SKETCHES ON SINGLE BOARD COMPUTERS

Student(s)

Ali Osman Berk Şapcı
Egemen Ertuğrul
Hakan Ogan Alpar

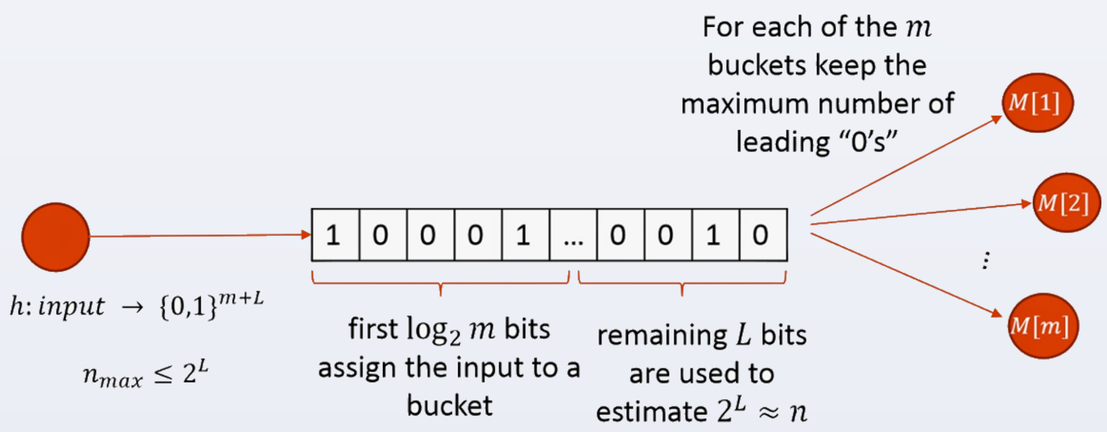
Faculty Member(s)

Kamer Kaya

Sabancı Üniversitesi

PURE
PROGRAM FOR UNDERGRADUATE RESEARCH

ABSTRACT



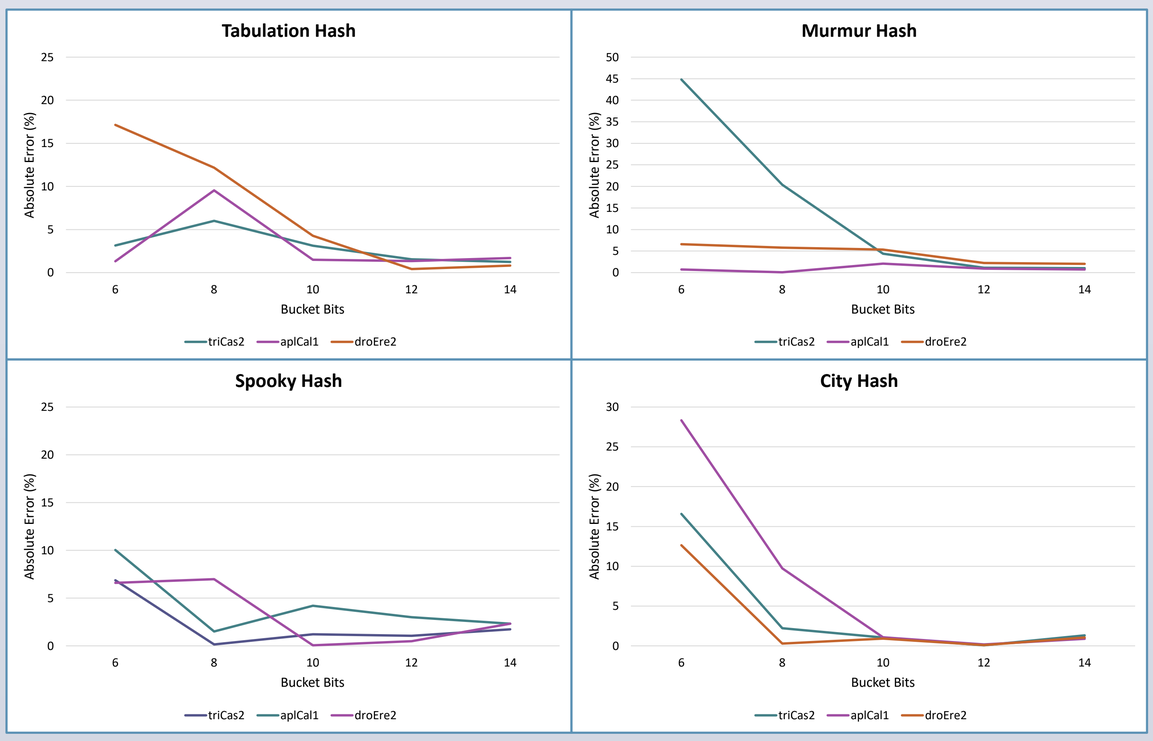
Data sketches are probabilistic data structures with mathematically proven error bounds that store information about the datasets using small amount of space using hash functions. They can approximate some predefined questions (i.e. cardinality or frequency estimation) about big data with high accuracy.

We chose a data sketch called HyperLogLog and tested it with many different configurations in order to find results that show least errors possible when estimating unique elements in large genomic datasets.

OBJECTIVES

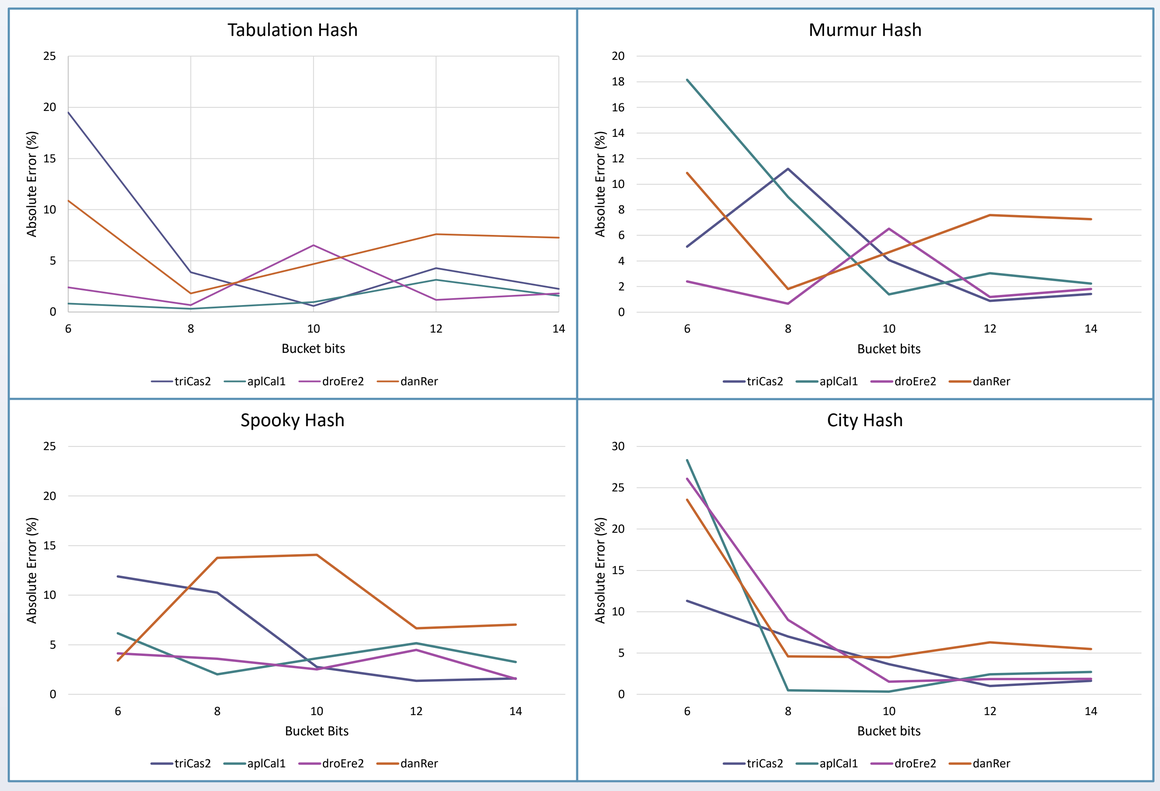
- Implementing and learning more about data sketches; especially HyperLogLog
- Comparing hash functions to find most suitable ones for different queries on genomic data
- Investigating effects of k-mer size, data size, register size and hash functions on the estimation accuracy of data sketches

K-MER 64 RESULTS



We focus on HyperLogLog and its behavior with different hash functions when we review our results. Murmur Hash, Spooky Hash, City Hash and Tabulation Hash functions are subject to our experiments. Comparison of the accuracy of hash functions on the datasets were done with a tool called DSK that gives exact cardinality result. Other parameters subject to the experiments were the different datasets, different K-mer sizes such as 32, 40, 48, 56, 64 and different bucket bits (a parameter of HyperLogLog) such as 6, 8, 10, 12, 14. Results of these experiments were used to try and find a correlation between hash functions and the accuracy of the data sketch.

K-MER 32 RESULTS



As seen on the graphs, HyperLogLog performs better as the K-mer sizes are increased regardless of the hash functions. We think that this is caused by the fact that lengthier K-mer sizes result in larger quantities of distinct elements and HyperLogLog performs better estimating with larger quantities of distinct elements. The same pattern of improvement can be seen with the bits taken for the buckets, again a parameter of HyperLogLog. Yet this improvement peaks at 10 bits in this case.

The hash functions considered, the graphs show that when the distinct element size is smaller(K-mer length 32 leads to less distinct elements than 64) Tabulation Hash returns the most accurate results, with larger distinct element sizes it still performs great but gets rivaled by City Hash.

CONCLUSION

Findings so far suggest that HyperLogLog data sketch for K-mer counting gives more accurate results with bucket bits around 10 and a large distinct element size utilizing Tabulation and City Hashes, yet there are other conditions to consider. Right now, even though our implementation of HyperLogLog saves a lot of space, it runs sequentially, therefore it takes time to process the datasets; parallelization is needed to bring our model to more desirable speeds. Also in our estimation formula, there is a correction constant that differs for each bucket size which might need tweaking. Finally a file size of 1GB, like the ones we used, might seem large at first, when in bioinformatics, file sizes can go much larger. Tests need to be done on these types of data to have more accurate results to see which hash function performs the best for HyperLogLog sketch.

REFERENCES

- UCSC Genome Browser Downloads. (n.d.). Retrieved from http://hgdownload.cse.ucsc.edu
- Dahlgaard, S., Knudsen, M., & Thorup, M. (2017). Practical Hash Functions for Similarity Estimation and Dimensionality Reduction. In *Advances in Neural Information Processing Systems* (pp. 6615-6625).
- Mohamadi, H., Khan, H., & Birol, I. (2017). ntCard: a streaming algorithm for cardinality estimation in genomics data. Bioinformatics, 33(9), 1324-1330.
- Heule, S., Nunkesser, M., & Hall, A. (2013, March). HyperLogLog in practice: algorithmic engineering of a state of the art cardinality estimation algorithm. In *Proceedings of the 16th International Conference on Extending Database Technology* (pp. 683-692). ACM.
- google/cityhash. (2013, August 1). Retrieved from https://github.com/google/cityhash
- aappleby/smhasher. (n.d.). Retrieved from https://github.com/aappleby/smhasher/blob/master/src/MurmurHash3.cpp
- SpookyHash: a 128-bit noncryptographic hash. (n.d.). Retrieved from http://burtleburtle.net/bob/hash/spooky.html