Big Data Approach to Fluid Dynamics Visualization Problem* **

Vyacheslav Reshetnikov[1 [0000−0002−3953−3440]], Egor Golubchikov[1 [0000−0002−0366−1424]], Andrey Pyatlin[1 [0000−0002−1851−702X]], Alexey Kuzin[1 [0000−0002−3715−5829]], Vladislav Kiev[1 [0000−0002−3790−2994]], Nikolay Shabrov[1 [0000−0001−7721−1824]], Alexey Zhuravlev[1,2 [0000−0002−9973−1705]], and Ekaterina Guseva[1 [0000−0002−7117−2454]]

[1] Peter the Great St.Petersburg Polytechnic University, St.Petersburg, Russia
[2] Reutlingen University, Reutlingen, Germany

Abstract. Present work is dedicated to development of the software for interactive visualization of results of simulation of gas dynamics problems on meshes of extra large sizes. Kitware ParaView visualization tool, which is popular among engineers and scientists is used as a frontend. The coupling of client and server instances of ParaView is used in the project. The crucial feature of the work is an application of Apache Hadoop and Apache Spark for distributed retrieving of simulation data from files on hard disk. The data is stored on the cluster in Hadoop Distributed File System (HDFS) managed by Apache Hadoop and is provided to ParaView server by Apache Spark data processing tool.

Keywords: Visualization · Spark · ParaView · Parquet.

1 Introduction

The capabilites of modern high performance computers and the level of development of special problem-oriented software packages of predictive modeling allow the user to increase resolution of numerical grids up to the order of billions nodes and more. Files of simulation results on larger meshes are represented by big data arrays, especially in the case of modeling of unsteady processes. Investigations show that there is a growing trend in the size of the data [1]. The size of data retrieved causes a problem of low speed of scientific visualization and analysis of the results.

While software for predictive modeling and hardware available for the wide spectrum of scientists allow to perform fluid dynamics evaluations on meshes up to 10 billions cells, visualization tools do not provide desired efficiency. One of the problems is low visualization speed. Moreover, it is not only linked with the

* Supported by Russian Science Foundation (Grant No. 18-11-00245)
** The research carried out with the financial support of the grant from the Program Competitiveness Enhancement of Peter the Great St.Petersburg Polytechnic University

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_38

rendering. Childs et al. [2] showed that time of input/output can be two orders higher than rendering and evaluations. One of the ways to reduce the time of input/output is the development of special algorithms for data processing not involving supercomputers technologies. This approach was performed in [3], [4], [5]. In this paper authors use another approach which assumes usage of supercomputer.

There is a lack of good tools which are able to handle extra large grids on the market of visualization systems. The most known scientific visualization brands such as ParaView, TecPlot, COVISE, TechViz do not support information about effective results presentation on the meshes of up to billion nodes large. At the same time, this kind of problem persists for aircrafts designers.

The scientific problem considered in this article is the problem of achieving an effective and rapid interactive visualization and analysis of results of predictive modelling of fluid dynamics problems for modern aircrafts on superlarge meshes. It is assumed that created software can be used by engineers who do not have any special knowledges in IT, that is why it should provide displaying of the results in the most convenient and easy to understand way. The software should give visualization of fields in real or almost real time for comfortable working process.

The key idea underlying the developed software package is the usage of distributed big data analysis tools such as Apache Hadoop in conjunction with Apache Spark [6]. They provide distributed retrieval of the data from cluster nodes that can seriously reduce time of data reading in comparison to traditional sequential approach. Hadoop is mainly used to support Hadoop Distributed File System (HDFS), and the server built on the base of the Spark framework provides distributed processing of queries for retrieving required dataset from a cluster. A plugin to ParaView developed by the authors, plays the role of client of Spark server. It is intended to send queries and is integrated with the server version of ParaView. The user's computer has the client version of ParaView, which receives final results of rendering from server ParaView.

The problem with the Apache Hadoop application for creation of packages for visualization of results of finite element modelling is considered in [7]. The example of an effective usage of HDFS is presented in [8]. An approach similar to the authors one was investigated in [9] with the difference that in that work Apache Hive was used instead of Apache Spark. Also the article [10] should be mentioned in which a hybrid approach is offered that assumes usage of HDFS and Kitware ParaView as a user interface. In the paper [12] Hadoop and Spark frameworks were applied for analysis and visualization of atmospheric phenomena modeling in Earth science problems. The attention was paid mainly to the analysis tasks.

2 Architecture

In this section the software architecture is described, as well as how particular program parts interact with each other during the typical visualization process.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_38

The software environment is built by "client-server" scheme and has the structure which is shown on the figure 1. Basic elements are:

- **Client ParaView**. The client version of the well-known scientific visualization Kitware's ParaView package is installed on the local computer and is intended to provide direct interaction with the user. It visualizes results of rendering obtained from ParaView server. This is the only component of the software, which works on the local machine.
- **Server ParaView**. The server version of ParaView which is installed on the cluster. It provides effective parallel rendering based on the data retrieved with the plugin to ParaView, designed by the authors.
- **Plugin to ParaView**. The plugin is made by the authors and is integrated into the server Paraview and is intended to efficient data reading. The reading is processed through the responds to the client SQL-queries to data server that is also run on the same or another cluster.
- **Data server**. The server is written using Python and uses the Apache Thrift framework. It receives queries from the ParaView plugin and gives data blocks back. The server forwards queries to the Spark system that retrieves data in a distributed manner from Hadoop Distributed File System.

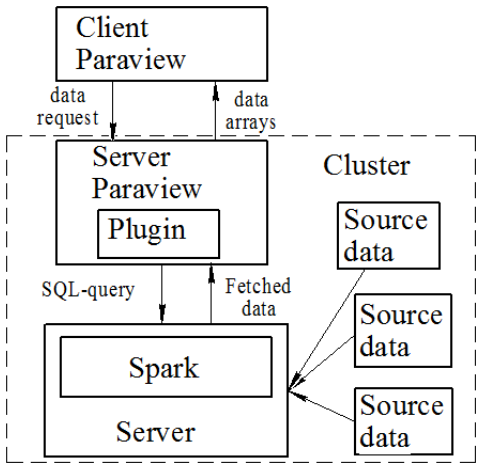

Fig. 1. The scheme of interaction of the software components

The interaction of client and server parts of ParaView is quite traditional approach of usage of ParaView that provides parallel model rendering [13]. Therefore, the development of the ParaView plugin and the data server is the main direction of the authors efforts. From the ParaView point of view the plugin is the regular plugin for the reading of the model data defined on the structured grid. At the present time the plugin gives VTK-object `vtkMultiBlockDataset`.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_38

But instead of direct data reading, plugin forms SQL-query to the data server and retrieves data in responce. The server transfers the query to Apache Spark. Spark executes distributed reading, collecting (`collect` operation) and sending data as a responce to the SQL-query. The advantage of such a scheme over direct reading from file is that the reading via Spark is performed in parallel on several nodes of the cluster, which can give an increase in speed, especially on large files.

The speed of reading essentially depends on the file format used. Initially, data is presented in text format Tecplot and takes 1.27 Gb. It is not suitable format for holding large data so it must be converted into another format. The format to be converted in must meet the following requirements. Firstly, the file size must be as small as possible. Secondly, it must provide relatively fast access to individual blocks of data, which is important in cases there is no need to read the entire file. It must support distributed storage and be readable by Spark. Based on the sum of these requirements, Apache Parquet [14] was chosen as the data format. On the one hand, it meets all requirements above. It is binary format so it has a smaller size in comparison to Tecplot. After conversion one frame of data takes about 400 MB. Besides, it provides fast accsess, distributed storage and can be read by Spark. Also it has a sufficient flexibility to form the necessary block structure of data storage. The format itself is a set of columns organized into a hierarchical structure under the control of a special scheme. The choice of the scheme is left to the person who writes the file. The scheme is the part of the format written to the file's metadata and can be restored while reading. Thus, a Parquet file can contain a highly complex hierarchical structure. Although Apache does not describe details of format implementation it provides open-source API for reading and writing. The basic ideas of working with Parquet are taken from [15].

As a method of storing data on a structured grid, a block structure was chosen, which is obtained from the initial index of parallelepiped grid elements by dividing it by parallel index planes in three directions. Each block spatially occupies an area in the form of a curvilinear hexahedron and represents a structured grid. The storage of such block inside the Parquet file is organized as a set of separate Parquet columns for each of the node coordinates and field components. Due to the specifics of the Parquet format, column addresses are stored in the file's metadata and each column can be accessed directly without reading the entire file so it solves the problem of selective reading of the necessary grid blocks.

At the moment, the solution is not adaptive to the manner of distribution of simulation output. That is why data must be redistributed in case of change of computer nodes numbers. Solution adaptivity to this kind of changes is the point for further research.

3 Usage example

An example of visualization of unsteady gas dynamics simulation results on the structured grid of hexahedrons is considered. The source data is written in

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_38

the form of time layers. Then each layer is initially stored in a separate file in Techplot format. The model contains about $5 \cdot 10^6$ nodes and each Techplot file has size of 1.27 Gb.

The processing of separate frame files during direct visualization of these frames sequence on a standalone computer in ParaView takes about 30 seconds, which is unacceptably slow for interactive mode. Such low speed can be attributed to the fact that Tecplot is not suitable format to hold an information of big data.

The same data is visualized with developed software. All components except of client ParaView are installed on the cluster. The nodes of supercomputer "RSC Tornado" of Saint-Petersburg Polytechnic University Supercomuter Center are used as a cluster for ParaView and data servers. Each node consists of two CPU Intel Xeon E5-2697 v3 (14 cores, 2.6 GHz) and 64 Gb RAM DDR4. Simple reading of Tecplot datafile located on one node with Paraview takes about 60 seconds.

Data files were converted to Parquet format before visualization in the developed environment and the size of one frame was reduced to about 400 MB. There were 20 frames. In total, the size of all frames was about 7.8 GB. The data of such rather small size is used only to illustrate approach. In further research size of each file is proposed to be bigger. During writing to Parquet format the data was transformed as described above: initial index parallelepiped of structured grid was divided by mutually orthogonal index planes into the parallelepipeds of smaller sizes. Inside each box each coordinate and each component field is separate Parquet column. Such a structure is effective on extraction of selected blocks because it does not require reading of the entire file. Each conversion takes about 40-45 seconds. In total, the whole dataset is converted on 10 MPI threads for 90 seconds.

Apache Spark is launched on the cluster under control of Slurm system. The reading of Parquet files in Spark is performed by the pyarrow library, as it provides a higher speed and requires significantly less memory than built-in Spark tools for Parquet reading. Data in Spark is represented in non-hierarchical plain structure.

Table 1. Time of reading and displaying of separate frames on 8 nodes of the cluster "RSC Tornado"

Frame	Reading time, s	Total time, s	fps
1	4.655	49.314	0.020
2	2.794	14.142	0.071
3	2.723	11.862	0.084
4	2.790	10.895	0.092
5	2.956	15.596	0.064

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_38

The time of displaying of the first five frames using 8 cluster nodes is shown in the table 1. The second column contains the time of direct reading of the Parquet file in Spark using pyarrow. The third column is the total time of the frame displaying. The last column is the number of frames per second, i. e. the inverse of the total time. Most of the time is spent on data transfer from the data server to the client ParaView, which indicates the need to use a faster network. The time of reading and transmitting the first frame is longer than the next ones, which is caused first of all by the necessity of reading the grid. The fact is that the grid remains unchanged during the transition from frame to frame, so it is read only once at the first frame. Starting from the second frame the system reads only the values of the displayed field. That is why from the second frame reading time does not change.

The increment of the number of nodes involved does not lead to the expected decrement of reading time in Spark. Finding the cause of this phenomenon is one of the tasks of further research.

4 Conclusions

A software environment for interactive visualization is developed. It provides visualization of simulation results evaluated on large numerical grids. The environment consists of a client ParaView, a server ParaView, a data server that forwards SQL queries to Apache Spark. The latter is used to increase the speed of reading large data by providing distributed access to them.

The experiments have shown the effectiveness of using data in Parquet format. Compared to the Tecplot text format this format provides a smaller file size, is directly readable by Apache Spark, and provides an ability to extract individual data blocks without having to read the entire file.

The experiments also showed the absence of scalability of the data reading speed in Spark with increasing number of nodes and high overhead for data transfer from Spark to the server PareView. These problems are tasks for further development.

Another challenge is to use more specific SQL queries. These can be requests to get data corresponding to the visible part of the model. In addition, there might be requests to retrieve data distributed across layers. The layer which data should be extracted depends on the camera position. If it corresponds to a higher detalization, the layer must contain more nodes.

References

1. Jin, X., Wah, B.W., Cheng, X. , Wang, Y.: Significance and challenges of big data research. Big Data Research, **2**(2), 59–64. (2015)
2. Childs, H., Brugger, E., Bonnell, K., Meredith, J., Miller, M., Whitlock, B., Max, N.: A contract based system for large data visualization. In: Visualization, 2005. VIS 05. IEEE, pp. 191–198. (2005)

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_38

3. Belyaev, S., Shubnikov, V., Motornyi, N.: Adaptive screen sampling algorithm acceleration for volume rendering. In: MCCSIS 2018 - Multi Conference on Computer Science and Information Systems; Proceedings of the International Conferences on Interfaces and Human Computer Interaction 2018, Game and Entertainment Technologies 2018 and Computer Graphics, Visualization, Computer Vision and Image Processing 2018, pp. 377–381. (2018)
4. Belyaev, S., Smirnov, P., Shubnikov, V., Smirnova, N.: Adaptive algorithm for accelerating direct isosurface rendering on GPU. Journal of Electronic Science and Technology, **16**(3), 222–231. (2018) doi:10.11989/JEST.1674-862X.71013102
5. Savchuk, D. A., Belyaev, S. Y.: Two-pass real-time direct isosurface rendering algorithm optimization for HTC vive and low performance devices. Paper presented at the Progress in Biomedical Optics and Imaging - Proceedings of SPIE, 10579. (2018) doi:10.1117/12.2292183
6. Apache Spark framework. http://spark.apache.org. Apache Spark is developed by Apache company. https://apache.org
7. Lange, B., Nguyen, T.: A Hadoop distribution for engineering simulation. [Research Report] INRIA Grenoble - Rhne-Alpes (2014)
8. Voinov, N., Drobintsev, P., Kotlyarov, V., Nikiforov, I.: Distributed OAIS-based digital preservation system with HDFS technology. In: 2017 20th Conference of Open Innovation Association, (FRUCT), St.Petersburg, pp. 491–497. (2017) doi:10.23919/FRUCT.2017.8071353
9. Artigues, A., Cucchietti, F. M., Montes, C. T., Vicente, D., Calmet, H., Marin, G., Houzeaux, G., Vazquez, M.: Scientific Big Data Visualization: a Coupled Tools Approach. Supercomputing Frontiers And Innovations, **1**(3), 4–18. (2014)
10. Mitchell C., Ahrens J., Wang J.: VisIO: Enabling Interactive Visualization of Ultra-Scale, Time Series Data via High-Bandwidth Distributed I/O Systems. In: Parallel & Distributed Processing Symposium (IPDPS), 2011 IEEE International, pp. 68–79. (2011)
11. Shujia Zhou, Xi Yang, Xiaowen Li, Toshihisa Matsui, Si Liu, Xian-He Sun, Weikuo Tao.: A Hadoop-Based Visualization and Diagnosis Framework for Earth Science Data. In: IEEE International Conference on Big Data, pp. 1911–1916. (2015)
12. Shujia Zhou, Xiaowen Li, Toshihisa Matsui, Weikuo Tao.: Visualization and Diagnosis of Earth Science Data through Hadoop and Spark. In: IEEE International Conference on Big Data, pp. 2974–2980. (2016)
13. ParaView software. http://www.paraview.org. ParaView is developed by Kitware company. http://www.kitware.com.
14. Columnar storage format. http://parquet.apache.org. Parquet is developed by Apache company. https://apache.org
15. Melnik S., Gubarev A., Long J. J., Romer G., Shivakumar S., Tolton M., Vassilakis T.: Dremel: Interactive Analysis of Web-Scale Datasets. In: Proc. of the 36th Int'l Conf on Very Large Data Bases, pp. 330-339. (2010)

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_38