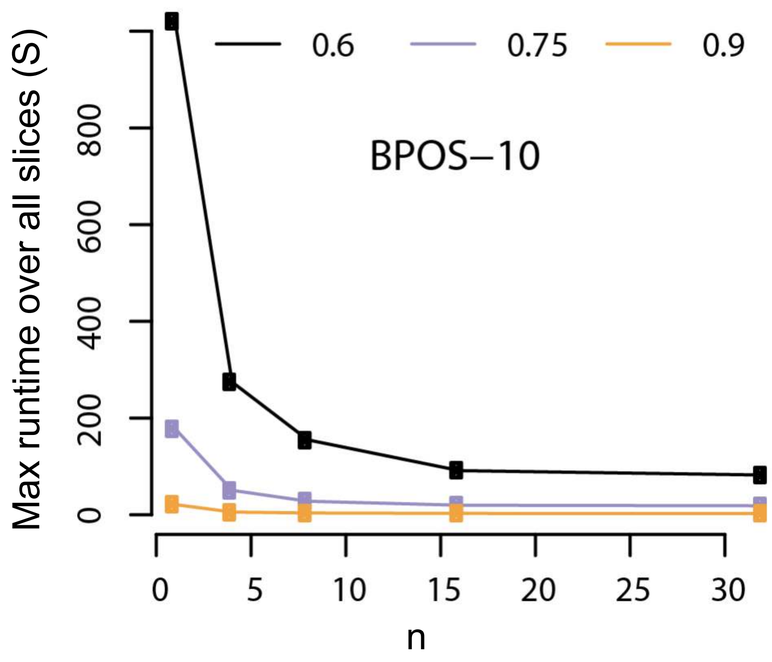
Example for n=2:



Distributed Approach: Dataflow



Experiments and discussion



Example runtime effect of cost-based heuristic.

- Limitations (by design): length and candidate skew.
- We provide further means to partition the slices, to prevent swapping, and to find suitable parameter values.

Outlook

- Additionally consider GPU,
- machine learning for candidate estimation,
- adaption to Big Data systems.

References, Code, Contact

- Extended paper: Fier, F., Freytag, J.C.: Scaling up set similarity joins using a cost-based distributed-parallel framework [extended paper] (2021). https://doi.org/10.18452/23209

- Code: https://github.com/fabiyon/dist-ssj-sisap
- Contact: fier@informatik.hu-berlin.de