

In-memory Preprocessing of Streaming Sensory Data Partitioned Relational Database Approach

József Marton and Sándor Gajdos

As the application area of sensors broadens amount of streaming sensory data to be processed is also increasing. Analytical systems based on heterogeneous sensory data require efficient preprocessing techniques hence integrated view of different data sources must be provided in order to calculate data in a short time period. In this paper, we present a partitioned database approach - i.e. a composite of memory-resident and disc-resident partition. This approach can provide two to five-times performance improvement in the data preprocessing stage. It can adapt to hardware resources efficiently, i.e. adding more CPU cores and random access memory increases performance in a near-linear fashion. This is especially the case when data streams to be integrated are independent of each other - which is a common premise.

Off-line transformations have at least two drawbacks. Temporary storing of data at some staging area not only wastes I/O and communication bandwidth but also decreases usefulness as data lose their freshness. That's why, along with recent research and development, in this paper we focus on near real-time transformation and integration of data into the database (repository). Near real-time preprocessing demands for high computational capacity which can be provided using several different approaches. Our proposal follows a resource-sensitive way.

Architecture of the proposed database backend, as seen on Figure 1, is a two-tier one. A traditional disc-resident database management system provides the long-term storage. On the top of this 1st layer works a memory resident database. This means that the working set of the current data and a recent portion of the historical data reside in the physical memory. Application and transformation logic layers see a coherent view of the database and can neglect the disc-resident and memory-resident partitioning scheme applied.

As fresh data arrive into the database, it is stored in the memory-resident partition. This way it is available in the memory address space of the application driving the preprocessing. It is guaranteed by the partitioning scheme that no additional I/O cost arises during the transformation. This means that the cost of this phase solely depends on the function applied. The result of the transformation is appended to the repository of data used for instant monitoring and prediction. The proposed partitioning scheme also guarantees that data frequently queried for is available in the memory-resident portion thus enabling near real-time analysis.

A test system has been implemented to measure the performance gain of the proposed database architecture. We found that using a consumer class PC the proposed scheme was able to process data at 22% of the specified performance. To provide an easily comparable measure, we have replaced the partitioned database in our implementation by a solely disc-resident database tier, and found that it worked at a performance of 10% utilizing the same hardware. In contrast, the original 3rd party implementation that ran on a server platform processed just at a rate of 5%.

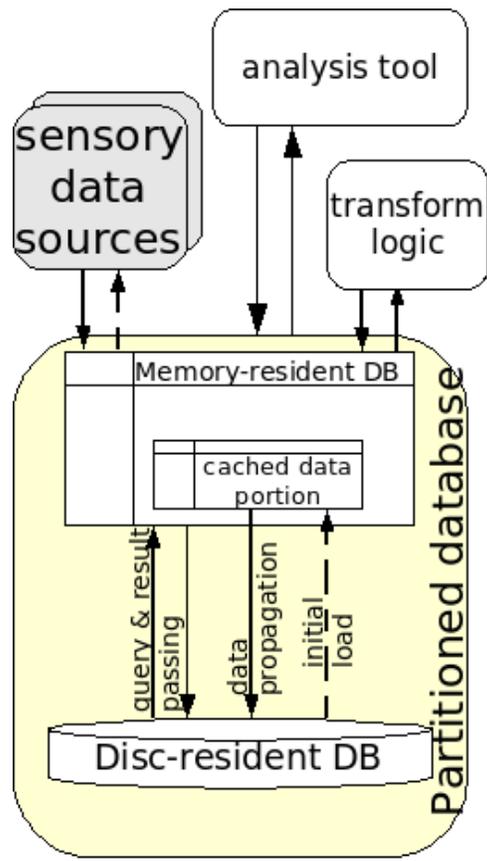


Figure 1: Architecture of the proposed partitioned database in its context.