Massive parallel in-memory database with GPU based query co-processor

Harald Frick
QuiLogic In-Memory DB Technology

ABSTRACT

This talk presents work on transforming SQL-IMDB, a commercial available in-memory database system, into a massive parallel, array structured data processor extending the "classic" query engine architecture with GPU based co-processing facilities. The chosen approach is not just a simple re-implementation of common database functionality like sorting, stream processing and joins on GPUs, instead we take a holistic view and extend the entire query engine to work as a genuine, in-memory, GPU supported database engine. We have partitioned the query engine so that both CPU and GPU are doing what they are best at. The new SQL-IMDBg query execution engine is a "Split-Work" engine which takes care to optimize, schedule and execute the query plan simultaneous and in the most efficient way on two (or more) different memory devices. The principal architecture of the engine, based on simultaneous managing multiple memory devices (local/shared/flash-memory), was a natural fit to include the new GPU/video memory as just another (high speed) memory device. All internal core engine data structures are now based on simple array structures, for maximum parallel access support on multi- and many core hardware. Data tables located on GPU video memory can always queried together with CPU local- and shared-memory tables in "mixed" query statements. Columns on GPU tables are also accessible through GPU based indexes. A special index structure was developed based on sorted containers supporting both CPU and GPU based index lookups. Table data can be manually and automatically split between CPU and GPU and is held in vertically partitioned columns, which ease the stream like processing for basic scan primitives and coalesced memory access mechanism on GPU devices. Based on our experience gained, we see the GPU/video memory as another important high speed memory device for in-memory database systems, but which do not yet fit well into the architecture of current database engines and therefore require a major effort in re-engineering the entire core database architecture.