Pythia: A Neural Model for Data Prefetching

Akshay Arun Bapat

University of Toronto
akshay.bapat@mail.utoronto.ca

1 Instance Optimized Database

Traditional databases are often general-purpose software systems that are typically not built for a specific workload or a data distribution. In general, these systems might provide good performance but probably not the optimal one. Recently, there has been a push towards leveraging machine learning based techniques to build database systems that are self-tuned or instance-optimized [4, 5, 3]. So far, it has achieved significant success in cardinality estimation and query optimization leading to faster query execution. However, there has been limited effort on optimization opportunities that are one level deeper, namely how to tailor the RDBMS buffer management module for correlated query workloads.

Buffer management [2, 1] has played a central role in improving RDBMS performance. Such improvement for a single query is achieved by exploiting locality of reference for page accesses and for multiple queries by enabling possible sharing of frequent page requests across queries. Both of these improvements can be pronounced if one can predict the access patterns of a query fairly accurately. An effective buffer manager ensures that the pages that are likely to be requested in the near future are prefetched in the buffer pool so that they do not result in costly blocked I/O operations later. Since predicting future access patterns is a challenging problem, RDBMSs employ empirical algorithms based on frequency and recency [2, 1]. While these approaches work well for simple access patterns, they often fail for more diverse and complex access patterns inherent in a typical OLAP setting.

2 Predicting query access patterns

Access patterns in databases are complex and challenging to predict. The access patterns of a complex SQL join query involving multiple relations are influenced by numerous factors such as selectivity (index scan or sequential scan), the join algorithm used (nested loop vs hash join), the order in which the relations are joined and so on. The use of indexes also violate sequential access patterns and results in *irregular sequences* due to the interleaving of accesses for index and base table pages.

Our empirical analysis shows that NLP techniques do not work well as the access patterns are too irregular (thereby having limited temporal patterns) but also very long (such as millions of tokens for a large relation). They also have distributional properties (frequency, co-occurrence and length of sequences) that are inimical to NLP based approaches. Additionally, they also require significant time for both training and inference.

`PYTHIA` is a deep ML predictive model tailored to predictions of the access patterns of complex correlated query workloads. It consists of two key components – a predictor and a prefetcher. Given a query, the goal of the predictor is to accurately output the relevant page accesses. The prefetcher then *asynchronously* fetches these pages and places them in the buffer pool.

We conduct extensive experiments with `PYTHIA` integrated into Postgres and demonstrate that it achieves significant accuracy and a speedup of upto 6x for queries in the DSB OLAP benchmark.

References

[1] Hong-Tai Chou and David J DeWitt. An evaluation of buffer management strategies for relational database systems. *Algorithmica*, 1(1-4):311–336, 1986.

[2] Wolfgang Effelsberg and Theo Haerder. Principles of database buffer management. *ACM Transactions on Database Systems (TODS)*, 9(4):560–595, 1984.

[3] Guoliang Li and Xuanhe Zhou. Machine learning for data management: A system view. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 3198–3201. IEEE, 2022.

[4] Guoliang Li, Xuanhe Zhou, and Lei Cao. Ai meets database: Ai4db and db4ai. In *Proceedings of the 2021 International Conference on Management of Data*, pages 2859–2866, 2021.

[5] Dimitris Tsesmelis and Alkis Simitsis. Database optimizers in the era of learning. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 3213–3216. IEEE, 2022.