# Artifact Evaluation Phase-1: Artifacts Available

This repository includes the artifacts of the paper “Obliviator: OBLIVIous Parallel Joins and other OperATORs in Shared Memory Environments”. This document is for the artifact evaluation phase-1, which targets artifact availability. For artifact functionality in phase-2, please refer to the second document ./document_2_functionality.pdf.

We first claim that our work is in full accordance with the USENIX'25 ethics guidelines. We propose new algorithms and implement systems with a positive impact on preserving data privacy. Our experiments involved neither testing on live systems without prior consent, nor human participants.

All our tests were executed either on synthetic datasets whose creation we describe or on already publicly available real-world datasets. They include TPC-H, a synthesized benchmark with created contents, a twitter social graph that is available to the public and contains the anonymized topology of the Twitter social network (also used in [1, 2, 3]), a public IMDb dataset that contains the public information of title names and actors (used in [4, 5]), a public Amazon dataset that records frequently co-purchased products (used in [5, 6]), a joke dataset that contains anonymous ratings of jokes by different users (used in [5, 7]), and slashdot dataset that contains technology news website with friend/foe links between users (used in [5, 8]). We would like to point out that none of these benchmarks/datasets can cause any type of harm and are strictly used to evaluate our algorithms.

Additionally, we open-source all artifacts required for recreating our algorithms and experiments. They include all our code in this paper, scripts to generate the synthesized dataset, scripts to process public benchmarks and datasets, configuration information, and scripts to reproduce our evaluation.

The file structure of the repository is shown below.

```
.
├── data/
│   ├── big_data_benchmark
│   ├── synthesized_dataset
│   ├── TPCH
│   ├── real_world_jokes
│   ├── real_world_IMDb
│   ├── real_world_slashdot
│   ├── real_world_twitter
│   └── real_world_amazon
├── fk_join
├── join
├── join_kks
├── opaque_shared_memory/
│   ├── fk_join
│   ├── operator_1
│   ├── operator_2
│   └── operator_3
├── operator_1
├── operator_2
├── operator_3
├── scripts
├── tpch_q3
├── tpch_q5
└── tpch_q6
```

It includes all artifacts and information required to recreate the algorithms and experiments in the paper. To elaborate, it includes the following artifacts (their corresponding paths are given in their following parentheses).

1. Code:
    a. Obliviator: non-foreign key join (./join), foreign key join (./fk_join), big data benchmark (BDB) query 1-3 [9] (./operator_1, ./operator_2, ./operator_3)
    b. shared-memory Opaque: BDB query 1-3 (./opaque_shared_memory/operator_1, ./opaque_shared_memory/operator_2, ./opaque_shared_memory/operator_3), foreign key join (./opaque_shared_memory/fk_join)
    c. improved KKS (./join_kks)
2. Ready-to-use datasets:
    a. BDB dataset (./data/big_data_benchmark)
    b. Slashdot dataset [10] (./data/real_world_slashdot)
    c. Jester joke dataset [7] (./data/real_world_jokes)
    d. Amazon co-purchasing dataset [11] (./data/real_world_amazon)
3. Scripts for datasets:
    a. Scripts to synthesize dataset according to the method described in [12] (./data/synthesized_dataset/generate_1, ./data/synthesized_dataset/generate_2, ./data/synthesized_dataset/generate_3.py)
    b. Scripts to download and pre-process TPC-H [13] (./Parallel-join/data/TPCH/figure10_tpch_1.py, ./Parallel-join/data/TPCH/figure10_tpch_2.py, ./Parallel-join/data/TPCH/figure10_tpch_3.py, /Parallel-join/data/TPCH/figure10_tpch_4.py, ./Parallel-join/data/TPCH/figure11_q3.py, ./Parallel-join/data/TPCH/figure11_q5.py, ./Parallel-join/data/TPCH/figure11_q6.py)
    c. Scripts to download and pre-process Twitter social graph dataset [1] (./data/real_world_twitter/download.sh, ./data/real_world_twitter/twitter_data_process_1.cpp, ./data/real_world_twitter/twitter_data_process_2.py)
    d. Scripts to download and pre-process IMDb dataset [14] (./data/real_world_IMDb/download.sh, ./data/real_world_IMDb/imdb_process_1.py, ./data/real_world_IMDb/imdb_process_2.py)
4. Configuration information of the virtual machine (describe in the second document ./document_2_functionality.pdf) and the hardware enclave (*parallel.conf, e.g., ./join/enclave/parallel.conf, ./fk_join/enclave/parallel.conf) that we used
5. Scripts to run the experiments in our paper:
    a. Figure 9: Obliviator and KKS on synthesized datasets (./scripts/figure9.sh)
    b. Figure 10: Obliviator and KKS on TPC-H (./scripts/figure10.sh)
    c. Figure 11: Obliviator and shared-memory Opaque on TPC-H and BDB (./scripts/figure11.sh)
    d. Table 1: Obliviator and KKS on the four real-world datasets (./scripts/table2.sh)
    e. Table 2: Obliviator and shared-memory Opaque on real-world IMDb dataset (./scripts/table3.sh)
6. Adequate instructions and guide to execute the codes described in the second document ./document_2_functionality.pdf.

Note that, due to the large size of datasets listed in 3, we provide the scripts to generate or download them, and also share scripts to process their raw files (e.g., unzip, transforming to .txt file). For the original KKS code, we do not include it here since it is open-sourced by [12] at [15].

The repository is a complete set of all artifacts and information required to reproduce algorithms and experiments in the paper, and no necessary information or components is missed. For further instructions on functionality evaluation, please refer to the second document ./document_2_functionality.pdf.

Please feel free to let us know if there are any issues with the evaluation.

Reference

[1] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and Krishna P. Gummadi. Measuring User Influence in Twitter: The Million Follower Fallacy. In Proceedings of the 4th International AAAI Conference on Weblogs and Social Media (ICWSM).
[2] Zhao Chang, Dong Xie, Sheng Wang, and Feifei Li. Towards practical oblivious join. In Proceedings of the 2022 International Conference on Management of Data. Association for Computing Machinery, 2022.
[3] Xiang Li, Nuozhou Sun, Yunqian Luo, and Mingyu Gao. Soda: A set of fast oblivious algorithms in distributed secure data analytics. Proceedings of the VLDB Endowment, 16(7):1671–1684, 2023.
[4] Kevin Lewi and David J Wu. Order-revealing encryption: New constructions, applications, and lower bounds. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pages 1167–1178, 2016.
[5] Shuyuan Li, Yuxiang Zeng, Yuxiang Wang, Yiman Zhong, Zimu Zhou, and Yongxin Tong. An experimental study on federated equi-joins. IEEE Transactions on Knowledge and Data Engineering, 2024.
[6] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. In Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, pages 1–8, 2012.
[7] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. information retrieval, 4:133–151, 2001.
[8] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Signed networks in social media. In Proceedings of the SIGCHI conference on human factors in computing systems, pages 1361–1370, 2010.
[9] AMPLab, University of California, Berkley. Big data benchmark. https://amplab.cs.berkeley.edu/benchmark/, 2014.
[10] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. Internet Mathematics, pages 29-123, 2009.
[11] Jure Leskovec, Lada A. Adamic, and Bernardo A. Huberman. The Dynamics of Viral Marketing. ACM Transactions on the Web (ACM TWEB). 2007.
[12] Simeon Krastnikov, Florian Kerschbaum, and Douglas Stebila. Efficient oblivious database joins. VLDB, 2020.
[13] Transaction Processing Performance Council. TPC Benchmark H (TPC-H). http://www.tpc.org/tpch/, 1992.
[14] IMDb Non-Commercial Datasets. https://developer.imdb.com/non-commercial-datasets/, 2024.
[15] Oblivious Database Join Algorithm. https://git.uwaterloo.ca/skrastni/obliv-join-impl, 2024.