Eurostars Project

3DFed – Dynamic Data Distribution and Query Federation

Project Number: E!114681 **Start Date of Project:** 2021/04/01 **Duration:** 36 months

Deliverable 5.2
A Report on 3DFed Evaluation based on LinkedGeoData and DBpedia Use Cases

Dissemination Level	Public
Due Date of Deliverable	April 30, 2024
Actual Submission Date	April 30, 2024
Work Package	WP5, Use Cases
Deliverable	D5.2
Type	Report
Approval Status	Final
Version	1.0
Number of Pages	15

Abstract:
Deliverable D5.2 (Report on 3DFed evaluation based on LinkedGeoData and DBpedia use-cases) aims to provide an overview and describe the results of the evaluation we have performed on the 3DFed tools and algorithms, namely the CostFed tool and the algorithms for partitioning RDF graphs, in the context of the large RDF knowledge graphs selected by OpenLink: the LinkedGeoData and the DBpedia datasets. The LinkedGeoData (LGD) dataset contains 1.3 billion RDF triples, while the DBpedia dataset contains 1.2 billion of them. The queries evaluated are real queries from the query logs of the official endpoints, which number in the hundreds of thousands. Additionally, we describe the creation of the LSQ metadata dataset for the DBpedia knowledge graph, which was used for the evaluation process, as well.

3DFed Project by Eurostars.

History

Version	Date	Activity	Author
0.1	11/03/2024	Initial draft	Milos Jovanovik
0.2	18/04/2024	Extended draft	Mirko Spasić
0.3	22/04/2024	Experiment results	Mirko Spasić
0.4	26/04/2024	Issued for review	Milos Jovanovik, Mirko Spasić
0.5	29/04/2024	Review	Muhammad Saleem
1.0	30/04/2024	Final approval and submission	Milos Jovanovik, Mirko Spasić

Author List

Organization	Name	Contact Information
OpenLink Software	Mirko Spasić	mspasic@openlinksw.com
OpenLink Software	Milos Jovanovik	mjovanovik@openlinksw.com
OpenLink Software	Hugh Williams	hwilliams@openlinksw.com
University of Paderborn	Muhammad Saleem	saleem@informatik.uni-leipzig.de

Contents

1 Introduction

Deliverable D5.2 "Report on 3DFed Evaluation based on LinkedGeoData and DBpedia Use Cases" aims to provide an overview and report on the results of the evaluation of the 3DFed tools and algorithms in the context of the large RDF knowledge graphs selected by OpenLink. The tools and algorithms tested include: the CostFed tool[1], a pure and index-based federation engine for federated SPARQL query processing over multiple SPARQL endpoints [5], and the PCM algorithm, a workload-based (predicates co-occurrence-based) partitioning using extended Markov clustering. The deliverable D3.1 [1] and D3.2 [3] introduced this type of partitioning and showed its superiority over the other techniques in terms of better query runtime performance, number of timeout queries, overall rank score, and number of distinct sources selected. The goal of this deliverable is to verify whether these algorithms lead to performance improvements for federated SPARQL queries over large datasets in real-world scenarios, whether workload-based partitioning is better than static random partitioning, and whether dynamic partitioning improves static partitioning. For the evaluation, our team used the latest versions of the LinkedGeoData and DBpedia datasets, each with more than one billion triples and more than one million real-world queries taken from the query logs of their official endpoints.

We additionally provide an RDF dataset of metadata about the DBpedia dataset, generated by benchmarking SPARQL queries from the DBpedia logs. These metadata contain information about query execution times, requested entities and properties, query characteristics, etc., which can be used by interested parties for data (re)distribution among nodes in a cluster deployment. The DBpedia metadata is available publicly, via a SPARQL endpoint, at `https://3dfed.demo.openlinksw.com/sparql/`.

The structure of this deliverable is as follows. Section 2 presents the general approach used in both use cases, the LGD use case and the DBpedia use case. It explains the evaluated test scenario and the experiments whose final results and key metrics are presented in Section 3. Section 4 shows the details of the creation of the LSQ metadata dataset for the DBpedia knowledge graph. Section 5 concludes the report and highlights the main results of Task 5.2.

2 Evaluation setup

2.1 Datasets

To evaluate the algorithms and tools developed in WP3, related to automatic data distribution and dynamic exchange, we used the latest official versions of the LGD and DBpedia datasets, which are publicly available:

- The LGD dataset contains 1.3 billion triples and was released at the end of 2015.
- The DBpedia dataset is more recent, namely 2022-12 release, which is currently loaded and available via the official SPARQL endpoint[2] and contains 1.2 billion triples.

2.2 Query workloads

Performance was measured on a subset of queries taken from the real query logs of the official SPARQL endpoints of LinkedGeoData and DBpedia.

[1] `https://github.com/dice-group/CostFed`
[2] `https://dbpedia.org/sparql`

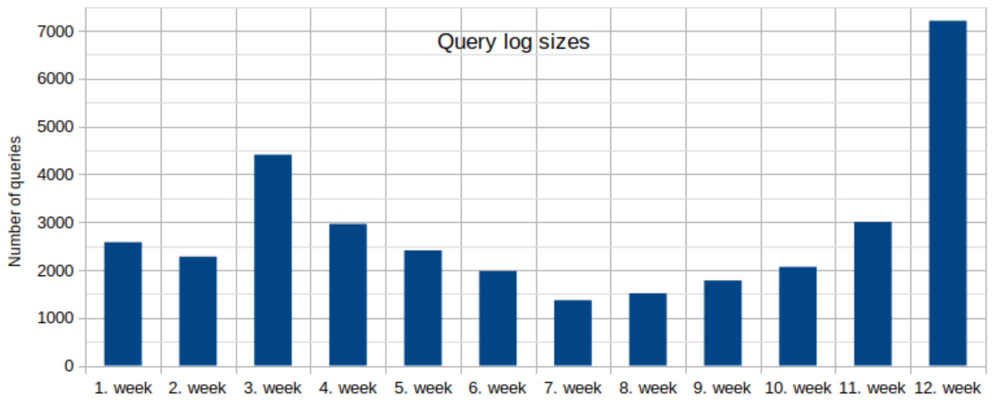

Figure 1: Number of LGD queries per week executed from March 27, 2016 to June 18, 2016.

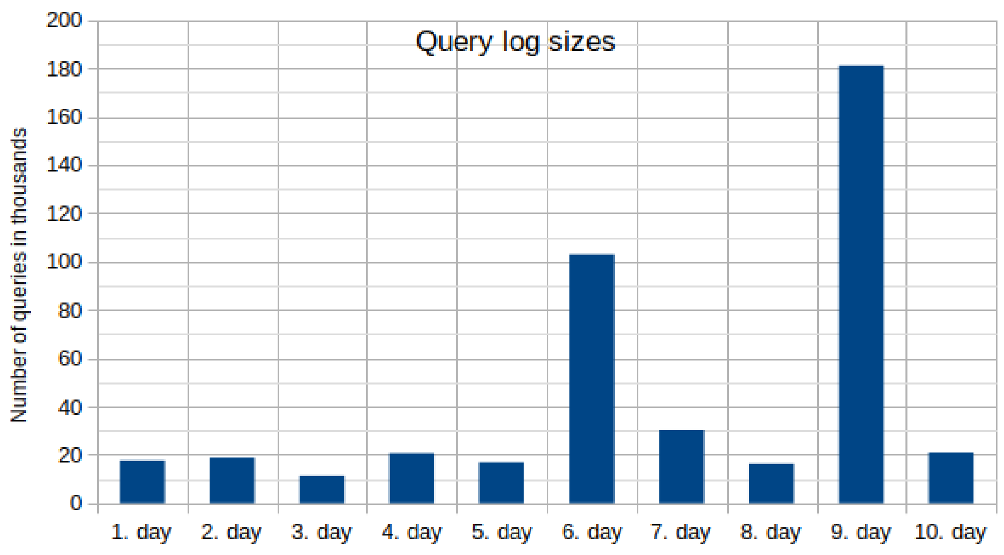

Figure 2: Number of DBpedia queries per day executed from June 1, 2023 to June 10, 2023 between 11:00 AM and 12:00 PM (CET).

- For the LGD experiments, the queries were taken from the publicly available LSQ 2.0 dataset [6], which contains all relevant information about the executions of all queries against the official LGD endpoint over a period of one year, starting from November 2015 to November 2016. For the experiments whose results are presented in Section 3.2, we used all queries from the 12-week time interval, starting at the end of March, as this interval contains the most queries compared to the other time frames. The number of queries per week can be found in Figure 1.
- The DBpedia queries come from the query logs of a publicly accessible endpoint managed by Openlink. They come from a period of 10 consecutive days (June 1 - 10, 2023), with only queries between 11:00 AM and 12:00 PM (CET) were taken into account (see Section 4). The number of queries per day can be found in Figure 2.

2.3 Summarized Approach

The evaluation approach consisted of a series of steps[3], each of them corresponding to two consecutive time frames of query workload and comprising the following four phases:

1. In the first phase of each step, the entire dataset was divided into 10 partitions using a workload-based RDF graph partitioning technique, i.e. the PCM algorithm based on all queries from the first time frame.

   In the LinkedGeoData experiments, we used weeks as time frame intervals. Therefore, the RDF graph partitioning algorithm was based on the co-occurrence of the predicates from the query workload of the first week.

   In the DBpedia experiments, we used queries in daily intervals instead of weekly intervals. The reason for this is that the DBpedia endpoint is much more popular than the LGD endpoint and its real query log contains three orders of magnitude more queries than the LGD query log. Therefore, in this case, the PCM algorithm partitioned the dataset based on the queries belonging to the first day of the selected time interval.

2. Afterwards, the obtained partitions were loaded into 10 independent Virtuoso instances, which were configured according to the obtained partitioning[4]. The version of Virtuoso used is the open source version 07.20.3233, which is available via the Docker system[5].

3. Then, the statistics (indexes) are generated for each endpoint, which contain relevant information about the distribution of data across the endpoints that the CostFed tool needs to perform appropriate queries in a distributed manner.

4. The evaluation of CostFed in terms of the average query execution time and the number of queries executed per second was performed for all queries of the following time frame (the next week in the LGD use case and the next day in the DBpedia use case). A sequence of queries from the corresponding time frame was executed twice:

   - the first time so that the Virtuoso instances had all the data in memory and to avoid disk accesses[6],
   - and the second sequence of queries represented the official evaluation.

Therefore, the distribution of the data among the virtuoso instances (data nodes) was based on the queries of the first week (LGD) or the first day (DBpedia) and then the queries of the following week (LGD) or the following day (DBpedia) were used for the evaluation.

We apply a new partitioning in each step of the experiment to obtain a dynamic partitioning that results in the data being exchanged between the partitions according to the query changes over time. As described in Deliverable 3.4 [2], the execution of queries in later steps is expected to be more efficient if data partitioning is performed in each step than if an initial static partitioning based on the query workload from the first time interval is used without any data shuffling and later data repartitioning. On the other hand, when the data is not shuffled, performance is expected to be higher when the initial static partitioning is based on the PCM algorithm than when the partitioning is not workload dependent [1, 3]. We confirm these two hypotheses in the results presented in Section 3.

[3]There were six steps for LGD use case and five of them for DBpedia.
[4]The number of buffers for a single instance is set to be able to store the whole database file in RAM.
[5]`https://hub.docker.com/r/openlink/virtuoso-opensource-7`
[6]Although all database files should already be loaded in RAM during the previous phase, i.e. the generation of the statistics.

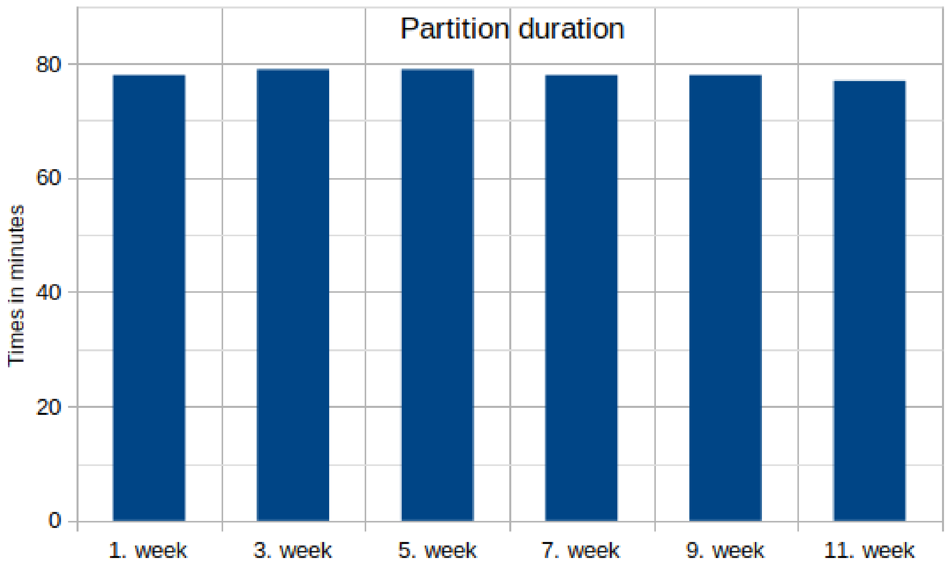

Figure 3: Duration of LGD dataset partitioning based on the real queries executed within each week.

3 Evaluation

In this section we present the results of our experiments, the structure of which was presented in the previous section. It contains the figures, the most important metrics and diagrams for the use cases LGD (Section 3.2) and DBpedia (Section 3.3). Both sections follow the same structure and present the same metrics, while the explanations of the obtained figures differ.

3.1 Hardware setup

The test platform used for all the experiments is a machine with Intel(R) Xeon(R) CPU E5-2630 with 2.30 GHz and 24 cores, the total RAM consists of 192 GB, while the operating system is CentOS Linux version 7 with kernel version 3.10.0-957.12.2.el7.x86_64.

3.2 3DFed Evaluation Using the LinkedGeoData Knowledge Graph

In this section, we present the results of our experiments regarding our LinkedGeoData use case.

The Figure 3 contains the number of minutes the PCM algorithm takes for each step of the evaluation. There are no big differences in the duration of this phase. It takes about 80 minutes and the duration depends mainly on the size of the dataset to be partitioned. The generated partitions are very unbalanced as they contain between 0 triples and almost half of the entire dataset. A measure of the imbalance of the partitioning is the Gini coefficient, whose values can vary between 0 and 1, where 0 means that the partitions are of equal size and the partitioning is very balanced, while 1 stands for a very unbalanced partitioning [3]. The Gini coefficients of the partitioning in this use case are shown in Figure 4 and are in the range of $[0.755, 0.832]$.

In the second phase of each step of the evaluation, these 10 generated partitions are loaded onto the corresponding individual Virtuoso servers. The loading times of each step are shown in Figure 5 and the number of minutes required is in the interval $[150, 172]$. Loading could be faster if the generated partitions were

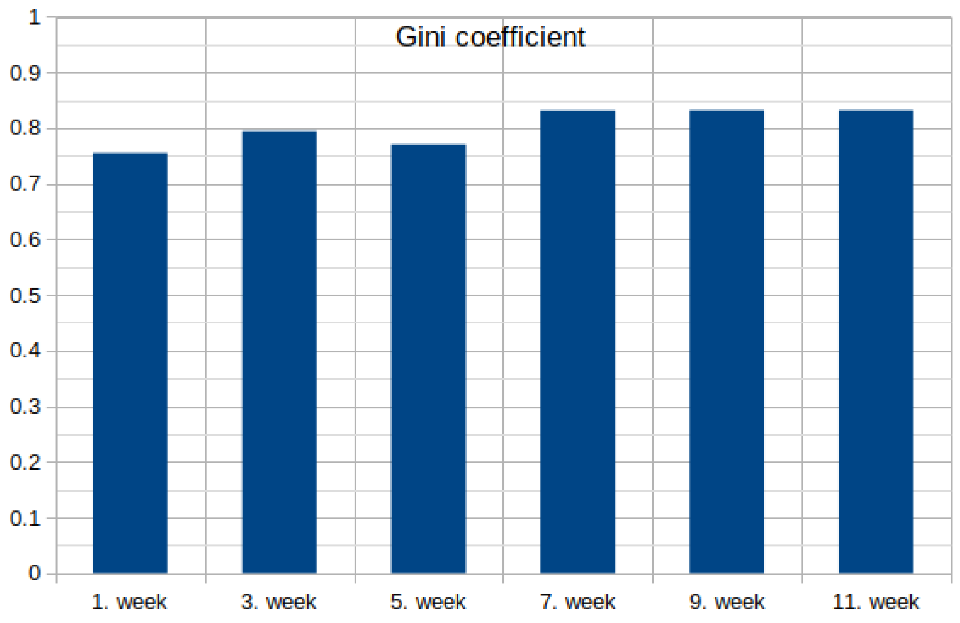

Figure 4: Gini coefficients of each partitioning of the LGD dataset.

additionally divided into several chunks. But in the case of balanced partitioning and considering that all 10 virtuoso instances were run on the same machine, this further division would not bring any improvement. The database file sizes for all virtuoso instances across all steps of the evaluation are shown in Figure 6. From this we can conclude that the largest part of the entire dataset is contained in three virtuoso instances, while the remaining seven instances contain only a very small part of the data.

In the third phase of each evaluation step, the statistics (indexes) are created based on the loaded data across 10 SPARQL endpoints provided by the virtuoso instances. This involves running many queries in parallel against all virtuoso instances, where this phase takes between 41 and 51 minutes in each step (Figure 7).

The most important metrics in the evaluation of the PCM algorithm and the CostFed tool are shown in Figure 8 and Figure 9 in terms of the average query execution time and the reverse metric, i.e. the number of queries executed per second. The trend is that the numbers are improving over time, from week to week. The average query execution time decreases from 2.2s to 0.8s when there is no data reshuffling (the red line in Figure 8) and from 1.3s to 0.6s in the case of dynamic data exchange (the blue line in Figure 8), while the number of queries executed per second increases from 0.46 to 1.21 queries per second in the case of static partitioning (the red line in Figure 9) and from 0.72 to 1.59 in the case of dynamic partitioning (the blue line in Figure 9). The number of successfully executed queries (queries that are supported by the CostFed tool[7] and are executed without a timeout limit) is the same or slightly higher for dynamic partitioning than for static partitioning.

Since the red line is almost always above the blue line (Figure 8) and below it (Figure 9), we can conclude from these two figures that the use of dynamic partitioning of data and the constant change of data distributions through the dynamic exchange of data between partitions and virtuoso instances have a good impact on the performance of query execution.

[7] In the query workload there are a lot of SPARQL queries that are not supported, e.g. DESCRIBE queries.

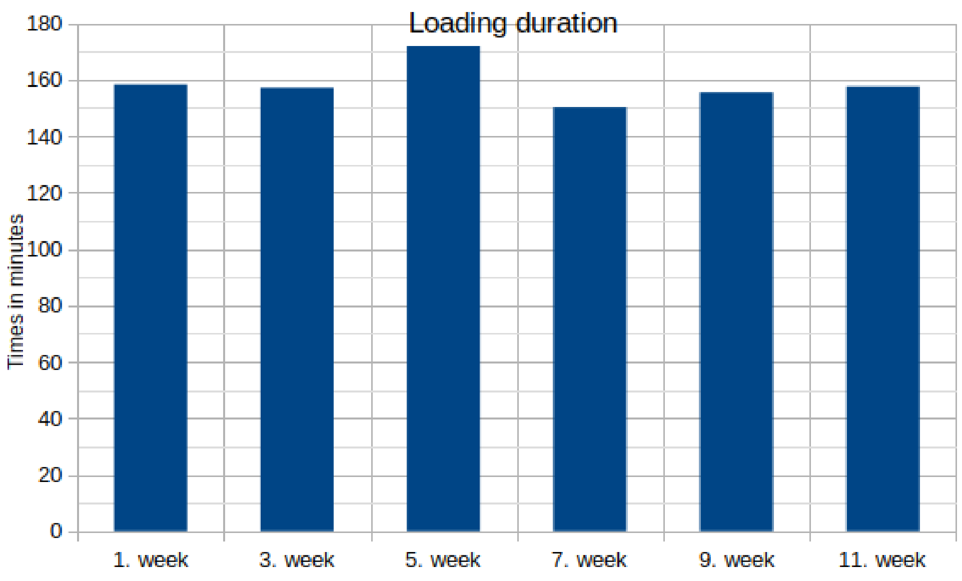

Figure 5: Total load time of the partitioned LGD dataset into 10 virtuoso instances.

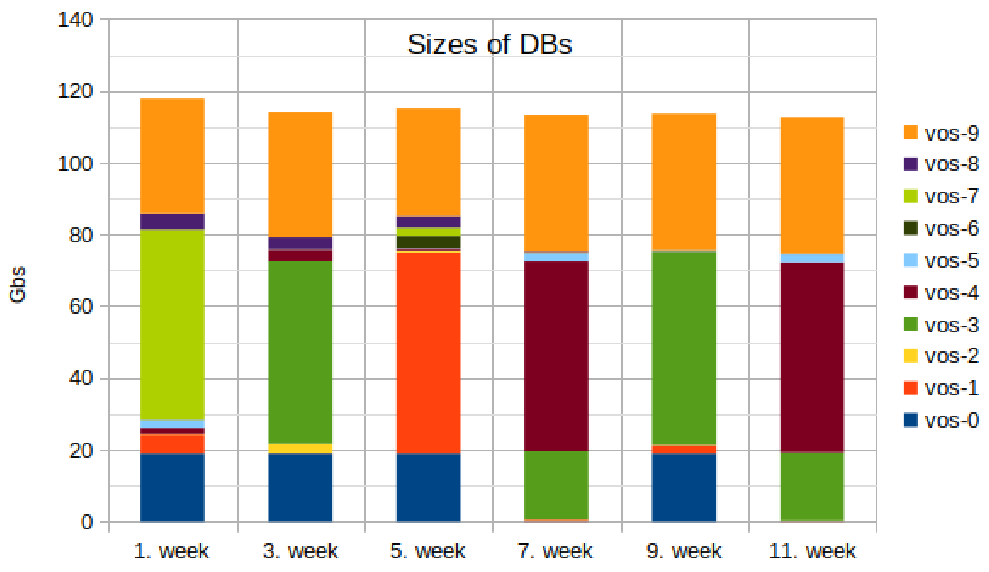

Figure 6: Sizes of the virtuoso database files containing the LGD partitions.

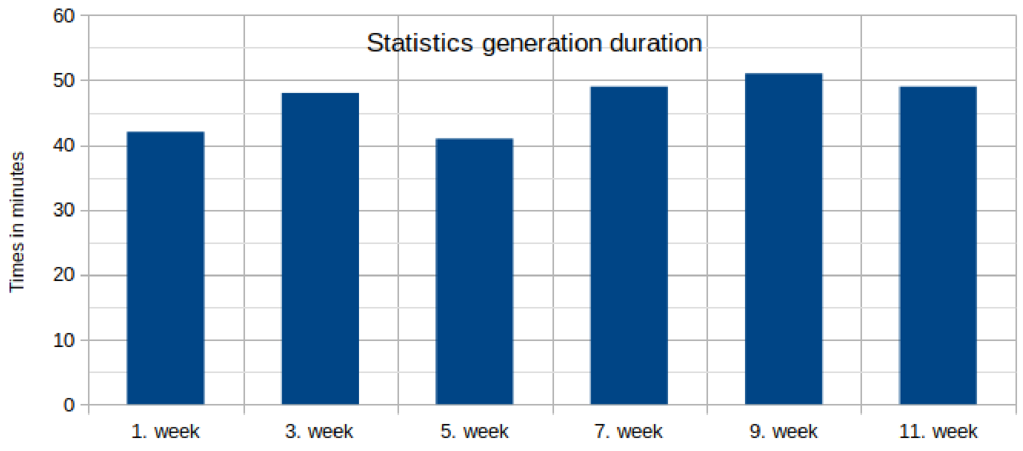

Figure 7: Duration of the statistics generation phase over 10 virtuoso instances containing the LGD partitions.

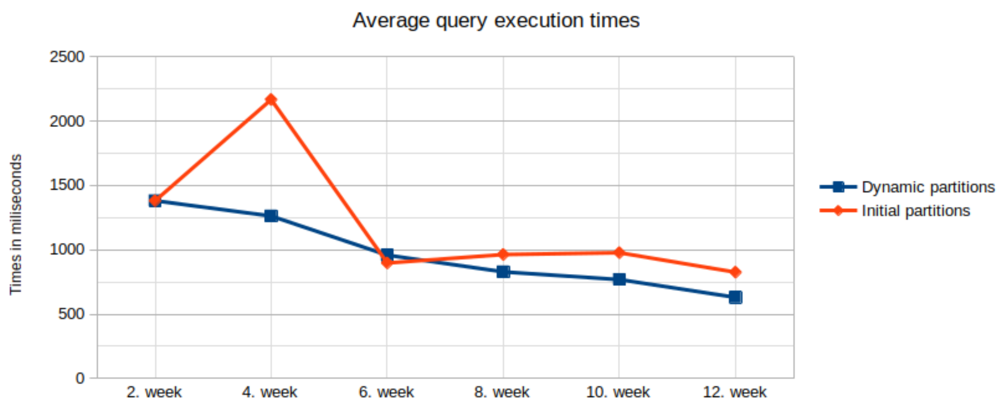

Figure 8: Average LGD query execution times over weeks using dynamic partitioning in each step and keeping the initial static partitioning from the first step of the evaluation.

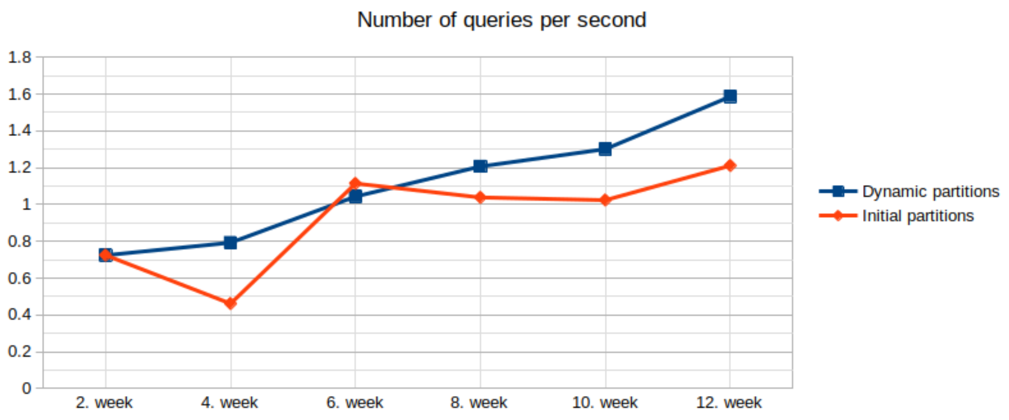

Figure 9: Number of LGD queries executed per second over weeks using dynamic partitioning in each step and keeping the initial static partitioning from the first step of the evaluation.

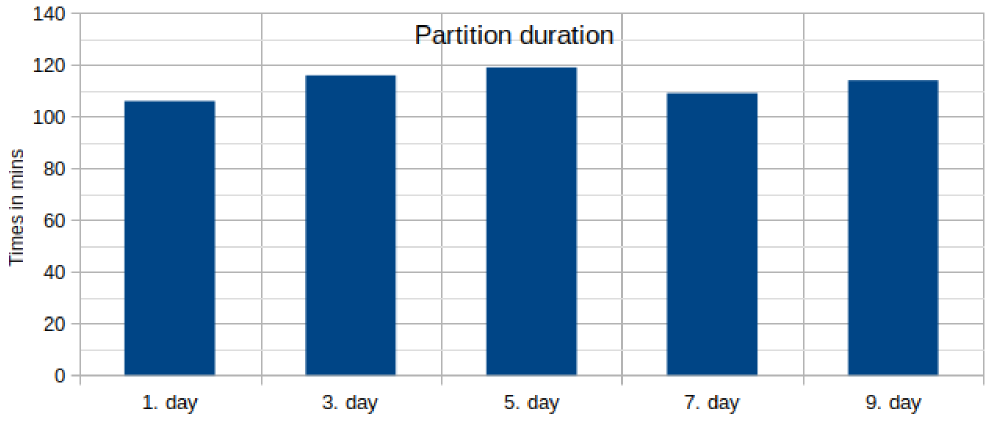

Figure 10: Duration of DBpedia dataset partitioning based on the real queries executed within each day.

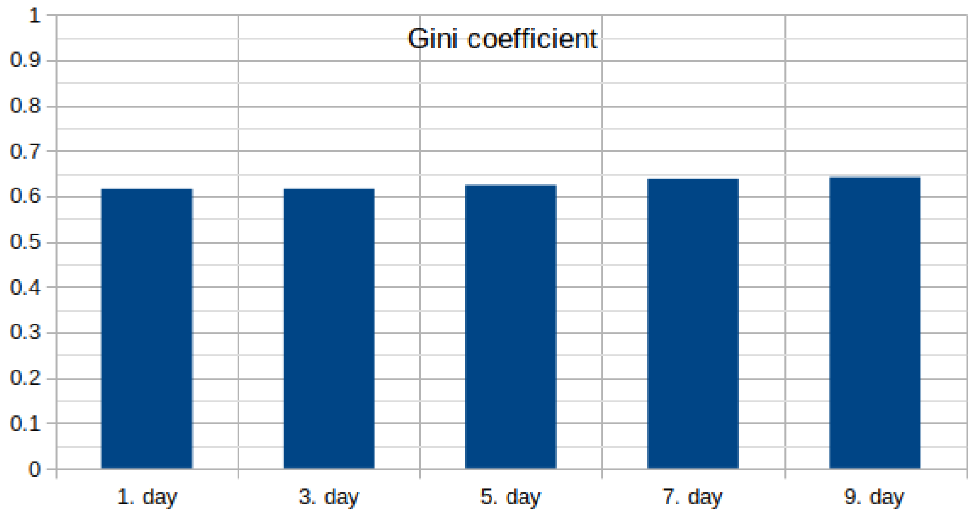

Figure 11: Gini coefficients of each partitioning of the DBpedia dataset.

3.3 3DFed Evaluation Using the DBpedia Knowledge Graph

Regarding our evaluation based on the DBpedia use case, we show the corresponding numbers, metrics and graphs already shown in Section 3.2 for the LGD use case, but using days as time intervals instead of weeks.

The PCM algorithm splits the dataset into the corresponding partitions within 106-119 minutes (Figure 10), which is slower than in the LGD use case. This is to be expected as the memory size of the DBpedia dataset is larger than the LGD one[8] (approximately 210 GB vs 190 GB). In this case, the generated partitions are more balanced, which is indicated by Gini coefficients in the range of $[0.62, 0.64]$ (Figure 11).

Loading these partitions on 10 virtuoso instances takes between 72 and 85 minutes (Figure 12) and is much faster compared to the LGD use case, mainly due to the type of the data[9]. Figure 13 shows that the load is better distributed across virtuoso instances than in the LGD use case.

The duration of statistics generation varies between 18 and 22 minutes (Figure 14), which is more than twice as fast as in the LGD case.

[8]Although there are more triples in the LGD dataset than in the DBpedia one.
[9]The LGD dataset contains a lot of geospatial data and the creation of the geo-indexes is time consuming.

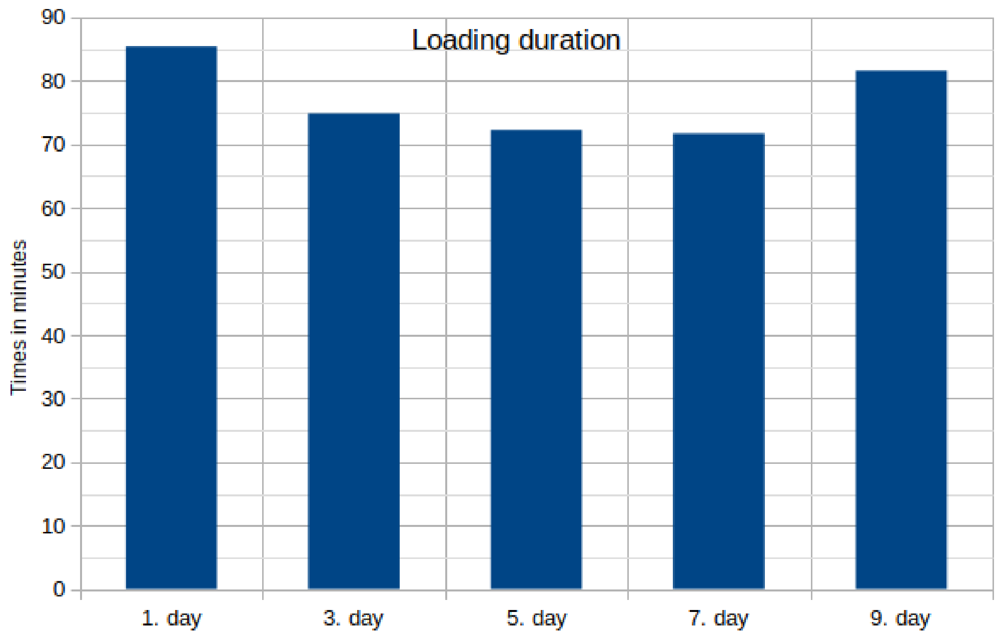

Figure 12: Total load time of the partitioned DBpedia dataset into 10 virtuoso instances.

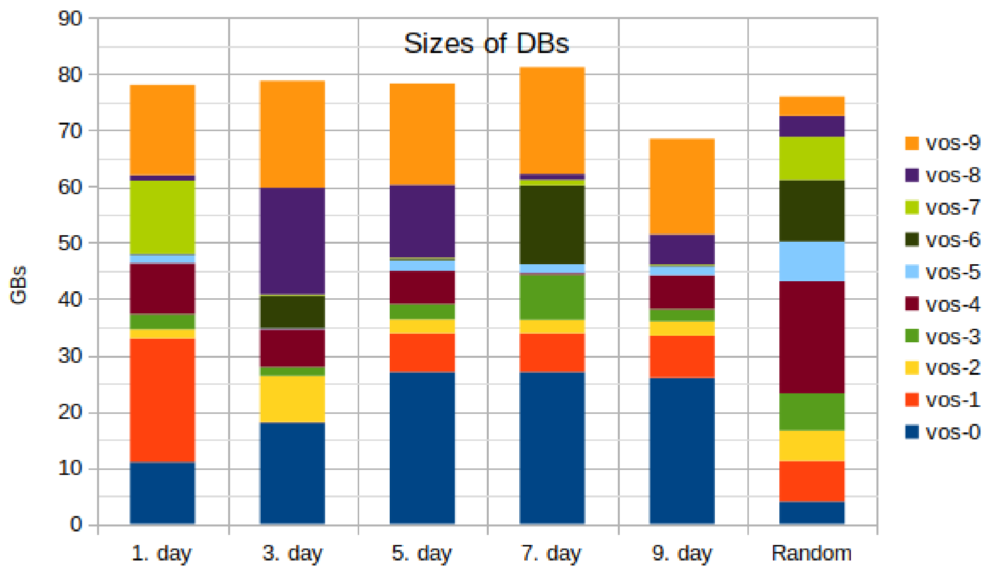

Figure 13: Sizes of the virtuoso database files containing the DBpedia partitions.

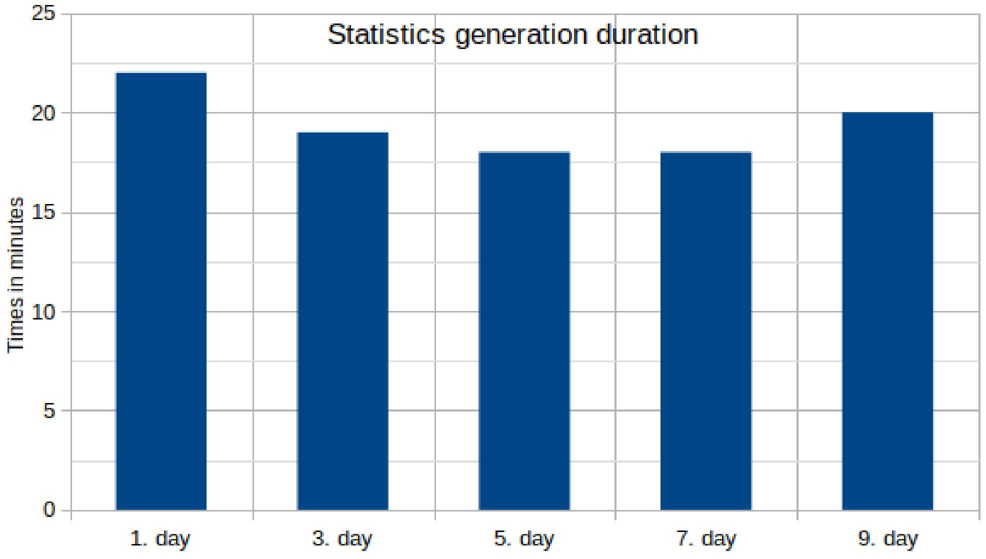

Figure 14: Duration of the statistics generation phase over 10 virtuoso instances containing the DBpedia partitions.

The most important metrics for the DBpedia use case are shown in Figure 15 and Figure 16. In contrast to the LGD use case, the average query execution time does not decrease constantly from day to day (the number of executed queries per second does not increase). The reason for this could be the different distribution of easy lookup and more sophisticated and demanding queries in the query workload from day to day[10]. The red and blue lines in these two figures correspond to those in the LGD use case, while the yellow lines represent the static, random partitioning[11], not aware of query workload. Similar to the LGD use case, the red line is always above the blue line (Figure 15), or below it (Figure 16), which proves the advantage of dynamic partitioning over static partitioning. The yellow line is always above the red line (Figure 15), or below it (Figure 16), which proves the advantage of workload-aware partitioning over random partitioning. Compared to LGD queries, the execution of DBpedia queries is on average one or two orders of magnitude faster.

4 Dataset of Benchmarked DBpedia SPARQL Queries

The dataset of benchmarked DBpedia SPARQL queries was build using the Linked SPARQL Queries (LSQ) framework [4, 6] for extracting, analyzing and benchmarking SPARQL queries. With LSQ 2.0, we were able to generate an RDF knowledge graph describing SPARQL queries extracted from the DBpedia logs, and make it publicly available at `https://3dfed.demo.openlinksw.com/sparql`.

The DBpedia logs used for this part of the task were obtained from the DBpedia team at OpenLink Software, which is responsible for maintaining and running the public DBpedia instance at `https://dbpedia.org/sparql`. We gathered anonymized logs from ten consecutive days (1st - 10th June 2023). On average, there were over 2 million SPARQL queries each day, varying from 1.3 up to 2.6 million queries per day. Given these large numbers of daily queries in the logs, we opted to use a reduced set of queries from each day in order to generate

[10]From the red and yellow lines in Figure 15 and Figure 16, which represent the main metrics when there is no data shuffling, i.e. when the partitioning is a static one that is performed only once at the beginning, we can clearly see that the queries of the last day are the most demanding on average.

[11]The dataset is divided into ten equal partitions, each containing approximately the first tenth of all triples, the second tenth, etc., respectively.

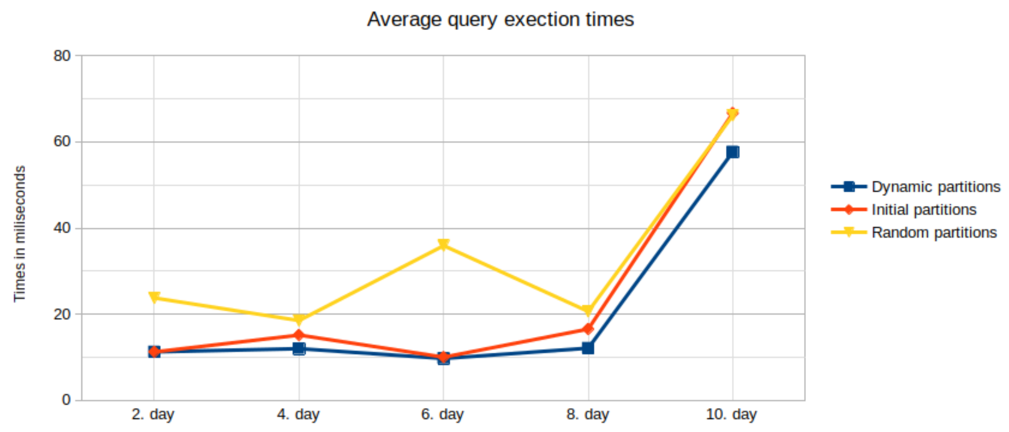

Figure 15: Average DBpedia query execution times over days using dynamic partitioning in each step and keeping the initial static partitioning from the first step of the evaluation (the workload-aware partitioning and the random one).

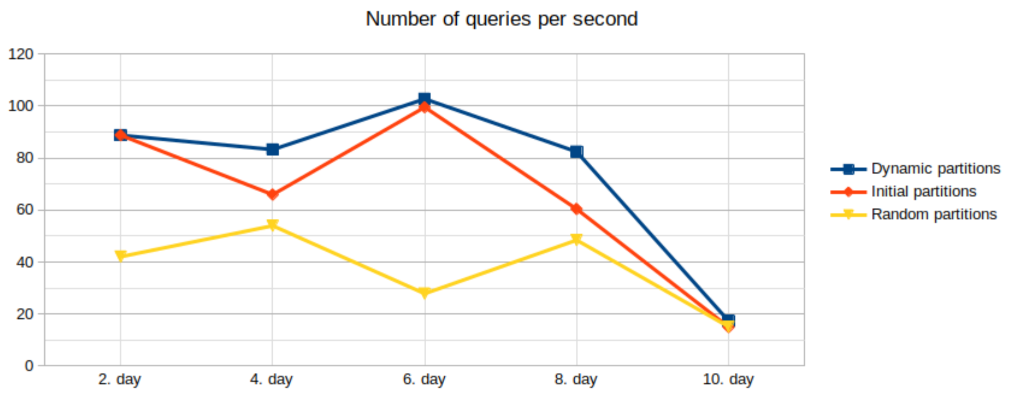

Figure 16: Number of DBpedia queries executed per second over days using dynamic partitioning in each step and keeping the initial static partitioning from the first step of the evaluation (the workload-aware partitioning and the random one).

the LSQ dataset for DBpedia. We used a one-hour window, between 11:00 AM and 12:00 PM (CET) each day, and used only the queries from this time window. This reduced the average number of queries per day to just below 130.000, varying from around 52.000 up to around 443.000 per day. This approach allowed us to have a representative slice from the DBpedia logs, and to be able to use them with the LSQ benchmarking algorithm to generate the RDF knowledge graph with query execution metadata from DBpedia.

The generated LSQ dataset contains a total of 405,601 named graphs, where each named graph represents a separate SPARQL query which has been found in the logs. Each of these queries can have multiple executions from different users, at different times. Each named graph contains metadata about the SPARQL query, i.e. the basic graph patterns of the query, used features such as functions and operators, along with statistics about these features, benchmark evaluation of the query, such as performance and result set characteristics, etc. Due to the large size of each named graph, a sample named graph for a given SPARQL query can be explored using the faceted search service of the Virtuoso instance with the dataset[12]. This allows users to drill down and explore the knowledge graph using the Virtuoso faceted search UI. For instance, the structural features of the same SPARQL query can be found at the following location[13].

The purpose of the dataset is to provide a basis for the evaluation presented in the previous sections, by providing access to the queries needed in the process.

Additionally, there's a list of useful queries which can be used on top of the generated dataset, aside from the evaluation process of the LGD and DBpedia datasets. They are published on the LSQ website[14]. They allow the selection of all queries of a certain type (SELECT, CONSTRUCT, ASK, etc.), the selection of queries with their timestamps, result set sizes and runtimes, the selection of queries returning actual results, selecting the number of triple patterns for each query, selecting the number of join vertices for each query, selecting the property paths for each query, selecting queries with specific classes as objects in the triple patterns, etc. They are very useful for exploring the logs of queries retrieved from DBpedia and analyzing their properties.

5 Conclusion

In this deliverable, we measured the fitness of the tools and algorithms developed within WP3 of this project (i.e. the CostFed tool for federated querying over multiple endpoints and the PCM algorithm for workload-aware RDF graph partitioning) for use in real-world application scenarios, including large datasets and query workloads from LinkedGeoData and DBpedia. In the experiments we performed, the datasets were the latest releases of LGD and DBpedia, each with more than one billion triples. The total number of queries evaluated, taken from the real query logs of the official SPARQL endpoints, is more than one million. The key metrics used in the evaluation were the average query execution time and the number of queries executed per second. The results clearly show a significant improvement of these two metrics when a workload-aware partitioning, i.e. the PCM partitioning algorithm, is used instead of a random partitioning of an RDF graph, as well as the superiority of a dynamic partitioning over a static one. This was also examined and proven using smaller datasets and less demanding query workloads within the tasks in WP3.

The query workload being used in the evaluation of the DBpedia use case is provided as an LSQ metadata dataset. It contains all the information about the execution of the queries against the official DBpedia SPARQL endpoint and can be used by third parties to (re)distribute the data to the nodes in a cluster deployment.

[12]https://3dfed.demo.openlinksw.com/describe/?url=http%3A%2F%2Flsq.aksw.org%2FlsqQuery--1_c-T7VsZ5YTHAOG48GFOOnHXNLQ05rrvLkmkSFga8.AA.AA.s.NY6jkA.AA.AA.AA.AA.AA.7xrpiwAA.AA.AA.AA

[13]https://3dfed.demo.openlinksw.com/describe/?url=http%3A%2F%2Flsq.aksw.org%2FlsqQuery--1_c-T7VsZ5YTHAOG48GFOOnHXNLQ05rrvLkmkSFga8.AA.AA.s.NY6jkA.AA.AA.AA.AA.AA.7xrpiwAA.AA.AA.AA-sf

[14]http://lsq.aksw.org/v2/usage/useful-queries.html

References

[1] Adnan Akhter, Muhammad Saleem, Alexander Bigerl, Axel-Cyrille Ngonga Ngomo, Mohammad Sajjadi, Milos Jovanovik, and Mirko Spasić. Initial Report on the Automatic Data Distribution. `https://www.3dfed.com/wp-content/uploads/2023/01/3DFed_deliverable_D3_1__Rev_1_1_.pdf`, December 2022. Project 3DFed Deliverable 3.1.

[2] Asal Alikhani, Muhammad Saleem, Mohammad Sajjadi, and Milos Jovanovik. Final Report on the Dynamic Data Exchange. `https://www.3dfed.com/wp-content/uploads/2023/12/3DFed_deliverable_D3_4.pdf`, September 2023. Project 3DFed Deliverable 3.4.

[3] Asal Alikhani, Muhammad Saleem, Martin Voigt, Axel-Cyrille Ngonga Ngomo, Mohammad Sajjadi, and Mirko Spasić. Final Report on the Automatic Data Distribution. `https://www.3dfed.com/wp-content/uploads/2023/01/3DFed_deliverable_D3_2.pdf`, January 2023. Project 3DFed Deliverable 3.2.

[4] Muhammad Saleem, Muhammad Intizar Ali, Aidan Hogan, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. LSQ: The Linked SPARQL Queries Dataset. In *International Semantic Web Conference (ISWC)*, volume 9367 of *LNCS*, pages 261–269. Springer, 2015.

[5] Muhammad Saleem, Alexander Potocki, Tommaso Soru, Olaf Hartig, and Axel-Cyrille Ngonga Ngomo. Costfed: Cost-based query optimization for sparql endpoint federation. *Procedia Computer Science*, 137:163–174, 2018. Proceedings of the 14th International Conference on Semantic Systems 10th – 13th of September 2018 Vienna, Austria.

[6] Claus Stadler, Muhammad Saleem, Qaiser Mehmood, Carlos Buil-Aranda, Michel Dumontier, Aidan Hogan, and Axel Cyrille Ngonga Ngomo. LSQ 2.0: A Linked Dataset of SPARQL Query Logs. *Semantic Web*, 15(1):167–189, 2024.