CATEGORY: ALGORITHMS - AL08

POSTER P6256 | CONTACT NAME Steffen Wittmer: steffen.wittmer@jedox.com

GPU TECHNOLOGY CONFERENCE

FAST PARALLEL BULK INSERTION IN GPU MOLAP DATABASES

jedox.

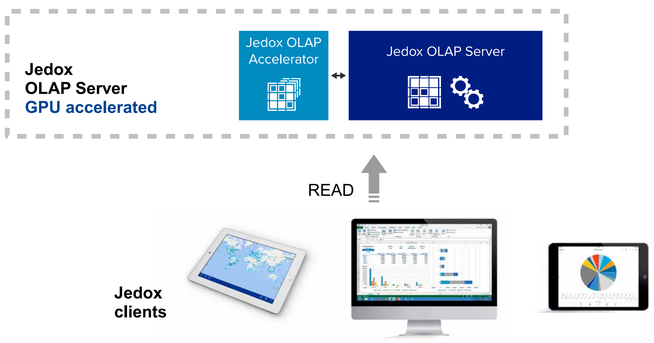
ABSTRACT

In-memory databases for online analytical processing (OLAP) offer high performance READ operations allowing for seamless and intuitive data analysis. With increasing amounts of available random access memory (RAM) in servers it is today possible to keep hundreds of millions of data points in memory and analyze them in almost real-time. In this context insertion of big data volumes imposes stronger requirements on performance of WRITE operations as well.

This work focuses on input processing of big data streams in an GPU accelerated in-memory OLAP (MOLAP) database by Jedox. We present a solution that supports fast insertion of high data volumes by avoiding the compute-expensive task of multidimensional sorting during the actual insertion phase. The main processing step achieves a significant speedup over existing CPU only version of the latest released Jedox OLAP server.

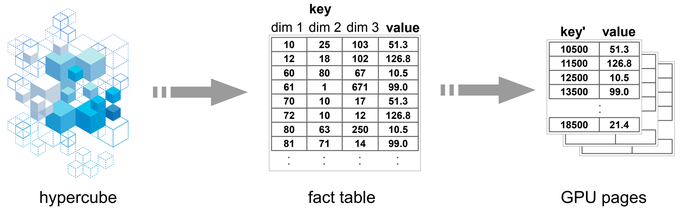
MOTIVATION

Typical scenarios for the occurence of high data volumes are finance, production, and social analytics as depicted below. Our solution targets scenarios with millions of data points being written each minute. In those scenarios it is not possible to trigger large write jobs at night in order to not interfere with processing of read requests sent by client software. Even big write operations are supposed to run concurrently with analytical operations such that run time performance becomes a major requirement for the implementation of the write task.



BACKGROUND

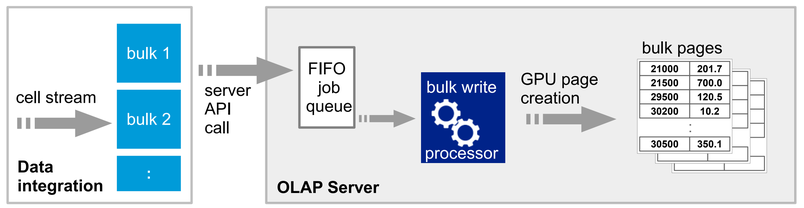
In OLAP data is modelled in a multidimensional cube, or hypercube. Each dimension of the hypercube represents an information category (e.g. Product, Region, or Date) containing entities (e.g. Product A) as elments. Each cell in the cube stores a multidimensional key (its address) and a value semantically defined by its key (e.g. Product A in Region B at Date C has value 100). The data storage can be imagined as a sorted fact table which can be subdivided into pages storing the multidimensional key in a compressed format. Because this format is used for GPU processing in Jedox OLAP Accelerator we call these pages *GPU pages*.



Internally a list of *clean pages* is kept containing sorted GPU pages with no intersections. At server startup all pages are clean. Analogously a list of *dirty pages* is kept that is updated whenever a write operation produces pages that are either not sorted internally and/or intersect with clean pages.

PRE-PROCESSING

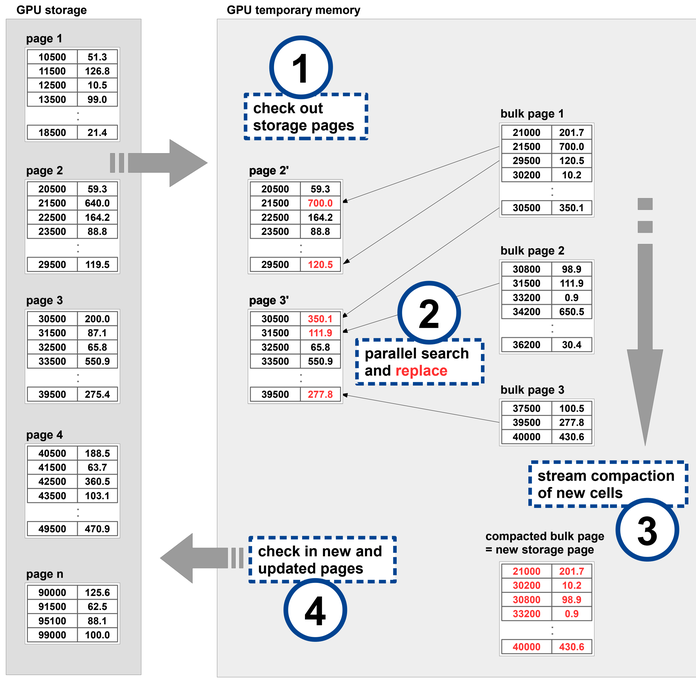
The pre-processing phase contains all steps performed until data is readily laid out in GPU memory. Input data is provided in a stream of cells by external systems and passed bulk-wise to the OLAP server in an *extract, transform and load* (ETL) process. On the server the cell bulk is transformed into *bulk pages* i.e. GPU pages residing in temporary global device memory holding input bulk data.



Note that on GPU memory the data storage page data is held in structure of arrays (SoA) format allowing for optimal parallel access. Arbitrary key lengths are supported. For sake of simplicity each key in the examples can be represented by a 64-bit unsigned integer. The value is stored in double precision.

MAIN PROCESSING

The principal idea in the presented solution is to skip a sorting step in the main processing of the bulk write operation. Here we exploit the fact that current implementation of Jedox OLAP Accelerator supports all operations to be performed on potentially unsorted storage, i.e. when dirty pages exist.



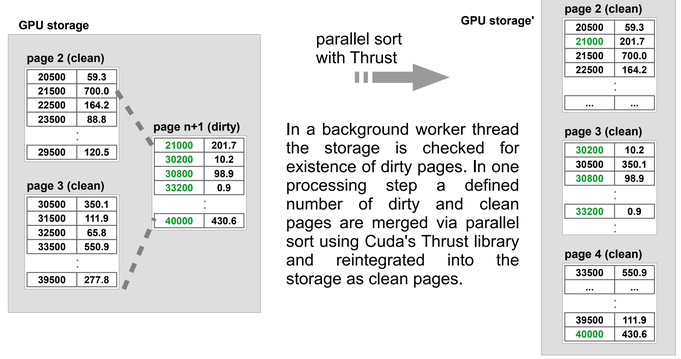
1 Pages to be updated are filtered according to key boundaries of bulk pages. In order to allow for readers to access the unmodfied version while the update takes place affected storage pages are cloned to temporary device memory.

2 A parallel binary search is performed to find a match in the bulk pages for each key in the storage pages. If found, the stored value is updated.

3 New cells to be inserted into the storage are filtered out by means of parallel stream compaction on bulk pages based on match information provided in the binary search step.

4 Checked out pages are re-inserted into the storage at their old position (because only values have changed, but not keys). All new pages are added to the storage. Note that pages in storage now might contain intersections. These will be removed in a background post-processing step.

POST-PROCESSING



In a background worker thread the storage is checked for existence of dirty pages. In one processing step a defined number of dirty and clean pages are merged via parallel sort using Cuda's Thrust library and reintegrated into the storage as clean pages.

EVALUATION

We evaluated the performance of the main processing step (including bulk page creation in GPU memory) on a productive social media database maintained by Jedox (see *Jedox Social Analytics Showcase* below). The data cube contains 18 million filled cells, making up 1 GB on RAM, and 0.5 GB on GPU RAM, respectively.

The test system configuration is as follows: Microsoft Windows Server 2012 R2, 2 Intel Xeon CPU E5-2643 @ 3.30 GHz (16 logical processors), 256 GB of DDR3 RAM, 1 NVIDIA Tesla K40 GPU. We compared run times with the latest release version of Jedox OLAP server without GPU acceleration (see *Jedox 6* below).

The run times for insertion yield a speedup factor of 4 compared to the CPU only version. As can be seen in the right figure there is not significant influence of the cube size (number of filled cells) on relative performance. Note that run times for background processing are not included in the figures. One integration step of dirty pages into the storage takes approximately 5 milliseconds and is performed whenever free resources are available. In addition we could not time a noteable speed difference for involved operations processed on (temporarily and locally) unsorted storage.

The speedup of up to factor 4 proves the overall effectiveness of the presented approach. The solution will be integrated into a future release of Jedox OLAP Accelerator.

Steffen Wittmer
steffen.wittmer@jedox.com

Alexander Haberstroh
alexander.haberstroh@jedox.com

Peter Strohm
peter.strohm@jedox.com

Jedox AG
Bismarckallee 7a, D-79098 Freiburg im Breisgau, Germany
Website: www.jedox.com
E-Mail: info@jedox.com

jedox.

Jedox Social Analytics Showcase

Get your free download of our showcase App and do Real-time Analysis of Social Media Data

available for iOS, Android and Microsoft

Jedox 6

Self-Service Business Intelligence & Performance Management

Discover free trial!

"Production Intelligence" is funded through the German Federal Ministry of Education and Research (BMBF) under the funding code 01IS15011 and is supervised by the German Aerospace Centre (DLR).

GEFÖRDERT VOM Bundesministerium für Bildung und Forschung

DLR Projektträger