Technical Disclosure Commons

Defensive Publications Series

August 14, 2018

Automatic Denormalization of Databases

Karthik Ravi Shankar

Gurunathan Nagarajan

Automatic denormalization of databases

ABSTRACT

Normalization is a technique in databases to eliminate data redundancy or inconsistent dependency. Normalizing is achieved by dividing larger tables of a database into smaller ones and defining relationships between them. Normalization can yield performance gains, such as improved response times, but only to an extent. Highly normalized databases are not performance optimal. Optimal database performance is often obtained at a sweet spot between optimizing sizes of individual tables and the number of tables. A highly normalized database is therefore often denormalized to improve performance. Traditionally, denormalization is driven by user or developer intuition.

This disclosure describes a machine-learning model to optimally denormalize a database based on, e.g., typical database queries, frequency of queries, response-times, projections of response time upon denormalization, etc. The techniques result in an optimal normal form of the database, in turn resulting in superior data integrity and performance.

KEYWORDS

database; relational database; database normalization; database denormalization; machine learning; database performance

BACKGROUND

Normalization is a technique in databases to eliminate data redundancy or inconsistent dependency. Normalizing is achieved by dividing larger tables of the database into smaller ones and defining relationships between them. By thus dividing the database, data that is not strictly germane to a particular query is not accessed. A database is said to go to a higher normal form upon normalization.

Highly normalized databases are not necessarily performance optimal. For example, there may be particular queries that cause retrieval of data spread across distinct tables of the database, causing a re-join of those tables, which can slow down responses to such queries. Performance optimality of a database thus occurs at a particular level of database normalization. To optimize query-response times, it is preferable in certain circumstances to denormalize a database, e.g., bring it to a lower normal form. In highly scalable systems, the level of normalization of a database is varied with the nature of queries. Traditionally, denormalization is driven by user or developer intuition, which can be slow and prone to human bias or errors.

DESCRIPTION

This disclosure describes use of machine-learning techniques to optimally denormalize a database to achieve superior data integrity and performance. The machine-learning model used for such denormalization is advantageously based on reinforcement learning, such that it learns from past predictions.

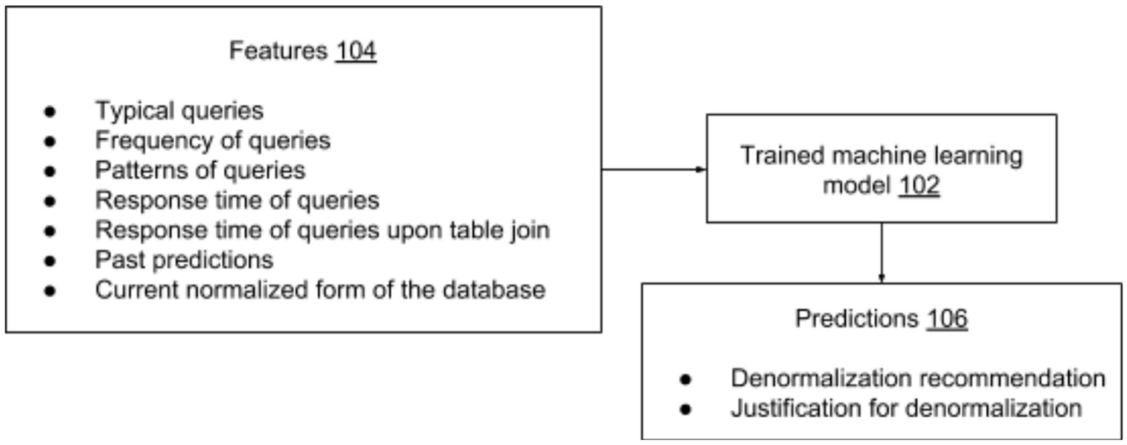

Fig. 1: Generating recommendations for denormalization

Fig. 1 illustrates a trained machine learning model (102) that provides predictions (106) for denormalization of databases. The predictions are based on input features (104) including,

e.g., typical queries, and frequencies, patterns and response-times thereof. Input features can also include projected response-times of queries if a certain table-join is performed (including time needed for table-join), performance of past predictions, the current normalized form of the database, etc. The model provides denormalization recommendations and justifications therefor (106) to the database designer, developer, or administrator.

The trained machine learning model can be, e.g., regression learning models, neural networks, etc. Example types of neural networks that can be used include long short-term memory (LSTM) neural networks, recurrent neural networks, convolutional neural networks, etc. Other machine learning models, e.g., support vector machines, random forests, boosted decision trees, etc., can also be used.

For faster and energy-efficient inferences, the machine-learning model can advantageously run on special-purpose hardware, e.g., GPU, DSP, processors or co-processors optimized for machine learning, etc. The machine-learning model can be run on a client device that is running an application that accesses the database, on a server, or on a combination of client device and server. Prior to operation, the machine-learning model is trained with known pairs of input features and optimal predictions. When database owners or administrators provide permission, the model also trains during operation using past predictions as feedback.

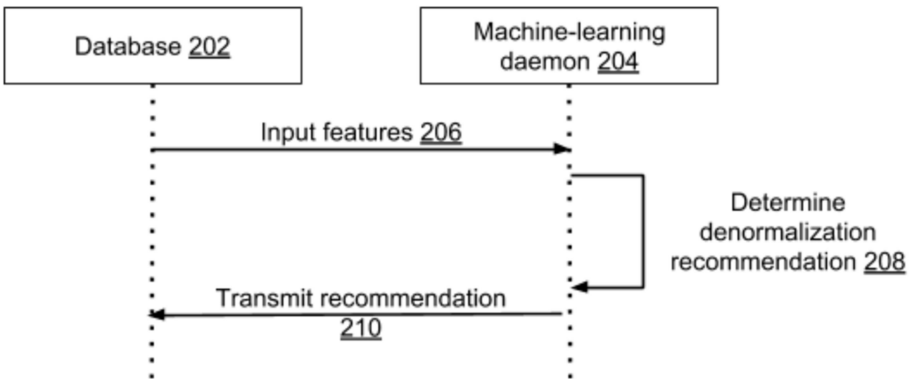

Fig. 2: Interactions between database and a trained model

Fig. 2 illustrates the interaction between the database and the machine-learning model, per techniques of this disclosure. The database (202) starts at the highest normal form (or any other form). The machine-learning model (204) is implemented, e.g., as a user space daemon separate from the database. Input features (206) regarding the database, e.g., as listed earlier, are provided to the model. Upon receiving input features, the machine-learning daemon determines a denormalization recommendation (208) and transmits the recommendation (210) to the database.

Implementing the machine-learning model as a user space daemon decouples the machine-learning model from the database, and also leverages the availability of user space libraries or frameworks. There is also a security advantage in keeping the trained model, which is less-privileged code, in user space, such that it interacts with the database via API calls.

In this manner, the machine-learning model (continuously) nudges the normalization level of the database towards a point where response-time is optimal given the nature of current queries to the database. The predictions of the machine-learning model include reasoning (or justification) behind a recommended table-join based on query patterns and statistics, which enables a database designer or administrator to make an informed decision on denormalizing the database. The techniques herein apply without substantial modification to any relational database. The techniques of this disclosure provide improved data integrity and performance of the database.

CONCLUSION

This disclosure describes a machine-learning model to optimally denormalize a database based, e.g., typical database queries, frequency of queries, response-times, projections of response time upon denormalization, etc. The techniques result in an optimal normal form of the database, in turn resulting in superior data integrity and performance.

REFERENCES

1. Lexa, Matej, and Radovan Lapár. "Semi-automatic mining of correlated data from a complex database: correlation network visualization." In 2016 *IEEE 6th International Conference on Computational Advances in Bio and Medical Sciences (ICCABS)*, pp. 1-2. IEEE, 2016.

2. "Automatic Denormalization" https://codeontime.com/blog/2012/04/automatic-denormalization