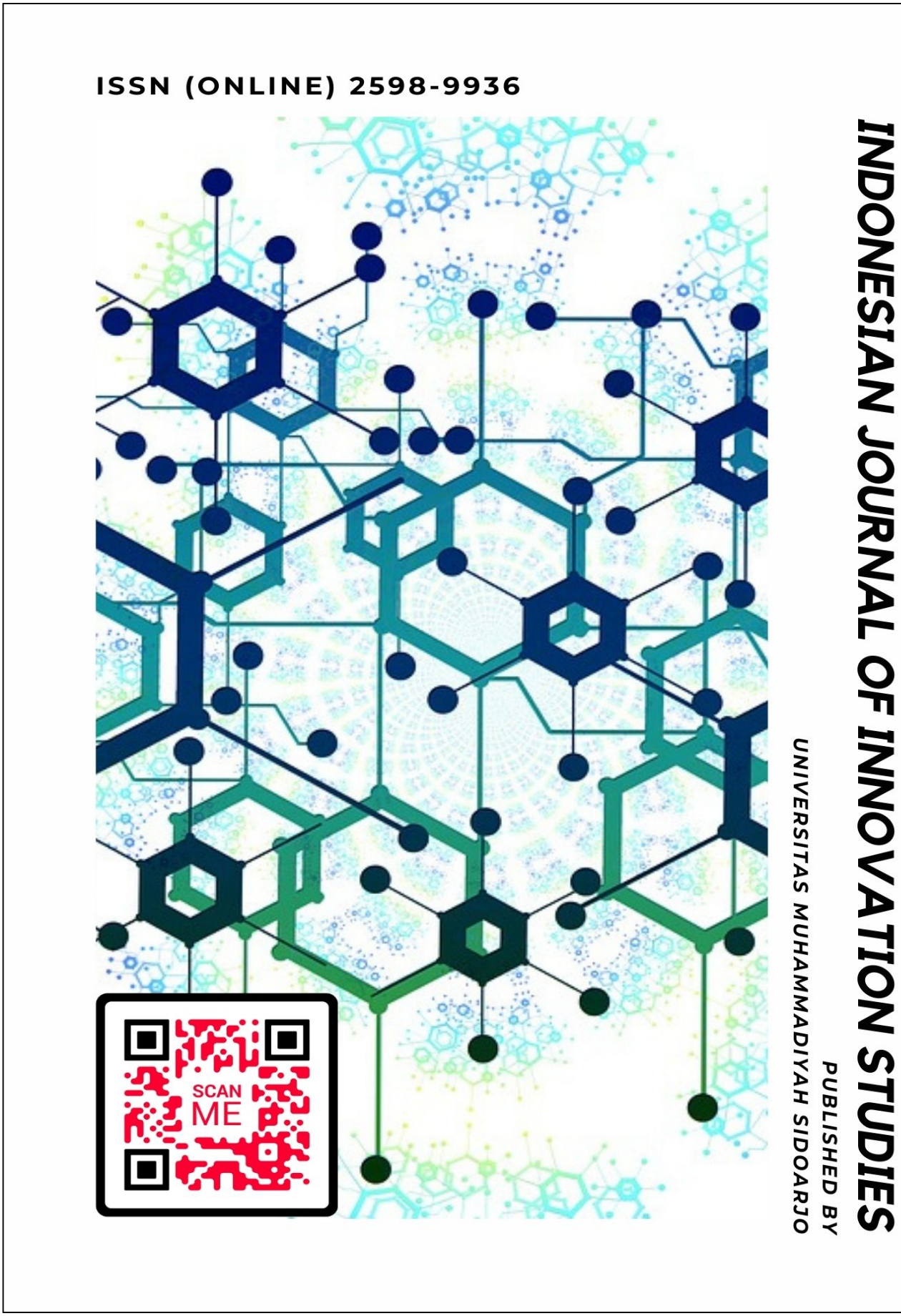
Indonesian Journal of Innovation Studies
Vol. 18 (2022): April . Article type: (Innovation in Computer Science)

Table Of Content

ISSN 2598-9936 (online), https://ijins.umsida.ac.id, published by Universitas Muhammadiyah Sidoarjo

Indonesian Journal of Innovation Studies
Vol. 18 (2022): April . Article type: (Innovation in Computer Science)



ISSN 2598-9936 (online), https://ijins.umsida.ac.id, published by Universitas Muhammadiyah Sidoarjo

Originality Statement

The author[s] declare that this article is their own work and to the best of their knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the published of any other published materials, except where due acknowledgement is made in the article. Any contribution made to the research by others, with whom author[s] have work, is explicitly acknowledged in the article.

Conflict of Interest Statement

The author[s] declare that this article was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright Statement

Indonesian Journal of Innovation Studies
Vol. 18 (2022): April . Article type: (Innovation in Computer Science)

EDITORIAL TEAM

ISSN 2598-9936 (online), https://ijins.umsida.ac.id, published by Universitas Muhammadiyah Sidoarjo

Indonesian Journal of Innovation Studies
Vol. 18 (2022): April . Article type: (Innovation in Computer Science)

Article information

Check this article update (crossmark)

Check this article impact (*)

Save this article to Mendeley

(*) Time for indexing process is various, depends on indexing database platform

ISSN 2598-9936 (online), https://ijins.umsida.ac.id, published by Universitas Muhammadiyah Sidoarjo

Indonesian Journal of Innovation Studies
Vol. 18 (2022): April . Article type: (Innovation in Computer Science)

Use of Graphics Processors in Data Compression Algorithms

Khalilova Shahnoza Uchkunovna, Uchkunovna@gmail.co, (1)
Master student at Tashkent University of information technologies, Uzbekistan

(1) Corresponding author

Abstract

Query co-processing on graphics processors (GPUs) has become an effective means to improve the performance of main memory databases. However, this co-processing requires the data transfer between the main memory and the GPU memory via a lowbandwidth PCI-E bus. The overhead of such data transfer becomes an important factor, even a bottleneck, for query co-processing performance on the GPU. In this paper, we propose to use compression to alleviate this performance problem.

Published date: 2022-04-12 00:00:00

ISSN 2598-9936 (online), https://ijins.umsida.ac.id, published by Universitas Muhammadiyah Sidoarjo

Introduction

Graphics processors (GPUs) have become an emerging and powerful co-processor for many applications including scientific computing [1] and databases [2]. The new-generation GPU has an order of magnitude higher memory bandwidth and higher GFLOPS (Giga FLoating point Operations Per Second) than the multi-core CPU. For example, an NVIDIA GTX 280 card contains 240 cores with a peak performance of 933 GLOPS and 141.7 GB/s memory bandwidth. Despite of the superb hardware performance, GPU co-processing requires data transfer between the main memory and the GPU memory via a low bandwidth PCI-E bus, e.g., with theoretical peak bandwidths of 4 and 8 GB/s on 16-lane PCIE v1.0 and v2.0, respectively. As a result, previous studies [3] show that such data transfer can contribute to 15–90% of the total time in relational query co-processing. Query co-processing performance can be further improved if such data transfer time is reduced. Compression has been long considered as an effec-tive means to reduce the data footprint in databases, especially for column-oriented databases [4, 5]. In this paper, we investigate how compression can reduce the data transfer time and improve the overall query co-processing performance on column oriented databases. Database compression has been extensively studied in column oriented databases (e.g., MonetDB/x100 [6] and C-store). Most of these systems adopt lightweight compression schemes, such as dictionary encoding, instead of more sophisticated compression algorithms such as gzip. With the lightweight compression schemes, column-oriented databases efficiently utilize vectorized executions, processing data tuples bulk-at-a-time or vector/array-at-a-time via (de)compressing multiple tuples simultaneously [7]. However, due to the poor decompression performance on CPUs, current systems often use a single compression scheme, even though cascaded compression (i.e., applying multiple lightweight compression schemes one after another on a data set) is potentially more effective. For example, Harizopoulos et al. showed that the column-oriented DBMS with the combination of two lightweight compression schemes hardly outperforms that with a single compression scheme. Given the superb memory bandwidth and GLFOPS of the GPU, we should revisit the commonly used lightweight compression schemes as well as their combinations on the GPU. The reduced data footprint by cascaded compression in GPU co-processing can potentially compensate the computational overhead.

GPU-based Data Processing

Due to the massive parallelism, GPUs have demonstrated significant speedup over CPUs on various query processing tasks, including sorting [8], index search, join [9], selection and aggregation. In addition to relational query processing, GPUs have been applied to other data management tasks, for example, data mining, spatial databases, MapReduce, scatter/gather and similarity joins. GPU-based data compression has been applied on the inverted index in information retrieval. In comparison, we not only investigate individual GPU-accelerated compression schemes for column-oriented databases, but also the combinations of different schemes for reducing the overhead of data transfer between the main memory and the GPU memory.

Database Compression

Compression is an effective method to reduce the data storage and to improve query processing performance. In disk-based databases, compression algorithms are typically computation-intensive, and they trade more computation for better compression ratios, since disk accesses are far slower than the CPU. Because the compression algorithms are complex, the compressed data are often required to be fully decompressed for query processing.

As the price of main memory drops, machines tend to have large amounts of main memory. Moreover, column-oriented databases allow query processing to read only the required columns, instead of the entire rows. Disk I/Os become less significant for such scenarios. As a result, studies on compression techniques in columnoriented databases [10] have focused on lightweight schemes such as Run-Length-Encoding (RLE) that balance between compression ratio and computational overhead. With these lightweight compression schemes, most queries can be evaluated without decompression. In addition, lightweight compression can easily be vectorized. Although the combination of lightweight schemes is undesirable on CPU-based column-oriented databases due to the computation overhead, the high memory bandwidth and the high computation capability of GPUs make combinations of multiple compression schemes practical. Moreover, compression can benefit GPU query co-processing by significantly reducing the overhead of data transfer between the main memory and the GPU memory.

CASCADED compression

Based on the GPU-accelerated, individual compression schemes, we further investigate the combinations of multiple compression schemes (namely cascaded compression) on the GPU. The core benefit of cascaded compression is to further reduce the data size by applying more than one lightweight scheme. While cascaded compression remains of low interest on the CPU, it appears promising to further alleviate the performance issues in data transfer and query processing on the GPU.

ISSN 2598-9936 (online), https://ijins.umsida.ac.id, published by Universitas Muhammadiyah Sidoarjo

The main question in cascaded compression is, given a number of individual compression schemes and a data set (a column), how do we find a feasible and good combination of the individual schemes (compression plan)? Our answer to this question is through a compression planner together with a cost model. In this section, we first present our compression planner, and then describe our cost model for GPU-based compression schemes.

Conclusion

GPU co-processing has demonstrated significant performance speedups on main memory databases. This paper further improves the performance of such co-processing with database compression techniques. Compression not only improves the query processing performance, but also alleviates the overhead of data transfer on the low-bandwidth PCI-E bus in GPU co-processing.

References

1. N. K. Govindaraju, S. Larsen, J. Gray, and D. Manocha. A memory model for scientific algorithms on graphics processors. In Supercomputing, 2006.
2. N. K. Govindaraju, B. Lloyd, W. Wang, M. Lin, and D. Manocha. Fast computation of database operations using graphics processors. In SIGMOD, 2004.
3. B. He, M. Lu, K. Yang, R. Fang, N. K. Govindaraju, Q. Luo, and P. V. Sander. Relational query co-processing on graphics processors. In TODS, 2009.
4. D. Abadi, S. Madden, and M. Ferreira. Integrating compression and execution in column-oriented database systems. In SIGMOD, 2006.
5. D. J. Abadi, S. R. Madden, and N. Hachem. Column-stores vs. row-stores: how different are they really? In SIGMOD,2008.
6. P. Boncz, M. Zukowski, and N. Nes. Monetdb/x100:Hyper-pipelining query execution. In CIDR, 2005.
7. M. Zukowski, S. Heman, N. Nes, and P. Boncz. Super-scalarRAM-CPU cache compression. In ICDE, 2006.
8. N. K. Govindaraju, J. Gray, R. Kumar, and D. Manocha. Gputerasort: High performance graphics coprocessor sorting for large database management. In SIGMOD, 2006.
9. B. He, K. Yang, R. Fang, M. Lu, N. K. Govindaraju, Q. Luo, and P. V. Sander. Relational joins on graphics processors. In SIGMOD, 2008.
10. C. Binnig, S. Hildenbrand, and F. Faerber. Dictionary-based order-preserving string compression for main memory column stores. In SIGMOD, 2009.

ISSN 2598-9936 (online), https://ijins.umsida.ac.id, published by Universitas Muhammadiyah Sidoarjo