SRSPG: A Plugin-based Spark Framework for Large-scale RDF Streams Processing on GPU

Tenglong Ren, Guozheng Rao, Xiaowang Zhang*, and Zhiyong Feng

College of Intelligence and Computing, Tianjin University, Tianjin, China
Tianjin Key Laboratory of Cognitive Computing and Application, Tianjin, China
* Corresponding author: xiaowangzhang@tju.edu.cn

Abstract. In this paper, we propose a plugin-based Spark framework (SRSPG) for large-scale RDF streams processing on GPU. Within this framework, We convert RDF streams to a RDF graph in a unified and simple way. Then we can apply various SPARQL query engines to process continuous queries and utilize GPU to accelerate queries. Computation Module provides a Spark-based Join algorithm utilizing GPU for parallel joining, obtaining the final results. Besides, we provide Compute Resource Management to balance the scheduling and task execution between GPU and memory resources. Finally, we evaluate our work bulit on gStore and RDF-3X on the LUBM benchmark. The experimental results show that SRSPG is effective for real-time processing of large-scale RDF streams.

1 Introduction

RDF streams, as a new type of dynamic dataset, can model real-time information in traffic monitoring, intelligent city and other fields. Real-time processing of large-scale RDF streams has become an important research topic nowadays. What is more, most of the existing RSP(RDF Steam Processing) systems are centralized, such as C-SPARQL [2], CQELS [4], EP-SPARQL [1]. These engines can not process large-scale RDF streams. A framework PRSP [5, 3] is presented to process continuous queries on large-scale RDF streams by exploiting various SPARQL query engines in a unified way.

Apache Spark is a fast computing engine designed for large-scale data processing. The intermediate results are stored in memory, so that large-scale data can be processed better. Graphics Processing Units(GPUs) is an efficient parallel processing way, which is widely applied and provides a higher level of speedup by executing multiple threads synchronously. However, there are few parallel computing systems based on GPU and Spark to process large-scale RDF streams. In addition, the scheduling mechanism of Spark and the task of GPU have to be taken into account.

In our work, we propose a plugin-based Spark framework for large-scale RDF streams processing on GPU. In order to make full use of computing power of GPU, we add Query Split module and Computation Module. The Query Split decomposes the SPARQL queries. Computation module is used to compute the intermediate results through on GPU. We evaluate our experiments on the benchmark LUBM. The experimental results show that our framework is effective and efficient.

2 Overview of SRSPG

The framework of SRSPG consists of the following six main parts: *Syntax Translator*, *Data Transformer*, *Query Trigger*, *Query Split*, *SPARQL API*, and *Computation Module* shown in Figure 1.

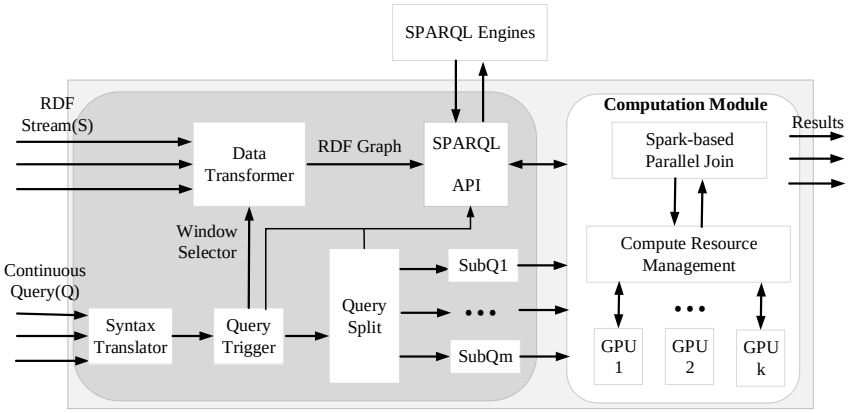

Fig. 1. The framework of SRSPG.

Syntax Translator The module of Syntax Translator translates continuous queries into unified queries. The translated query is sent to Query Trigger module.

Query Trigger The part of Query Trigger receives the unified queries, splitting them into two parts: $\rho(Q)$ and window selector. $\rho(Q)$, as the core SPARQL query, is pushed into SPARQL API and Query Split module. Window selector is in the form of a 6-tuple which are encapsulated into a management subsets and sent to Data Transformer. Let $p = [S, \alpha, \beta, \gamma, tf, \rho(Q)]$ where S is the RDF stream to be processed; α represents the window size; β is the updating time of windows; γ is the updating latency of windows; tf is used to determine whether RDF streams are processed. $\rho(Q)$ is the core SPARQL query.

Data Transformer This module transforms RDF streams to capture snapshots based on window selector obtained from Query Trigger, and converting these snapshots to RDF graphs. We convert RDF streams into continuous window data by Esper or other DSMS. Finally, the RDF graphs are sent to SPARQL API.

SPARQL API SRSPG provides SPARQL API for users, which makes it possible for SPARQL engines (centralized and distributed) to process RDF streams.

Query Split Query Split module decomposes queries into some subqueries based on the weight of predicates. We assign weights to predicates when loading RDF data. The higher the frequency of predicates in RDF graph, the greater the weight and the greater the impact on query results.

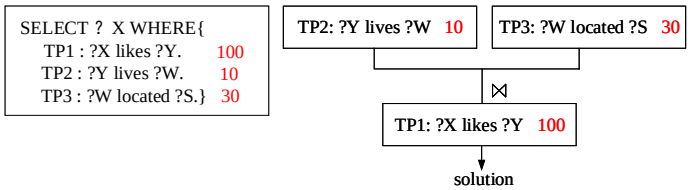

Fig. 2. An example of SPARQL query splitting.

Computation Module The module of Computation Module proposes Join parallel algorithm based on Spark, utilizing GPU for parallel joining. The tasks can be disassembled into some streams. Spark supports multi-threaded computing, while GPUs are usually serial. The situation leads to contention for GPU resources among threads. So we present Compute Resource Management to balance the scheduling and task execution of GPU, GPU and memory resources.

3 Experiments and Evaluations

Our experiments are evaluated on server equipped with a 4 CPUs with 6 cores and 64GB memory, a NVIDIA GTX590 GPU, which has 24GB device memory and is clocked at 1.35GHz. The version of operating system is Ubuntu 14.04. In order to support GPU on YARN, we use version 3.1.2. We use RDF-GPU and gStore-GPU by employing RDF-3X and gStore within SRSPG. Our experiments utilized LUBM dataset.

The experiments uses the standard query Q1 and Q2 provided by LUBM. Figure 3 shows that when S = 240s and SET = 235s, SRSPG uses GPU, the query time of stream data decreases, gStore improves the speed by more than two times than RDF-3X. When the data size is small, GPU acceleration is not obvious, mainly due to data communication and transmission time problems. The larger the data scale, the better the acceleration effect.

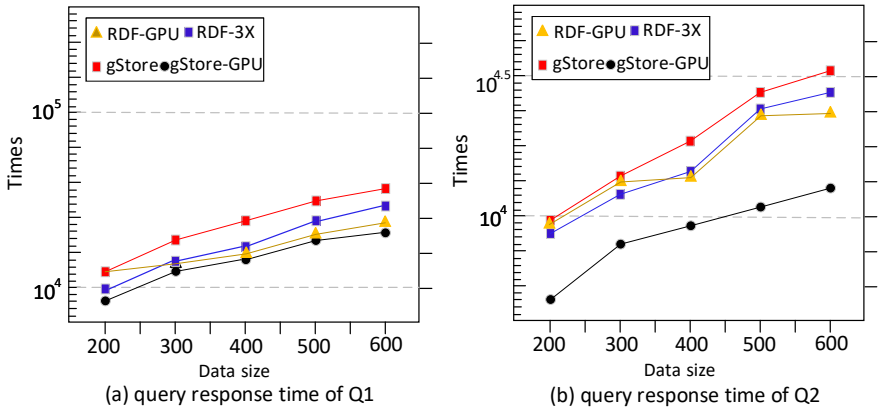

Fig. 3. Querying time in different engines under GPU and CPU.

4 Conclusions

In this paper, we proposed a plugin-based Spark framework for real-time large-scale RDF streams processing on GPU in an efficient and simply way. In the future, we will take advantage of more novel computing hardware to increase the speed of large-scale RDF streams processing such as FPGA.

5 Acknowledgments

This work is supported by the National Key Research and Development Program of China (2017YFC0908401) and the National Natural Science Foundation of China (61672377,61972455). Xiaowang Zhang is supported by the Peiyang Young Scholars in Tianjin University (2019XRX-0032).

References

1. Anicic D., Fodor P., Rudolph S., and Stojanovic N.: EP-SPARQL: A unifed language for event processing and stream reasoning. In: *Proc. of WWW 2011*, pp. 635–644.
2. Barbieri D.F., Braga D., Ceri S., Della Valle E., Grossniklaus M.: Querying RDF streams with C-SPARQL. *SIGMOD Rec.*, 39(1), 20–26 (2010).
3. Fang H., Zhao B., Zhang X., Yang X.: A united framework for large-scale resource description framework stream processing. *J. Comput. Sci. Technol.*, 34(4): 762-774 (2019).
4. Le-Phuoc D., Dao-Tran M., Parreira J. X., Hauswirth M.: A native and adaptive approach for unifed processing of linked streams and linked data. In: *Proc. of ISWC 2011*, pp.370–388.
5. Li Q., Zhang X., Feng Z.: PRSP: A plugin-based framework for RDF stream processing. In: *Proc. of WWW 2017 (Poster)*, pp.815–816.