DBKDA 2024

The Sixteenth International Conference on Advances in Databases, Knowledge, and Data Applications

ISBN: 978-1-68558-138-1

March 10th –14th, 2024

Athens, Greece

DBKDA 2024 Editors

Andreas Schmidt, University of Applied Sciences Karlsruhe, Germany

Erik Buchmann, Universität Leipzig, Germany

DBKDA 2024

Foreword

The Sisteenth International Conference on Advances in Databases, Knowledge, and Data Applications (DBKDA 2024), held between March 10 – 14, 2024, continued a series of international events covering a large spectrum of topics related to advances in fundamentals on databases, evolution of relation between databases and other domains, data base technologies and content processing, as well as specifics in applications domains databases.

Advances in different technologies and domains related to databases triggered substantial improvements for content processing, information indexing, and data, process and knowledge mining. The push came from Web services, artificial intelligence, and agent technologies, as well as from the generalization of the XML adoption.

High-speed communications and computations, large storage capacities, and load-balancing for distributed databases access allow new approaches for content processing with incomplete patterns, advanced ranking algorithms and advanced indexing methods.

Evolution on e-business, ehealth and telemedicine, bioinformatics, finance and marketing, geographical positioning systems put pressure on database communities to push the 'de facto' methods to support new requirements in terms of scalability, privacy, performance, indexing, and heterogeneity of both content and technology.

We take here the opportunity to warmly thank all the members of the DBKDA 2024 Technical Program Committee, as well as the numerous reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to DBKDA 2024. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the DBKDA 2024 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that DBKDA 2024 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the fields of databases, knowledge and data applications.

We are convinced that the participants found the event useful and communications very open. We also hope that Athens provided a pleasant environment during the conference and everyone saved some time for exploring this beautiful city

DBKDA 2024 Chairs:

DBKDA 2024 Steering Committee
Fritz Laux, Reutlingen University, Germany
Andreas Schmidt, Karlsruhe Institute of Technology / University of Applied Sciences
Erik Hoel, Esri, USA
Lisa Ehrlinger, Software Competence Center Hagenberg GmbH, Austria
Peter Kieseberg, St. Pölten University of Applied Sciences, Austria
Erik Hoel, Esri, USA

DBKDA 2024 Publicity Chairs
Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain
José Miguel Jiménez, Universitat Politecnica de Valencia, Spain

DBKDA 2024

Committee

DBKDA 2024 Steering Committee
Fritz Laux, Reutlingen University, Germany
Andreas Schmidt, Karlsruhe Institute of Technology / University of Applied Sciences
Erik Hoel, Esri, USA
Lisa Ehrlinger, Software Competence Center Hagenberg GmbH, Austria
Peter Kieseberg, St. Pölten University of Applied Sciences, Austria
Erik Hoel, Esri, USA

DBKDA 2024 Publicity Chairs
Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain
José Miguel Jiménez, Universitat Politecnica de Valencia, Spain

DBKDA 2024 Technical Program Committee

Taher Omran Ahmed, University of Technology and Applied Sciences, Ibri, Oman / Alzintan University, Libya
Julien Aligon, Institut de Recherche en Informatique de Toulouse (IRIT) | Université Toulouse 1 Capitole, France
Alaa Alomoush, University Malaysia Pahang, Malaysia
Emmanuel Andres, Hôpitaux Universitaires de Strasbourg, France
Vincenzo Arceri, Università degli Studi di Parma, Italy
Zeyar Aung, Masdar Institute of Science and Technology, UAE
Qiushi Bai, Microsoft, USA
Aruna Bansal, IBM India Pvt. Ltd., India
Christian Beecks, University of Hagen, Germany
Jam Jahanzeb Khan Behan, Université libre de Bruxelles (ULB), Belgium / Universidad Politécnica de Cataluña (UPC), Spain
Flavio Bertini, University of Parma, Italy
Vincenzo Bonnici, University of Parma, Italy
Savong Bou, University of Tsukuba, Japan
Ali Boukehila, University of Annaba, Algeria
Zouhaier Brahmia, University of Sfax, Tunisia
Martine Cadot, LORIA, Nancy, France
Alessandro Castelnovo, Intesa Sanpaolo S.P.A / University of Milano Bicocca, Italy
Basabi Chakraborty, Iwate Prefectural University, Japan / Madanapalle Institute of Technology and Science, India
Sanjay Chaudhary, Ahmedabad University, India
Yung Chang Chi, National Cheng Kung University, Taiwan
Jong Choi, Oak Ridge National Laboratory, USA
Stefano Cirillo, University of Salerno, Italy
Miguel Couceiro, LORIA, France
Malcolm Crowe, University of the West of Scotland, UK

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

MosaicDB: An Efficient Trusted / Untrusted Memory Management for Location Data in Database

Tomoya Suzuki*, Taisho Sasada†, Yuzo Taenaka‡ and Youki Kadobayashi§
Division of Information Science, Nara Institute of Science and Technology

Email: *suzuki.tomoya.sp9@is.naist.jp, †sasada.taisho.su0@is.naist.jp, ‡yuzo@is.naist.jp, §youki-k@is.naist.jp

***Abstract*—Location data has been used for various purposes in digitized society but includes waypoints directly related to personal privacy, such as home addresses. To hide such sensitive waypoints, some applications provide Endpoint Privacy Zones (EPZs) that keep a portion of the track secret. However, most service providers placing databases on a cloud face the potential risk of exposing sensitive waypoints to cloud service providers. Existing studies have proposed databases using Trusted Execution Environments (TEE) that protects sensitive data in a trusted and completely secure memory region. However, as TEE inevitably limits the size of the trusted memory, databases proposed in these studies cannot use the trusted memory efficiently due to the fundamental design that handles all data in the trusted memory. Moreover, the memory outside of the trusted memory, called untrusted memory, remains vacant even if the trusted memory is fully used, thereby leading to insufficient memory utilization on a whole system and decreased database performance. In this study, we propose MosaicDB, a memory-efficient and trusted database for location data, using both trusted and untrusted memory. To enhance memory utilization efficiency, MosaicDB handles only sensitive waypoints within the EPZ in the trusted memory while handling non-sensitive waypoints in the untrusted memory. Experimental results show that MosaicDB improves memory utilization efficiency, thereby achieving a 25% reduction in execution time for selection queries compared to the database that handles all data in the trusted memory.**

Keywords*—*Database; Trusted Execution Environment; Intel SGX; Location data; Endpoint Privacy Zones; Cloud Computing.

I. Introduction

In recent years, the popularity of Fitness Tracking Social Networks (FTSNs) has expanded as the number of health-conscious people has increased. FTSNs allow users to track outdoor activities and share their routes with other users. By sharing routes, users can enhance the enjoyment of their activities and maintain their motivation. However, sharing routes also raises privacy concerns, as other users can browse routes that often include sensitive waypoints, such as homes or workplaces.

FTSNs enable users to designate Endpoint Privacy Zones (EPZs) to prevent privacy leakage from sharing routes. An EPZ allows users to hide some routes, as shown in Figure 1. Ongoing research is also being conducted to facilitate the establishment of more robust EPZs [1][2]. That is, sensitive waypoints are protected on an application. However, waypoints on a database still suffer from an exposure risk in a cloud environment. As modern application services, including database systems, are deployed in a cloud environment, waypoints in the database may be stolen by the Cloud Service Provider (CSP) with the highest privileges on the cloud system. Although existing databases offer disk-level encryption to protect sensitive data, a CSP may steal unencrypted data or encryption keys directly from memory, leading to significant privacy leakage, as shown in Figure 2.

To overcome these threats, databases using Trusted Execution Environments (TEE) have been proposed [3][4][5][6]. TEE creates a trusted region in memory using hardware-level security mechanisms. Any privileged software, such as an operating system and hypervisor, cannot directly read and write the confidential data managed by a database since the trusted memory is completely isolated from the main (untrusted) memory. However, as TEE severely restricts the size of the trusted memory, databases proposed in these studies are now facing a challenge of performance degradation due to a shortage of the trusted memory. This challenge arises from their fundamental design of handling all data in the trusted memory.

In this study, we propose MosaicDB, a memory-efficient and trusted database for location data. MosaicDB selectively handles location data (waypoints) in the trusted memory, following the necessity of data protection in the application context. As an application conceals all waypoints within EPZs and exposes the remaining waypoints, we only need to protect waypoints within EPZs. Moreover, the number of waypoints within EPZs is less than that outside the EPZs. We utilize this characteristics of location data and thus attempt to efficiently use both trusted and untrusted memory. That is, MosaicDB only protects waypoints within EPZs in the trusted memory so that they are not exposed. As other waypoints outside EPZs are already public on an application, MosaicDB handles them without any special treatment in the untrusted memory. This data management method based on application context allows us to synchronously protect sensitive data that users do not want to be exposed in the database, and to effectively utilize both trusted and untrusted memory on the database server.

The structure of this paper is as follows: In Section II, we provide an overview of the fundamental features of Intel Software Guard Extensions (SGX), TEE employed in this study. In Section III, we outline related works on databases utilizing TEE and their challenges. In Section IV, we explore the detailed design of MosaicDB and query processing flow. In Section V, we present the results of the experimental evaluation. In Section VI, we describe the limitations of MosaicDB and outline future directions. Finally, in Section VII, we conclude the paper by summarizing the contributions of this research.

 ISBN: 978-1-68558-138-1

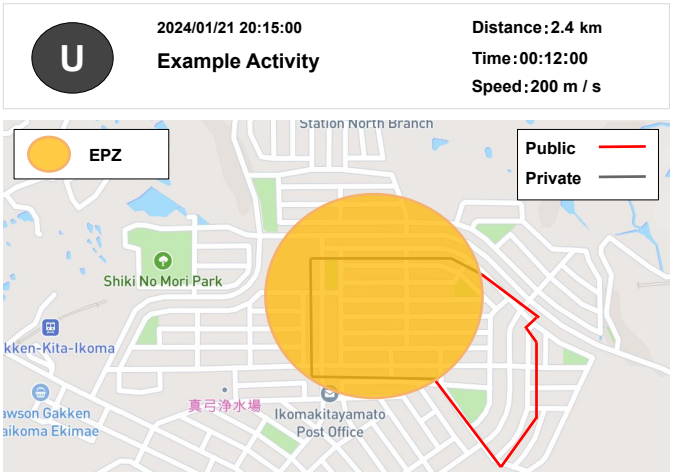

Figure 1. Example of EPZ

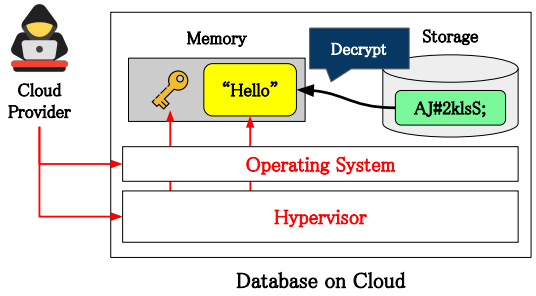

Figure 2. Threat of data theft by malicious cloud providers

II. Intel SGX

We protect the database using Intel SGX, the most widely used TEE in cloud environments. SGX is installed in 6th generation Intel CPUs and later, which creates a trusted region (known as enclaves) in memory. SGX achieves the confidentiality and integrity of sensitive processes using the encrypted and isolated physical memory known as the enclave page cache (EPC). CPUs fully control access to the enclave, thus preventing attacks on processes from privileged software, such as the operating system or hypervisor. This powerful protection provided by SGX enables cloud users to safely execute processes in untrusted cloud environments. The following subsections describe a limitation of SGX, the sealing feature for secure data persistence, and the remote attestation feature for secure communication between a database and an application server.

A. Hardware limitation of the enclave

One of the major limitations of Intel SGX is the size of enclave memory. Intel CPUs up to 10th generation support a maximum enclave memory size of 128 MB to 256 MB, while Xeon scalable processors from the 3rd generation and later support a maximum enclave memory size of 512 GB. When the enclave memory footprint exceeds this size limitation, it leads to highly inefficient EPC paging, significantly decreasing the performance of processes runnning in enclaves. Although the available enclave memory size has increased in 3rd generation and later Xeon scalable processors, the available enclave memory size will be smaller depending on the system configuration. For example, in Microsoft Azure's DCsv3 series, only 16 GB of enclave memory is available for a virtual machine with a total of 32 GB memory. Therefore, if the database uses only enclave memory, the non-enclave memory cannot be used, resulting in inefficient memory utilization.

B. Data persistence with sealing

Since the enclave is a memory region allocated to a process, the data in the enclave will be lost when the process is stopped. To address this, SGX provides sealing and unsealing functions to persist data in storage securely. These functions enable the encryption and decryption of protected data using an enclave-specific key. However, cryptographic operations in sealing incur significant overhead, requiring SGX applications to minimize the frequency of sealing operations whenever possible. In this study, we minimize this overhead by exclusively protecting location data within the EPZ through sealing.

C. Authentication and key exchange with remote attestation

Remote attestation [7] is an authentication protocol that mutually verifies the integrity of enclaves between SGX applications. It verifies the integrity of enclaves with which it communicates and exchanges the key used to encrypt the communication content, thereby ensuring secure communication between SGX applications. Remote attestation is available only from codes in the enclave. This study uses this protocol to secure communication between the application servers and databases.

III. Related work

Several databases using SGX have been proposed [3][4][5][6]. CryptSQLite [3] protects all the data in the enclave. Its design is straightforward; however, it is suitable only for cases involving small-scale data because the memory load in the enclave increases as the data size increases.

EnclaveDB [4] uses SGX to protect tables managed in memory. The simple design of protecting the entire table makes it easy to integrate into existing RDBMSs; however, this consumes a significant amount of the enclave memory while making it impractical to utilize untrusted memory efficiently.

StealthDB [5] addressed the problem of excessive enclave memory usage by protecting only primitive operators (e.g., $\geqq, \leq, +, *$) that process unencrypted data in the enclave. The amount of memory used by the operations remains almost constant, minimizing the utilization of the enclave memory in all database query processing and acheiving high memory utilization efficiency. However, an increase in the transitions between the enclave and untrusted memory significantly degrades database performance.

Yoshimura et al. [6] proposed an RDBMS designed to reduce enclave memory usage and transitions between the enclave and untrusted memory by protecting only specific columns in the enclave. This approach boosts the memory utilization

 ISBN: 978-1-68558-138-1

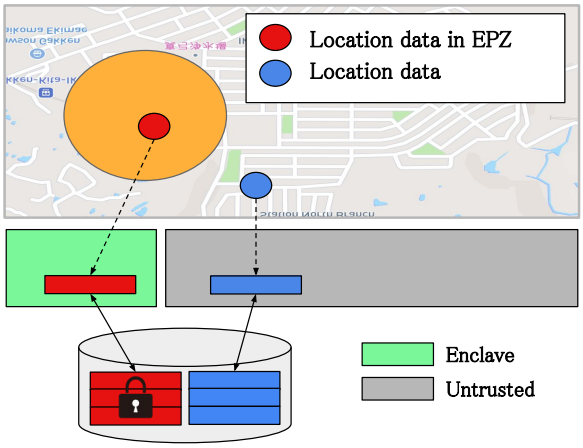

Figure 3. Concept of MosaicDB

efficiency as long as there are few columns under protection. However, when dealing with location data that includes columns such as latitude, longitude, and time, nearly all columns require protection, which can decrease the memory utilization efficiency. This method also does not allow selective protection of specific data (row) based on application context. This leads to unnecessary protection of non-sensitive data in the specific application context.

Based on the above considerations, we propose a memory-efficient and trusted database that uses both enclave and untrusted memory, focusing on location data.

IV. MosaicDB

In this section, we describe our threat model, fundamental design, and detailed query execution flow in MosaicDB.

A. Threat model

To design the architecture of MosaicDB, we define our threat model. Our threat model assumes a CSP as an main adversary able to access all the software (such as an operating system and hypervisor) and hardware except for enclaves. A CSP can continuously access the database's memory using memory dump or cold boot attacks. Since we assume that a CSP cannot tamper with the code and data in the enclave or the SGX hardware, attacks on SGX hardware [8][9][10] are beyond the scope of this study. Note that while we assume a malicious CSP is the most critical threat, any malicious software, such as malware in the database server, is also considered a threat in this study. Furthermore, we assume that FTSN users appropriately keep sensitive waypoints private using their defined EPZs. Therefore, attacks that attempt to identify sensitive waypoints using publicly disclosed waypoints discussed in [1][2] are beyond the scope of this study.

B. MosaicDB concept and archtecture

A concept of MosaicDB is shown in Figure 3. MosaicDB capitalizes on the fact that location data outside the EPZ is public in the application context, reducing the need to handle it as sensitive data in the database. When inserting location data, MosaicDB checks whether location data is included within the EPZ by calculating the geographical distance between the center of the EPZ stored in MosaicDB and location data. In all operations performed by MosaicDB, the location data within the EPZ are managed in the enclave, whereas other location data is managed in the untrusted memory. MosaicDB manages location data by utilizing both memory and storage, similar to typical RDBMSs. MosaicDB encrypts the location data loaded into the enclave by using the sealing before persistence.

The architecture of MosaicDB is shown in Figure 4. MosaicDB has general components in the typical RDBMSs, such as the parser, planner, executor, and storage engine. To realize location data management using both the enclave and untrusted memory, we duplicate the executor, responsible for query execution, and the storage engine, which accesses data on buffers and storage, respectively. The executor and storage engine within the enclave handle queries involving sensitive location data, whereas those in untrusted memory execute queries related to non-sensitive location data. The parser, which generates an abstract syntax tree from a query string; the planner, which generates an execution plan; and the preprocessor, which checks whether the location data is contained within the EPZ, are placed in the enclave because they handle unencrypted location data that may or may not be within the EPZ. The following subsections describe the query execution flow using these components.

C. Query execution flow

We will describe the query execution flow of MosaicDB by dividing it into three parts. Note that MosaicDB is currently designed to handle only simple CRUD (CREATE, INSERT, UPDATE, DELETE) queries, and supporting more complex queries is future work.

1) *Query analysis and optimization*

In the initial stage of query execution, MosaicDB analyzes queries and creates optimized execution plans. The network module (① in Figure 4) first receives the encrypted query string from the client, and the decryptor (② in Figure 4) decrypts the query string in the enclave. The keys used for encryption and decryption are exchanged via remote attestation when the client connects to the database. Then, the parser (③ in Figure 4) generates a query tree, which is an abstract syntax tree, from the query string and passes it to the planner. Finally, the planner (④ in Figure 4) generates a plan tree, which is an optimized execution plan, from the query tree and passes it to the preprocessor (⑤ in Figure 4).

2) *Checking location data within the EPZ*

If the query is `INSERT`, the preprocessor checks whether the location data in the plan tree is contained within the EPZ. If contained within the EPZ, it passes the plan tree to the trusted executor; otherwise, it passes it to the untrusted executor (⑥ in Figure 4). If the query is not `INSERT`, the checking process is ignored because the plan tree of a query like `SELECT * FROM locations;` does not contain location data, and the plan tree is passed to both the trusted and untrusted executors. EPZs are

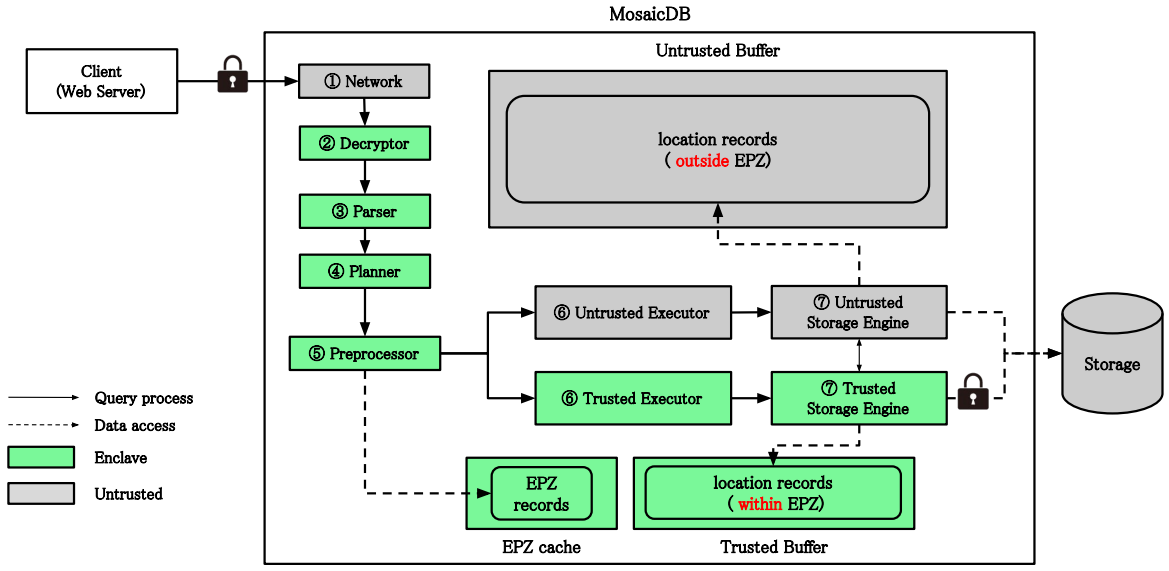

Figure 4. Overview of MosaicDB

cached in memory as an dictionary with user ID as a key (*EPZ cache* in Figure 4) because reading EPZ records on the storage has an extra cost because of sealing. Through an evaluation, we will confirm that the EPZ cache can effectively mitigate overhead, even with an increased number of EPZs.

3) ***Processing the plan tree in the enclave and untrusted memory***

The trusted and untrusted executors execute queries according to the plan tree. The execution process performed by the trusted and untrusted executors is largely similar to a typical RDBMS, but there is a difference in terms of the query execution results. In MosaicDB, if the execution of either the trusted or untrusted executor fails, the final query execution result is a failure. This allows data updates caused by failed transactions to be rolled back, thereby preventing data inconsistency.

The trusted and untrusted storage engines (⑦ in Figure 4) engines insert, scan, update, and delete location data in memory and storage according to requests from the executor. The location data is managed in buffers (*Trusted / Untrusted Buffer* in Figure 4), which are located in the enclave and untrusted memory respectively. They are organized into fixed-length blocks called pages. The location data within the EPZ are stored in a page in the enclave (*Secure page* in Figure 5), and other location data is stored in a page in the untrusted memory (*Normal page* in Figure 5). The secure page is encrypted using the sealing and is decrypted only in the enclave, so an attacker cannot steal the location data in the secure page. The metadata page in the normal buffer is a page that stores metadata such as the page IDs of secure / unsecure pages, and is referenced from both the enclave and untrusted memory. After query execution, the executors return the result to the client via the network module. In the case of queries that need to return records, the merged records obtained in the trusted and untrusted executors are encrypted and returned.

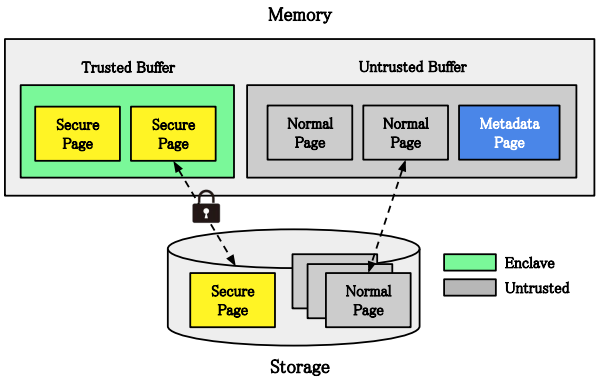

Figure 5. Page management in MosaicDB

V. Evaluation

To evaluate the overhead and memory utilization efficiency of MosaicDB, we compare the execution times and memory usage of INSERT and SELECT queries between MosaicDB and the baseline, where all the location data is managed in the enclave. In the current implementation, the execution flows of UPDATE and DELETE queries are almost the same as those of a typical RDBMS; therefore, we exclude these queries from the evaluation.

We used an Intel(R) NUC Kit NUC7PJYH as the experimental environment. The CPU is an Intel(R) Pentium(R) Silver J5005, the memory is 16 GB, and the storage is 256 GB. For the experiments, we used Geolife trajectory datasets [11] provided by Microsoft from which we extracted the amount of data required for each experiment. All EPZs in the experiment were set to be within 5 km of Peking University, where the location data in the dataset is concentrated.

 ISBN: 978-1-68558-138-1

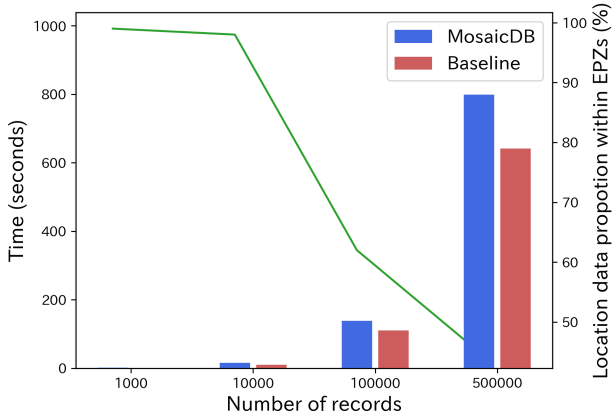

Figure 6. Execution time of INSERT

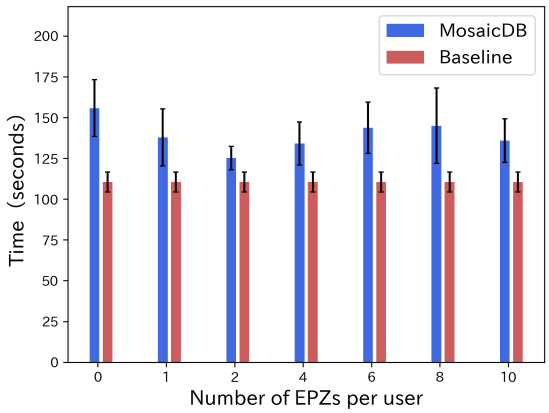

Figure 7. Execution time of INSERT while varying number of EPZs

A. Execution Time

We evaluated the execution times of INSERT and SELECT queries while varying the data size. We conducted two different experiments for INSERT queries. One experiment involved keeping the number of EPZs fixed while varying the data volume, and the other involved maintaining a fixed data volume while changing the number of EPZs. In the former experiment, we fixed the number of EPZs to one. In the latter experiment, the maximum number of EPZs was set to ten, which aligns with the typical number of EPZs expected in general FTSNs. We used 100,000 records in the latter experiment, and the execution times were averaged over ten runs in both experiments. Finally, the green line in Figure 6, 8 represents the proportion of location data within the EPZ.

Figure 6 shows that MosaicDB increases an overhead by 1.2 to 1.6 times compared to the baseline in INSERT query. On the other hand, an increase in the number of EPZs resulted in minimal additional overhead, as shown in Figure 7. This indicates that the increase in the number of EPZs can be reduced by EPZCache, while there is some overhead in MosaicDB. When there are no EPZs, MosaicDB's execution time is at its slowest because all location data is processed in the untrusted memory. Query processing in the untrusted memory involves additional overhead, such as plan tree serialization and deserialization, as well as additional transitions between the enclave and untrusted memory, making it more costly than executing queries in the enclave.

Figure 8 shows that MosaicDB can reduce SELECT query execution time by up to 25% compared to the baseline. MosaicDB achieves this performance improvement due to the reduced overhead of sealing, as it does not encrypt location data outside the EPZ, unlike the baseline.

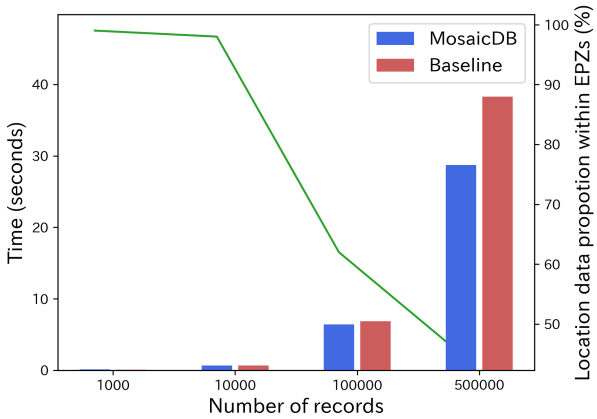

Figure 8. Execution time of SELECT

B. Memory Usage

We estimated the memory usage of MosaicDB while varying the amount of data stored in the enclave and untrusted memory for INSERT and SELECT queries. Since MosaicDB determines whether to store location data in the enclave or untrusted memory during insertion, the amount of location data stored in the enclave or untrusted memory remains constant for INSERT and SELECT queries. Consequently, we calculate the memory usage as the product of the number of records and the size of a record.

Figure 9 shows that both the enclave and untrusted memory are used when the number of records exceeds 100,000. This observation aligns with the trend in Figure 6 and Figure 8, where the proportion of location data within the EPZ decreases as the number of records increases. It suggests that if the proportion of location data within the EPZ is sufficiently small, MosaicDB can effectively utilize the untrusted memory while keeping the enclave memory usage in check. For instance, if the proportion of location data within the EPZ is around 10%, approximately 90% of location data can be accommodated in the untrusted memory. Thus, we conclude that MosaicDB allows for more efficient utilization of the server's memory compared to conventional methods that manage all data in the enclave.

 ISBN: 978-1-68558-138-1

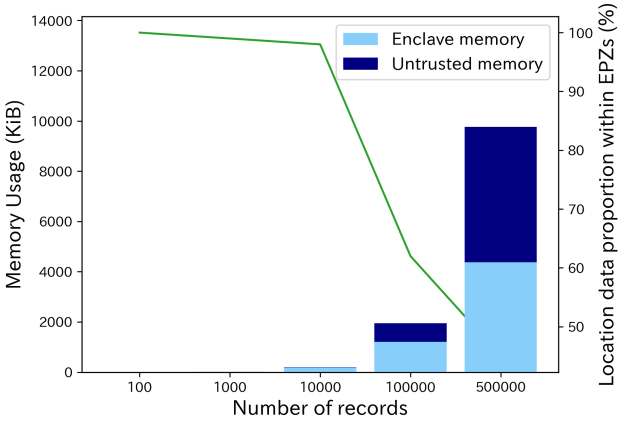

Figure 9. Enclave / untrusted memory usage in MosaicDB

VI. Discussion

In this section, we discuss the necessity of additional evaluation experiments to demonstrate the practicality of MosaicDB, as well as the limitations and future directions of MosaicDB.

A. Extended evaluation queries

In this paper, we evaluated the execution time and memory usage of the MosaicDB by using only simple INSERT and SELECT queries. However, the basic evaluation of these queries alone is insufficient to demonstrate the practicality of MosaicDB in FTSNs. For example, in real FTSNs, queries that simultaneously process location data and other data (e.g., queries that JOIN location data and other data) must be executed with high throughput; however, the performance of the MosaicDB in executing such queries has not been measured. In addition, such queries tend to consume a large amount of temporary buffer space in the RDBMS, which affects not only the execution time but also memory usage in the enclave. Therefore, it is necessary to confirm whether MosaicDB can improve the memory utilization efficiency for such queries in the future.

B. Evaluation of actual memory load

In our evaluation, we estimated the memory usage by quantifying the amount of location data stored in the enclave and untrusted memory. However, for a more accurate evaluation of the memory utilization efficiency of MosaicDB, it is necessary to measure the actual memory load on both the enclave and untrusted memory. In the first generation of SGX, the physical memory usage of the enclave is determined during enclave initialization, making the physical memory usage unsuitable for evaluation. Therefore, we plan to measure the enclave memory load by monitoring the SGX paging.

C. More flexible memory management

Considering memory efficiency, it is ideal to distribute the utilization of both the enclave and untrusted memory evenly. However, in MosaicDB, all location data outside the EPZ are stored in the untrusted memory. Consequently, there is a risk of overloading the untrusted memory when there is an extreme shortage of location data within the EPZ or an abundance of enclaves. A more flexible approach to data management tailored to the load conditions of the enclave and untrusted memory is necessary to address these variations in application conditions and database server memory setups.

VII. Conclusions

In this paper, we proposed MosaicDB, a memory-efficient and trusted database that manages location data using both the enclave and untrusted memory in SGX. MosaicDB utilizes the characteristics of FTSNs and protects only the location data within the EPZ in the enclave so that both the enclave and untrusted memory can be effectively utilized. Performance evaluation showed that MosaicDB can efficiently utilize the entire memory of the server. The extension of evaluation queries, evaluation of the actual memory load, and more flexible memory management are future works.

Acknowledgments

This work was partly supported by the ICS-CoE Program at the Information technology Promotion Agency, Japan.

References

[1] W. U. Hassan, S. Hussain, and A. Bates, "Analysis of privacy protections in fitness tracking social networks -or- you can run, but can you hide?," in *27th USENIX Security Symposium (USENIX Security 18)*, (Baltimore, MD), pp. 497–512, USENIX Association, Aug. 2018.

[2] K. Dhondt, V. Le Pochat, A. Voulimeneas, W. Joosen, and S. Volckaert, "A run a day won't keep the hacker away: Inference attacks on endpoint privacy zones in fitness tracking social networks," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pp. 801–814, Association for Computing Machinery, 2022.

[3] Y. Wang *et al.*, "Cryptsqlite: Sqlite with high data security," *IEEE Transactions on Computers*, vol. 69, no. 5, pp. 666–678, 2019.

[4] C. Priebe, K. Vaswani, and M. Costa, "Enclavedb: A secure database using sgx," in *2018 IEEE Symposium on Security and Privacy*, pp. 264–278, 2018.

[5] A. Gribov, D. Vinayagamurthy, and S. Gorbunov, "Stealthdb: a scalable encrypted database with full sql query support," in *Proceedings on Privacy Enhancing Technologies Symposium*, pp. 370–388, 2019.

[6] M. Yoshimura, T. Sasada, Y. Taenaka, and Y. Kadobayashi, "Memory efficient data-protection for database utilizing secure/unsecured area of intel sgx," in *DBKDA 2023, The Fifteenth International Conference on Advances in Databases, Knowledge, and Data Applications*, pp. 38–43, 2023.

[7] A. Ittai, G. Shay, J. Simon, and S. Vincent, "Innovative technology for cpu based attestation and sealing," in *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy (HASP '13)*, 2013.

[8] J. Götzfried, M. Eckert, S. Schinzel, and T. Müller, "Cache attacks on intel sgx," in *Proceedings of the 10th European Workshop on Systems Security*, pp. 1–6, Association for Computing Machinery, 2017.

[9] P. Borrello, A. Kogler, M. Schwarzl, M. Lipp, D. Gruss, and M. Schwarz, "ÆPIC leak: Architecturally leaking uninitialized data from the microarchitecture," in *31st USENIX Security Symposium (USENIX Security 22)*, pp. 3917–3934, USENIX Association, 2022.

[10] J. V. Bulck *et al.*, "Foreshadow: Extracting the keys to the intel SGX kingdom with transient Out-of-Order execution," in *27th USENIX Security Symposium (USENIX Security 18)*, pp. 991–1008, USENIX Association, 2018.

[11] Y. Zheng, H. Fu, X. Xie, W.-Y. Ma, and Q. Li, *Geolife GPS trajectory dataset - User Guide*, geolife gps trajectories 1.1 ed., 7 2011.

 ISBN: 978-1-68558-138-1

Implementing the draft Graph Query Language Standard

The Financial Benchmark

Malcolm Crowe
Emeritus Professor
University of the West of Scotland
United Kingdom
e-mail: malcolm.crowe@uws.ac.uk

Fritz Laux
Emeritus Professor
Reutlingen University
Germany
e-mail: fritz.laux@reutlingen-university.de

***Abstract*—The International Standards Organization (ISO) is developing a new standard for Graph Query Language, with a particular focus on graph patterns with repeating paths. The Linked Database Benchmark Council (LDBC) has developed benchmarks to test proposed implementations. Their Financial Benchmark includes a novel requirement for truncation of results. This paper presents an open-source implementation of the benchmark workloads and truncation.**

Keywords- typed graph language; property graph management; relational database; implementation; truncation.

I. INTRODUCTION

The growth in the use of graph models has led to the development of standards including the publication of ISO 9075-16: Property Graph Queries (PGQ) [1], and the imminent emergence of a draft international standard for Graph Query Language (GQL) [2]. These developments draw on experience with commercial graph database products and envisage a clear convergence at the conceptual level between graph-based and relational database management, while GQL remains a separate standard. The principal novelty of GQL is its support for repeating graph patterns, which are useful in many applications including detection of fraud, analysis of supply chains, and cybersecurity [3].

Our previous work [4] has recommended the implementation of graph databases by extending the capabilities of a suitable Relational Database Management System (RDBMS) using metadata and additional built-in data types and syntax, particularly for the graph-oriented CREATE and MATCH statements, and has presented a working open-source solution that conforms to the usual requirements for RDBMS including transactions and security. In this paper we present an open-source RDBMS implementation, PyrrhoDB [5] that is able to perform graph creation and pattern matching including repeating patterns and also aligns well with the draft international GQL standard.

In particular, we will focus on the Financial Benchmark from the Linked Data Benchmark Council (LDBC), which explores the important use case of fraud detection and contains sample databases and illustrative workloads. The benchmark allows the performance of different implementations to be compared and introduces the new concept of truncation for managing the extent of searching, especially for historical data.

The benchmark envisages a database built to collect data on transfers between (possibly blocked) accounts, multiple ownership of accounts and relationships with and between companies, loan applications, guarantees, and remote operation of accounts (possibly using blocked or stolen devices), with a view to discovering and documenting criminal behavior including theft, fraud, and money laundering. The UML diagram is shown in Figure 1.

When graph databases contain event data accumulated over years, simply searching for a particular suspicious graph pattern can take an unreasonably long time. In extreme cases, where early detection is important (tight latency requirements), but nodes of interest have millions of edges to be investigated (power-law distribution of data) despite all available restrictions, it can become desirable to have a tunable mechanism to *truncate* the number of edges searched at each stage. The proposal in the benchmark is to maintain deterministic behavior by specifying a specific ordering to be used when the number of edges to be traversed exceeds a threshold. This threshold should be tunable on a per-query basis.

Naturally, the benchmark does not specify a mechanism for truncation. In this paper we offer an efficient implementation of this concept suitable for the direct, incremental, search algorithm in our open-source RDBMS.

The plan of this paper is to review the new implementation details in Section II. Section III presents an illustrative example, and Section IV provides some conclusions.

II. IMPLEMENTATION DETAILS

We begin with a brief review of the graph pattern matching support in the standard, and the syntax definitions used in our relational database implementation. Further details are available in the references. Section B below discusses LDBC's truncation concept and the added syntax for this feature used in our implementation.

A. Node and Edge Types

Our implementation of GQL using relational technology is fully described in [4] and [5]. Its database server accepts and directly implements both SQL and GQL source from the client, and its storage consists of the transaction log.

 ISBN: 978-1-68558-138-1

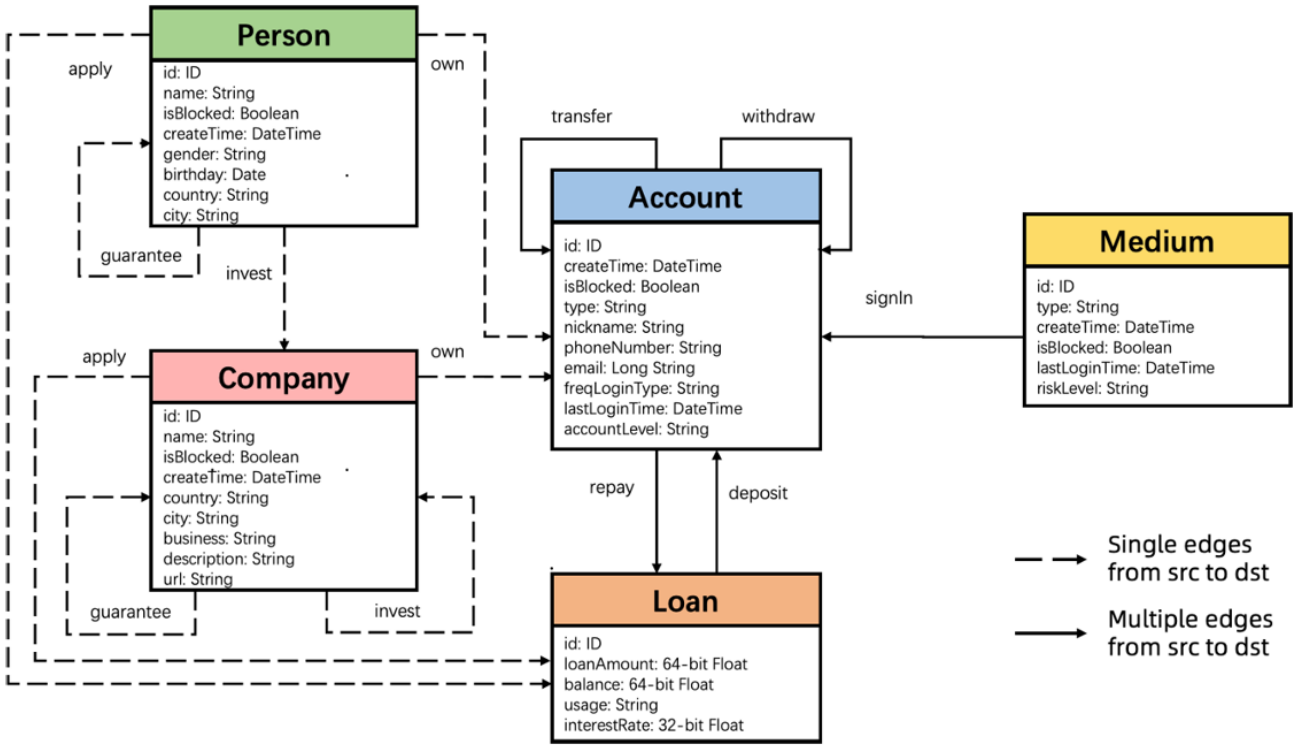

Figure 1. The LDBC FinBench data schema (from [6])

Specifically, GQL's CREATE statement is executed by a deep traversal of its graph patterns described while obeying the implied SQL DDL and DML actions, and the MATCH statement has its own query engine, which constructs a derived table resulting from a deep traversal of its graph patterns. Most MATCH statements have a dependent statement: for example, the GQL YIELD or RETURN statement, which provides expressions to construct results for the client. For example:

```
MATCH (P:Person{name:'Hatfield'})-[:own]
    ->(A:Account) return P.id,A.id
```

This example will give a table showing the id and account numbers of the persons called Hatfield. Unbound identifiers such as P and A above can be introduced at any point in the pattern, as shown in our next example, which also shows a MATCH statement without a dependent statement.

The MATCH statement allows the user to specify a graph fragment in queries instead of using joins. For example, with the scenario shown in Figure 1, the following query shows the details of all transfers in the database from any account owned by Hatfield:

```
MATCH (:Person{name:'Hatfield'})-[:own]->()
-[:transfer{amount:m,"timestamp":d}]->()
<-[:own]-(:person{name:r})
```

Figure 2 shows the result when this query is applied to the small LDBC financial benchmark database sf001.

The implementation begins by treating node and edge types are special kinds of SQL user defined types. Then each node or edge type corresponds to a base table in the relational database, whose rows are specific nodes and edges in the graph. Edges must identify two nodes: for directed

```
SQL> MATCH (:Person{name:'Hatfield'})-[:own]->()-[:transfer{amount:m,"timestamp":d}]->()
<-[:own]-(:person{name:r})
|----------|-------------------|----------------|
|M         |D                  |R               |
|----------|-------------------|----------------|
|2977613.82|07/10/2022 04:35:24|Skundric        |
|6888877.75|16/10/2022 03:43:21|Hamahang        |
|989617.6  |26/10/2022 18:14:50|Buchkin         |
|4024112.15|27/10/2022 10:04:02|Alfaro Siqueiros|
|----------|-------------------|----------------|
SQL>
```

Figure 2. A simple MATCH statement applied to the LDBC Financial Benchmark database sf001 (available from[6]).

edges these are called the source or leaving node and the destination or arriving node. The implementation constructs primary and foreign key indexes to support this structure.

The CREATE statement allows creation of nodes and associated edges in line using special tokens: nodes are enclosed in parentheses and edges join these using tokens `-[ , ]->` , `<-[ , ]-`. One or more such patterns can be provided in a single CREATE statement. Within the parentheses or square brackets there is provision for an alias, a type label, and properties. The alias can be used to refer to the node or edge later in the same statement. Execution of the CREATE statement constructs new nodes and edges in the database with the given properties.

The MATCH statement allows retrieval of graph data by providing one or more patterns of nodes, edges and properties, similarly written to the CREATE statement, which is tested against the contents of matching database tables. A pattern will generally yield a table of bindings of new identifiers encountered in the pattern, which can be used in a dependent statement (e.g., a CREATE or RETURN statement). The RETURN statement can also contain aggregations whose scope is the entire binding table.

In addition to the simple graph patterns such as those in CREATE statements, MATCH statements can contain quantified path patterns (an example is given below), which match a sequence of nodes and edges in the database that traverses the path pattern a number of times that conforms to requirements in the quantifier such as ? (0 or 1), + (1 or more), * (0 or more) or {a,b} (at least a and not more than b). The rules provide for management of duplicate edges, nodes, or bindings.

In the resulting binding table, aliases that occur within such repeating patterns will have values that are arrays: one element for each traversal of the path pattern.

B. The LDBC Truncation concept

In the financial benchmark specification [6], there is a concern that in selecting edges to follow from a given node (for example, traversing a set of transfers to or from an account) there may be hundreds or even millions of edges at each step, resulting in billions of cases to consider. It suggests a mechanism "to do truncation on the edges when traversing out from the current vertex", and to specify a sort order on such vertices to achieve consistency of results.

Since the traversal mechanism takes place inside the implementation of the MATCH statement, it makes sense to us to allow the truncation parameters to be specified as part of the creation of the MATCH statement, and we have constructed a syntax for this. The full syntax for Match in PyrrhoDB is shown in Figure 3. It includes:

```
MatchStatement = MATCH
    [Truncation] Match {',' Match}
    [WhereClause] [Statement] .

Truncation = TRUNCATING TruncationSpec
    {',' TruncationSpec} .

TruncationSpec = [EdgeType_id]
    ['(' OrderSpec {',' OrderSpec} ')'] '=' int .
```

The Truncation clause defines an upper bound for the number of edges to be traversed from a node in a step of the match process. The limit can be applied differently to specific edge types. Limits specified for supertypes of selected edges are also applied, as is the unnamed limit if present. It is explicit in the financial benchmark specification that the resulting truncation is performed within the execution of the database engine, and it is made deterministic by the specified ordering. There is an example in Figure 4 below.

The financial benchmark describes the truncation order as an enumeration and gives example values that are specific to the benchmark scenario, such as `TIMESTAMP_DESCENDING` and `AMOUNT_ASCENDING`. The syntax for OrderSpec is not shown here: in its simplest form it is a column name, but it can be a scalar expression optionally followed by ASC or DESC. Neither SQL nor GQL specifies a mechanism passing a parameter such as this to a stored procedure, but textual substitution is supported in prepared statements, which thus implement the notion of *general parameter* found in the GQL draft standard.

C. The Financial Benchmark Example

Figure 4 shows the first complex read-only query in the Financial Benchmark. The node types involved are Medium

```
MatchStatement = MATCH [Truncation] Match {',' Match} [WhereClause] [Statement] .
Truncation = TRUNCATING TruncationSpec{',' TruncationSpec} .
TruncationSpec = [EdgeType_id] ['(' OrderSpec {',' OrderSpec} ')'] '=' int .
Match = (MatchMode [id '='] MatchNode) {'|' Match} .
MatchNode = '(' MatchItem ')' {(MatchEdge|MatchPath) MatchNode}.
MatchEdge = '-[' MatchItem '->' | '<-' MatchItem ']-' .
MatchItem = [id | Node_Value] [GraphLabel] [ Document | Where ] .
MatchPath = '[' Match ']' MatchQuantifier .
MatchQuantifier = '?' | '*' | '+' | '{' int , [int] '}' .
MatchMode = [TRAIL|ACYCLIC| SIMPLE] [SHORTEST |ALL|ANY] .
```

Figure 3. PyrrhoDBMS's Match statement syntax [5].

and Account, and the only edge type is Transfer. The accompanying text in [6] reads: “Given an Account and a specified time window between startTime and endTime, find all the Account that is signed in by a blocked Medium and has fund transferred via edge1 by at most 3 steps. Note that all timestamps in the transfer trace must be in ascending order(only greater than). Return the id of the account, the distance from the account to given one, the id and type of the related medium.”

To be relevant for this example, each link in a transfer chain must occur later than its predecessor, and this is why the timestamps are constrained to be in ascending order. To implement this, we define the following stored function that compares a given timestamp with the timestamp property of the last element of the given array:

```
create function later (a Transfer array, t timestamp)
returns boolean
    begin
        declare c int=cardinality(a);
        if (c=0) then
                return true
        else
                return a[c-1]."timestamp"<t
        end if
    end
```

The specification uses the SQL reserved word **timestamp** as a property name, so double quotes are needed on each occurrence of the name of this property (the occurrence of timestamp in the function heading declares the parameter **t** as having type **timestamp**).

Our implementation of the complex query described above reads as follows (parameters are in red, outputs in blue, internal identifiers in green):

```
MATCH
 truncating Transfer
   ("timestamp" truncationOrder)=truncationLimit
 trail p=(m:Medium{isBlocked:true})
    -[:signIn where "timestamp">startTime and
        "timestamp"<endTime]->
    (:Account{id:otherId})
        [()-[x:transfer
 where "timestamp" >startTime and "timestamp" <endTime
        and later(p.x,"timestamp")]->()]{1,3}
    (:Account{id:id1})
 return
    otherId,
    (cardinality(p)-3)/2 as accountDistance,
    m.id as mediumId,
    m.type as mediumType
 order by (accountDistance,otherId,mediumId)
```

Cardinality is an SQL function, and the cardinality of the path **p** is the total number of nodes and edges traversed: the formula here computes the account distance as the number of Transfer edges traversed.

The path identifier gives SQL code such as the above access to the binding table during and after construction, so that **p.x** above refers to the current value of the **x** column of the binding table, that is, before the new **x** edge is added to it. On the other hand, **p** also gives access to the path of nodes and edges, so that **p[i]** is the **i**th member of the path (a node or an edge), and the cardinality of **p** is the length of the path.

Despite the multiple joins implied and the repeated execution of the stored procedure, execution of this statement is commendably fast: on the sf0 sample database

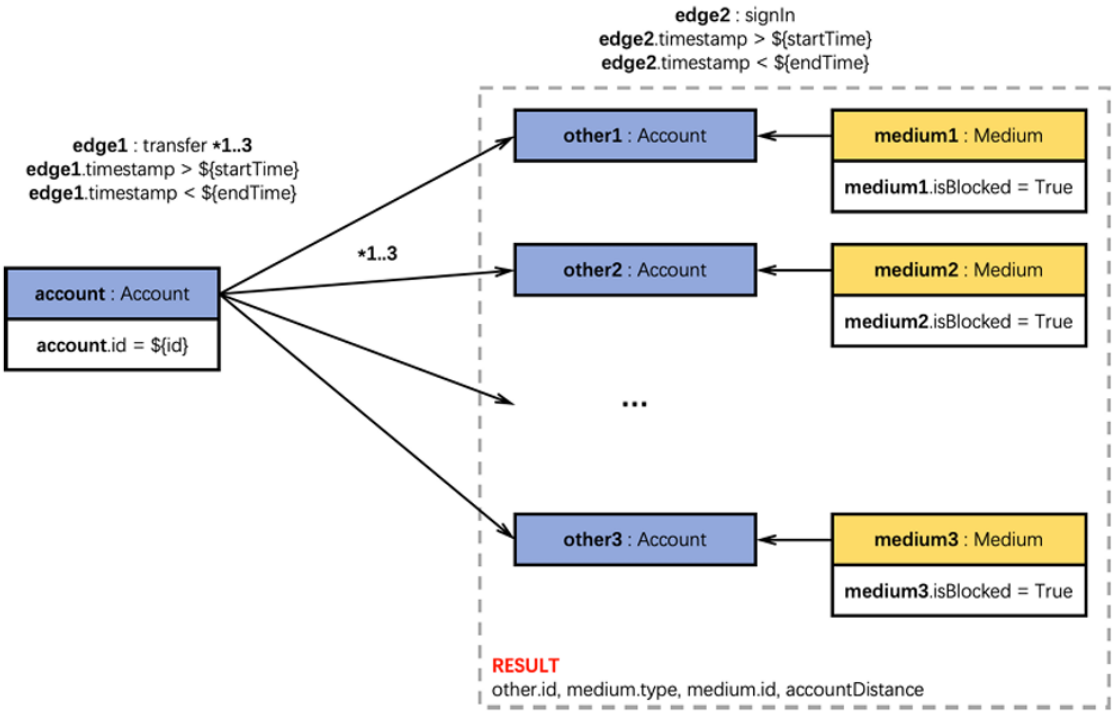

Figure 4. A complex read-only query (from [6]).

from LDBC, with the truncation defined as **transfer ("timestamp" desc)=10**, start time **timestamp'2022-01-01'**, end time **timestamp'2022-12-31'**, and id1 **4884435270860017215** it yields 3 rows in 7 seconds. Having identified the otherId accounts involved (here, 223491131508261941), an investigator can then investigate further.

III. Conclusions and Further Work

This step in our research into database technology was inspired by the LDBC Financial Benchmark [6], which suggested that truncation of graph pattern matching will often be a practical necessity for large graphs. We have proposed a general mechanism for search truncation, which in initial tests seems to be usable for searches in any property graph. With this in place, our prototype implementation is able to perform search efficiently even in large graphs.

As implementations of the draft international standard 39075 start to appear, there will be an opportunity to refine our proposal and compare it with other implementations of the benchmark.

References

[1] ISO 9075-16 Property Graph Queries (SQL/PGQ), International Standards Organisation, 2023.

[2] https://www.GQLStandards.org, October 4, 2023 – GQL status update [retrieved 18 October 2023].

[3] N. Francis et al.. A Researcher's Digest of GQL. 26th International Conference on Database Theory (ICDT 2023), Mar 2023, Ioannina, Greece. doi:10.4230/LIPIcs.ICDT.2023.1, pp. 1-22. https://hal.science/hal-04094449 [retrieved: 18 October 2023]

[4] M. Crowe and F. Laux, "Database Technology Evolution II: Graph Database Language", IARIA International Journal on Advances in Software, vol. 16 numbers 3 and 4, 2023, pp. 192-203, ISSN: 1942-2628.

[5] M. Crowe, PyrrhoV7alpha, https://github.com/MalcolmCrowe/ShareableDataStructures [retrieved: Dec 2023].

[6] Linked Data Benchmark Council: The LDBC Financial Benchmark (version 0.1.0), https://arxiv.org/pdf/2306.15975.pdf [retrieved Jan 2024].

 ISBN: 978-1-68558-138-1

Constructing and Analyzing Different Density Graphs for Path Extrapolation in Wikipedia

Martha Sotiroudi, Anastasia-Sotiria Toufa, Constantine Kotropoulos
Department of Informatics, Aristotle University of Thessaloniki
Thessaloniki, 54124, Greece
Email: {marthass, toufaanast, costas}@csd.auth.gr

***Abstract*—Graph-based models have become pivotal in understanding and predicting navigational patterns within complex networks. Building on graph-based models, the paper advances path extrapolation methods to efficiently predict Wikipedia navigation paths. The Wikipedia Central Macedonia (WCM) dataset is sourced from Wikipedia, with a spotlight on the Central Macedonia region, Greece, to initiate path generation. To build WCM, a crawling process is used that simulates human navigation through Wikipedia. Experimentation shows that an extension of the graph neural network GRETEL, which resorts to dual hypergraph transformation, performs better on a dense graph of WCM than on a sparse graph of WCM. Moreover, combining hypergraph features with features extracted from graph edges has proven to enhance the model's effectiveness. A superior model's performance is reported on the WCM dense graph than on the larger WIKISPEEDIA dataset, suggesting that size may not be as influential in predictive accuracy as the quality of connections and feature extraction. The paper fits the track Knowledge Discovery and Machine Learning of the 16th International Conference on Advances in Databases, Knowledge, and Data Applications.**

Keywords*—*Wikipedia Dataset; Path Extrapolation; GRETEL; Dual Hypergraph Transformation; Graph Neural Networks.

I. INTRODUCTION

Graph structures offer an intuitive and powerful means to capture relationships and interactions within various kinds of data, paving the way for advanced analysis through the prism of Graph Neural Networks (GNNs) [1]–[5]. From node classification [6]–[8] to link prediction [9] [10], GNNs have proven indispensable across a spectrum of applications. Among these, the task of link prediction focuses on path inference, namely to predict an agent's trajectory over a graph.

The efficacy of such models is inherently tied to the quality and structure of the underlying graph. In this context, our work pivots on the creation of the Wikipedia Central Macedonia (WCM) dataset, a new dataset comprising paths extracted from the huge graph of Wikipedia, with a specific emphasis on articles related to Central Macedonia, Greece. The dataset tries to simulate human navigation paths as in WIKISPEEDIA [11] game, where users are asked to navigate from a given source to a given target article by only clicking Wikipedia links. Our objective is to leverage this dataset to address the problem of path inference.

WCM dataset is specifically designed to navigate through the complexities of Wikipedia's topology. It takes "Central Macedonia" as the starting article, from which it explores the external links through a series of random walks. Each step is contingent on a set of well-defined validity criteria. This ensures that each selected link is pertinent and non-redundant, providing a true reflection of the path an agent might traverse within the bounds of this thematic cluster. The dataset constructed for this study is made publicly available [12]. It comprises two separate files within the `Wikipedia_Dataset` directory, representing the `Dense Graph` and the `Sparse Graph` structures, each containing details of the paths, unique articles, path identifiers, categories, edges, hyperedges, observations, and path lengths. The code to create the WCM dataset can be found at [13].

The interest in the path inference problem has led to the development of advanced models like GRETEL [14], which has demonstrated promise in leveraging path extrapolation on graphs. GRETEL works as a generative model trying to capture the directionality of the path. It has been applied to both navigation data and paths constructed on the Wikipedia graph. This paper applies a graph transformation method based on the Dual Hypergraph Transformation (DHT) [15]. This method, as demonstrated in [16] [17], extends the traditional graph framework enabling connections among multiple nodes (i.e., vertices) within a hypergraph. Hypergraphs are suitable for this purpose because their edges can connect any number of nodes, not just two, as in a conventional graph. The new representation is able to capture more complex interactions between the data, and new more representative features can be extracted [18].

Here, in pursuit of advancing path extrapolation methods, WCM dense and sparse graphs are employed to assess both the original GRETEL and the Dual GRETEL variant in environments of varying complexity, providing a thorough insight into its adaptability and accuracy in different graph densities. To capture a comprehensive range of interactions within the data, a feature extraction process is implemented as proposed in [14] [16] [17]. [16] introduces an enhanced model, DualGRETEL+, that applies dual hypergraph transformation and a second-order optimizer to GPS navigation data, showing improved path inference capabilities. [17] assesses path extrapolation using GRETEL on Wikipedia data, with a focus on extracting informative features through the DHT.

The paper is structured as follows: Section II provides a detailed description of the dataset creation and its characteristics, along with an overview of the features employed and the

 ISBN: 978-1-68558-138-1

GRETEL model. A detailed exposition of the experiments and results is found in Section III. The paper concludes in Section IV, underscoring the profound impact of graph density on the path extrapolation with graph neural networks.

II. METHODOLOGY

This section focuses on the methodical approach to creating and analyzing the WCM, outlining the comprehensive process of collecting, categorizing, and extracting features from Wikipedia data to construct various graph types for path extrapolation.

A. Dataset Creation

The dataset is created through a crawling process designed to traverse the vast interconnected landscape of Wikipedia, with Wikipedia Central Macedonia article [19] serving as the focal starting point. During data collection, we remained cognizant of the load implications on Wikipedia's servers. We inserted a pause of one second between two requests, safeguarding against potential server overload while accessing Wikipedia's data. This was a measure of digital courtesy and sustainability.

The path generation process begins with the Central Macedonia Wikipedia article. From this starting point, the crawler extracts all the external links associated with the current article. A subsequent article is then randomly selected from the set of external links, adhering to certain validity checks, ensuring the relevance of the link and its absence from the current path. To maintain the integrity of the dataset and concentrate solely on core articles, stringent validation criteria are instated. The process of path creation continues until the generated path either attains a predetermined length ranging from 4 to 7 articles or encounters an article devoid of valid external links. The algorithm employs a well-defined criterion to ensure the relevance and validity of each article within the path. The function `is_valid_title` is utilized to exclude titles containing terms like `Talk`, `User`, `File`, 'ISO', percentages, hashes, or colons, and those consisting solely of digits. This careful filtering is instrumental in maintaining a dataset focused on content-rich articles, avoiding disambiguation pages, meta-articles, or other forms of non-standard content that could detract from the dataset's integrity.

To ensure the intelligibility of the dataset, each Wikipedia article is associated with a distinct identifier. Leveraging tensor manipulation, the identifiers for the linked articles are distilled and organized within distinct tensor frameworks. These tensors serve as the foundation for the node indices within the constructed graph. To further aid our analysis, each trajectory's length is documented, and each article in the trajectory is associated with its unique identifier. This process is reiterated until a grand total of 3000 paths emerges. The graph G is created comprising m nodes and n edges, where nodes represent articles and edges denote links between articles. The extracted paths are referred to as *trajectories*. We have documented these trajectories, noting their lengths and the articles they connect. The graph is represented using the Graph Markup Language.

Two distinct graph types have been created, each following a unique path selection process:

Dense Graph: This is formed by a modified path selection protocol within the crawler. Here, the crawler opts for a random choice from the first five external links of an article. We choose the first five external links for path selection to intentionally narrow down the possible trajectories, aiming for a denser graph structure that facilitates a more focused analysis of interconnected topics. This results in a connected network among a smaller subset of 912 nodes, 1311 edges, and 3000 paths. The same process of path generation, involving the extraction of links, applying validity checks, and documenting each trajectory with unique identifiers, is followed as in the general dataset creation.

Sparse Graph: This graph follows the initial broader selection process, incorporating a more extensive set of 7307 nodes, 10612 edges, and 3000 paths. The selection is made from all the external links.

B. Article Categorization

Categorization provides a structured framework to analyze the dataset. Organizing articles into distinct categories enables researchers to identify content trends and patterns within the generated paths. This categorization not only enriches the dataset but also amplifies its potential utility for diverse research, analytical, and educational purposes.

Our categorization strategy focuses on dynamic online querying using DBpedia [20]. In order to determine the category of a given Wikipedia article, we rely on the SPARQL endpoint of DBpedia. Each article is queried to retrieve its semantic type from DBpedia's ontology. Whenever an explicit type is not obtained or if there are errors during the querying process, the articles are classified under `subject.General`.

C. Feature Extraction

In addition to graph generation, a feature extraction process is conducted to leverage semantic information from the content of the articles and to capture complex interactions in the graph structure. According to [14], the feature vector for the nodes corresponds to its `in/out degree`, and its length is 2. For edges, the feature vector contains the Text Frequency - Inverse Document Frequency (`TF-IDF score`), capturing the semantic similarity between source and destination articles of a hyperlink [21], and the number of times the link was clicked in the training dataset of paths (`nof`).

1) Dual Hypergraph Transformation

The framework commences with the configuration of a conventional graph, designated as G having n nodes and m edges. Node features are represented by a feature matrix $\boldsymbol{F} \in \mathbb{R}^{n \times d}$, and edge features by a feature matrix $\boldsymbol{E} \in \mathbb{R}^{m \times d'}$. Here, d and d' are the size of node and edge feature vectors, respectively. Considering an undirected graph, the incidence matrix is defined as $\boldsymbol{M} \in \{0, 1\}^{n \times m}$. In the case of a directed graph, the incidence matrix is defined as $\boldsymbol{M} \in \{-1, 0, 1\}^{n \times m}$.

 ISBN: 978-1-68558-138-1

In any case, the incidence matrix represents the relationships between nodes and edges in a graph, indicating which nodes are connected by specific edges.

The conventional graph and the corresponding dual hypergraph are represented as $G = (\boldsymbol{F}, \boldsymbol{M}, \boldsymbol{E})$ and $G^* = (\boldsymbol{F}^*, \boldsymbol{M}^*, \boldsymbol{E}^*)$ respectively. $\boldsymbol{F}^*$ represents the node features of hypergraph while $\boldsymbol{E}^*$ represents the hyperedge features. The DHT algorithm interchanges the roles of nodes and edges of the original graph [15]. That is, the edges of the original graph are reinterpreted as nodes in the dual hypergraph, while the original nodes become hyperedges in the dual hypergraph. Accordingly, $\boldsymbol{F}^* = \boldsymbol{E} \in \mathbb{R}^{m \times d'}$ and $\boldsymbol{E}^* = \boldsymbol{F} \in \mathbb{R}^{n \times d}$. The incidence matrix of the dual hypergraph is the transpose of the incidence matrix of the original graph, i.e., $\boldsymbol{M}^* = \boldsymbol{M}^\top$. The transformation is mathematically defined as:

$$G = (\boldsymbol{F}, \boldsymbol{M}, \boldsymbol{E}) \rightarrow G^* = (\boldsymbol{E}, \boldsymbol{M}^\top, \boldsymbol{F}) \tag{1}$$

Notably, the DHT is a reversible transformation, ensuring that applying it to G^* recaptures the initial graph G, thereby preserving the structural and feature integrity of the transformation.

2) Features extracted from the dual hypergraph

Following the methodology proposed in [16], the original graph is transformed into its corresponding dual graph by applying the DHT algorithm in order to capture more complex interactions among edges. Two new features are extracted, namely the `similarity-hyperedge` and the `DHnode-in-out-degree`. The first feature assumes an undirected graph, while the second one assumes a directed graph. The implementation of dual hypergraph feature extraction, which significantly enhances the predictive accuracy of our models, can be found in [22].

For the `similarity-hyperedge` feature, the first step is to construct the incidence matrix $\boldsymbol{M} \in \{0,1\}^{n \times m}$. Row vector $\boldsymbol{q}_l \in \{0,1\}^m$ of $\boldsymbol{M}$, corresponds to node l. The cosine similarity between the incidence row vectors $\boldsymbol{q}_v$ and $\boldsymbol{q}_u$ is computed, where v is the source node and u is the target node of an arbitrary edge e. The corresponding vector in the $\boldsymbol{M}^*$ matrix is a column vector $\boldsymbol{q}_l^* \in \{0,1\}^m \equiv \boldsymbol{q}_l^\top$. The position of each 1 in this column vector indicates which nodes of the dual hypergraph are connected with the hyperedge l^*.

For the `DHnode-in-out-degree`, a directed graph G is assumed. The corresponding incidence matrix is defined as $\boldsymbol{M} \in \{-1,0,1\}^{n \times m}$. To extract features associated with the input and output degrees of the dual hypergraph nodes, determining the direction of hypergraph edges becomes essential. This involves an examination of the column vector of $\boldsymbol{M}^*$ $\boldsymbol{q}_l^* \equiv \boldsymbol{q}_l^\top$. The position of each 1 in this column vector indicates which nodes of the dual hypergraph are connected with the hyperedge l^*. For every combination (v_i^*, v_j^*), we verify the existence of a path $e_i \rightarrow e_j$ in the original graph that passes through the scrutinized node l. The new feature is the in-degree and out-degree of dual hypergraph nodes which are normalized by the maximum observed degree $D_{\max}$ in the hypergraph to facilitate comparison across different nodes:

$$\text{Normalized In/Out-Degree } \left(v_i^*\right) = \frac{\text{In/Out-Degree } (v_i^*)}{D_{\max}} \tag{2}$$

The aggregation of `similarity-hyperedge` and `DHnode-in-out-degree` results in `Similarity-Hyperedge-DHnode-In-Out-Degree` feature. These enhanced features are particularly critical in the sparse graph context, where the reduced number of connections demands a more nuanced approach to capturing node relationships. In the dense graph, with its inherently richer connectivity, these features play a pivotal role in distilling the essence of the network's complexity into a format conducive to advanced path prediction algorithms.

The feature extraction procedure is performed on the sparse graph with 7,307 nodes and 10,611 edges and the dense graph with 912 nodes and 1,311 edges.

D. Path Extrapolation Employing GRETEL

The paper addresses path extrapolation focusing on predictive path analysis via the GRETEL model [14]. The graph G consists of nodes and edges, represented as $G = (\mathcal{V}, \mathcal{E})$, with $n = |\mathcal{V}|$ denoting the node count and $m = |\mathcal{E}|$ the edge count, respectively. An agent progresses through the graph, stepping from node v_i to v_j contingent on the presence of a directed edge $e_{i \rightarrow j} \in \mathcal{E}$.

The agent's position at time t is a sequential set of traversed nodes, symbolized as a given prefix $p = (v_1, v_2, \ldots, v_t)$. Let the path suffix $s = (v_{t+1}, \ldots, v_{t+h})$ be a collection of potential future for prediction horizon h. Within this setting, GRETEL is leveraged to estimate the conditional likelihood $\Pr(s \mid h, p, G)$ of path suffix s given the prefix p, the horizon h, and the graph G. The agent's position at each step t is encoded by a sparse vector $\mathbf{x}_t \in \mathbb{R}^n_{\geq 0}$ normalized to a unit sum, with its i-th element reflecting the likelihood of the agent being at node v_i.

GRETEL constructs a generative model that considers the directionality of edges via a latent graph with edge weights informed by a Multi-Layer Perception (MLP) that respects the graph's inherent directionality. The model's essence lies in its ability to forecast paths by learning from the traversed sequences, leveraging node features and the collective path history. More specifically, the non-normalized weights of each edge are computed by

$$z_{i \rightarrow j} = \mathrm{MLP}(\boldsymbol{c}_i, \boldsymbol{c}_j, \boldsymbol{f}_i, \boldsymbol{f}_j, \boldsymbol{f}_{i \rightarrow j}), \tag{3}$$

where $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ are the pseudo-coordinates of the sender and the receiver node, respectively, while $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$ denote the features of the sender and the receiver node, respectively. In (3), $\boldsymbol{f}_{i \rightarrow j}$ is the feature vector of the edge that connects the sender and the receiver node. The computed MLP outputs are normalized with the softmax function. The pseudo-coordinates $\boldsymbol{c}_i$ are computed using a GNN of K layers. They are the agent representations $\boldsymbol{x}_\tau$ for $\tau \in \mathcal{I}$, where $\mathcal{I}$ denotes a trajectory. The non-zero elements of $\boldsymbol{x}_\tau$ refer to the distance between the agent and the K closest graph nodes normalized to measure

 ISBN: 978-1-68558-138-1

one. Let $\vec{e}_t$ and $\vec{e}_t$ define the edges that go from $v_t \rightarrow v_{t+1}$ and $v_t \rightarrow v_{t-1}$, respectively. Let also $\boldsymbol{x}_t$ be the last position of the agent. GRETEL [14] can be trained through the *target likelihood*. That is, given a target distribution $\boldsymbol{x}_{t+h}$, the model tries to estimate the destination distribution $\hat{\boldsymbol{x}}_{t+h} \in \mathbb{R}^{n\times 1}$ over a horizon h by the non-backtracking walk [23]

$$\hat{\boldsymbol{x}}_{t+h} = \boldsymbol{B}_\phi^+ \boldsymbol{P}_\phi^h \boldsymbol{B}_\phi \boldsymbol{x}_t. \tag{4}$$

Let $w_\phi(e_{k\rightarrow j})$ stand for the normalized MLP weights. In (4), $\boldsymbol{P}_\phi \in \mathbb{R}^{m\times m}$ has elements

$$[\boldsymbol{P}_\phi]_{e_{i\rightarrow j}, e_{k\rightarrow l}} = \begin{cases} 0 & \text{if } j \neq k \text{ or } i = l \\ \frac{w_\phi(e_{k\rightarrow l})}{1-w_\phi(e_{k\rightarrow i})} & \text{otherwise,} \end{cases} \tag{5}$$

$\boldsymbol{B}_\phi$ is a $m \times n$ matrix with $[\boldsymbol{B}_\phi]_{e_{i\rightarrow j},k} = 0$ if $k \neq i$ and $w_\phi(e_{k\rightarrow j})$, otherwise, and $\mathbf{B}_\phi^+$ stands for the pseudoinverse of $\mathbf{B}_\phi$. Such an approach integrates node and edge feature vectors, the former delineating the in/out-degree and the latter embedding the textual and usage-based similarity metrics. These primal features are pivotal in the model's capacity to estimate the suffix likelihood, aiding in approximating the path probability $\Pr(s \mid h, \varphi, G)$. In the paper, we will aggregate the original edge features $\boldsymbol{f}_{i\rightarrow j}$ with the features extracted from the dual hypergraph.

III. Experiments and Results

To quantify the structure of each graph, we calculate the density, which provides a measure of how complete the graph is. The density is defined as the ratio of the number of edges m to the number of possible edges, with the formula given for a directed graph without loops as

$$D = \frac{m}{n(n-1)}, \tag{6}$$

where n is the number of nodes.

TABLE I. Dataset Characteristics

Datasets	Nodes	Edges	Density
Sparse Graph	7307	10612	2×10^{-4}
Dense Graph	912	1311	1.58×10^{-3}
Wikispeedia	4604	119882	5.66×10^{-3}

Table I summarizes the characteristics of the graphs used in the experiments, providing a clear comparison of the number of nodes, edges, and density across the Sparse Graph, the Dense Graph, and WIKISPEEDIA. Based on the characteristics outlined in Table I, the sparse graph demonstrates a lower density ratio due to its larger node count. In contrast, the dense graph, with fewer nodes, exhibits a higher density ratio. Notably, the WIKISPEEDIA dataset possesses the greatest density ratio of the three.

The following metrics are used to assess the feature vectors. *Target probability* measures the average chance that the model will choose a node with non-zero likelihood. *Choice accuracy* measures how accurate the decisions of an algorithm are at each crossroad of the ground-truth path, connecting nodes v_t and v_{t+h}. It is computed on nodes whose degree is at least 3. *precision top1* measures how often the correct next step appears in the model's first prediction only, while *precision top5* evaluates how often the correct next step appears within the model's first five predictions.

In all experiments, the node feature vector includes the in/out degree for the nodes, retaining a constant size of two, underscoring consistent complexity in nodal characteristics despite the variation in graph densities.

An empirical assessment of model performance using the features derived from the original graph and those of the corresponding dual hypergraph is conducted. In the case of `original edges`, the `TF-IDF score` and `nof` features are used, yielding a feature vector of size 2. By aggregating the features `similarity-hyperedge` of length 1, `DHnode-in-out-degree` of length 2, and their combination `similarity-hyperedge-DHnode-in-out-degree` of length 3, the associated edge feature vector has length 3, 4, and 5, respectively.

Table II summarizes the performance of the GRETEL model with original edge features and the features extracted from the dual hypergraph (Dual GRETEL) added on top of the original edge features on the Sparse Graph.

Table III repeats the model's performance assessment on the Dense Graph. Table IV details the model's performance on the WIKISPEEDIA dataset. This dataset encapsulates the essence of human navigational strategies within Wikipedia, compiling 51318 completed paths from the WIKI GAME where participants navigate through article links towards a target article, with an aim for efficiency in both clicks and time.

The modularity class algorithm in `Gephi` [24] is used to identify the clusters within the network. These clusters contain nodes that are more densely connected to each other than to nodes in different clusters. The resulting clusters are indicated by the color coding of the nodes. The size of each node is proportional to its degree, reflecting the number of connections it has within the network. This allows for the immediate visual identification of highly connected nodes. The visible labels on the nodes in the figures were chosen because they have higher degree values, which show their importance in the graph, and they represent the main topic of each cluster within the expansive Wikipedia network.

Figure 1 represents the dense graph of Wikipedia. The selective navigation results in a dense network with several clusters, one of which is built around the Central Macedonia article, connecting closely related topics. Adjacent nodes like 'History of Greece' and 'Politics of Greece' form clusters that delve into the nation's past and governance, and 'Geographic Coordinate System' and 'France' appear as nodes indicative of broader geographical discourse.

The visualization of the sparse graph in Figure 2 reveals a network that unfolds from the Central Macedonia article, forming a large, primary cluster due to the random link selection strategy, and extending outward into a sparse array of smaller clusters. These smaller clusters are thematic, with subjects such as European countries, Greek cities, and historical events.

 ISBN: 978-1-68558-138-1

TABLE II. PERFORMANCE METRICS (%) ON THE SPARSE GRAPH

	GRETEL	Dual GRETEL		
Metrics	**Original Edges**	**Similarity-Hyperedge**	**DHnode-In-Out-Degree**	**Similarity-Hyperedge-DHnode-In-Out-Degree**
target probability	68.76 ± 0.0044	68.76 ± 0.0019	68.99 ± 0.0064	$\mathbf{69.71 \pm 0.0038}$
choice accuracy	$\mathbf{51.18 \pm 0.0011}$	38.69 ± 0.0042	39.60 ± 0.0082	39.24 ± 0.0090
precision top1	66.65 ± 0.0050	66.71 ± 0.0012	66.65 ± 0.0025	$\mathbf{67.14 \pm 0.0045}$
precision top5	80.62 ± 0.0019	80.62 ± 0.0019	80.68 ± 0.0023	$\mathbf{80.98 \pm 0.0036}$

TABLE III. PERFORMANCE METRICS (%) ON THE DENSE GRAPH

	GRETEL	Dual GRETEL		
Metrics	**Original Edges**	**Similarity-Hyperedge**	**DHnode-In-Out-Degree**	**Similarity-Hyperedge-DHnode-In-Out-Degree**
target probability	0.0030 ± 0.0021	$\mathbf{19.1007 \pm 0.0004}$	18.8741 ± 0.0033	19.0980 ± 0.0026
choice accuracy	$\mathbf{48.0602 \pm 0.0135}$	27.8261 ± 0.0084	29.8662 ± 0.0096	29.5318 ± 0.0086
precision top1	0.001 ± 0.0023	19.8995 ± 0.0074	18.0904 ± 0.0075	$\mathbf{20.5025 \pm 0.0067}$
precision top5	0.2513 ± 0.0012	83.2161 ± 0.0088	82.8141 ± 0.0258	$\mathbf{83.8694 \pm 0.0112}$

TABLE IV. PERFORMANCE METRICS (%) ON THE WIKISPEEDIA DATASET

	GRETEL	Dual GRETEL		
Metrics	**Original Edges**	**Similarity-Hyperedge**	**DHnode-In-Out-Degree**	**Similarity-Hyperedge-DHnode-In-Out-Degree**
target probability	6.42 ± 0.1	$\mathbf{6.74 \pm 0.1}$	6.44 ± 0.2	6.2 ± 0.1
choice accuracy	22.16 ± 0.4	$\mathbf{23.2 \pm 0.1}$	22.88 ± 0.1	21.86 ± 0.4
precision top1	11.6 ± 0.2	$\mathbf{12.7 \pm 0.1}$	12.14 ± 0.1	11.66 ± 0.3
precision top5	30.1 ± 0.1	$\mathbf{30.14 \pm 0.1}$	30.02 ± 0.05	30 ± 0.09

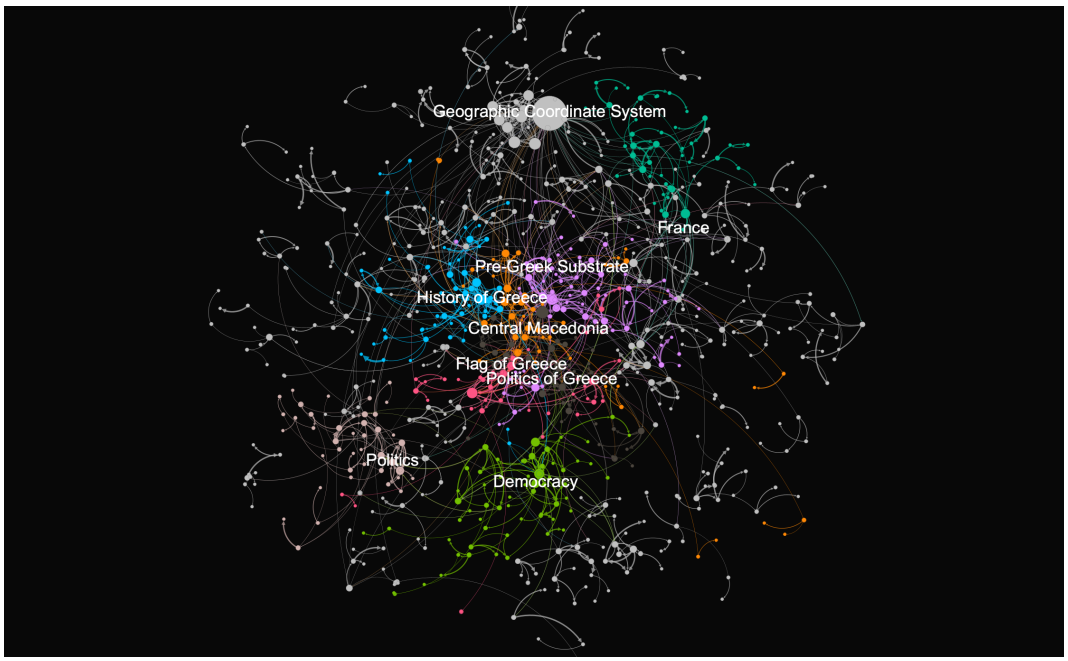

Figure 1. Dense Wikipedia Graph.

The construction methods of the two graphs distinctly shape their representations. The dense graph demonstrates that the Central Macedonia article forms a cluster, with surrounding clusters closely related in theme, predominantly focusing on Greece. This clustering suggests that the used method tends to group related topics tightly together. On the other hand, the sparse graph shows a different pattern where the Central Macedonia article and closely linked articles stand out in num-

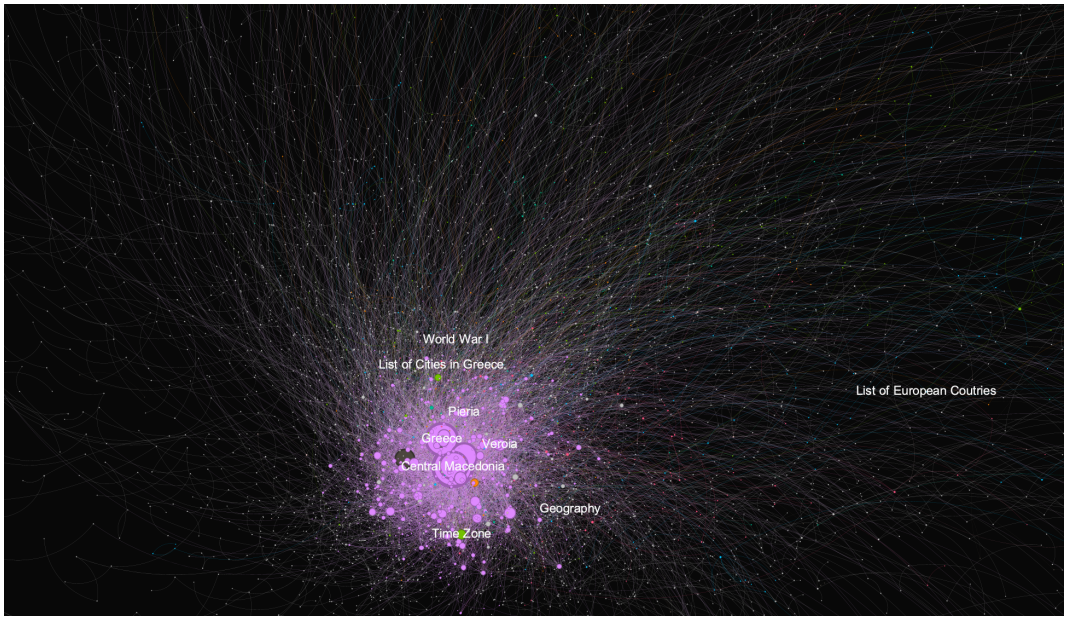

Figure 2. Sparse Wikipedia Graph.

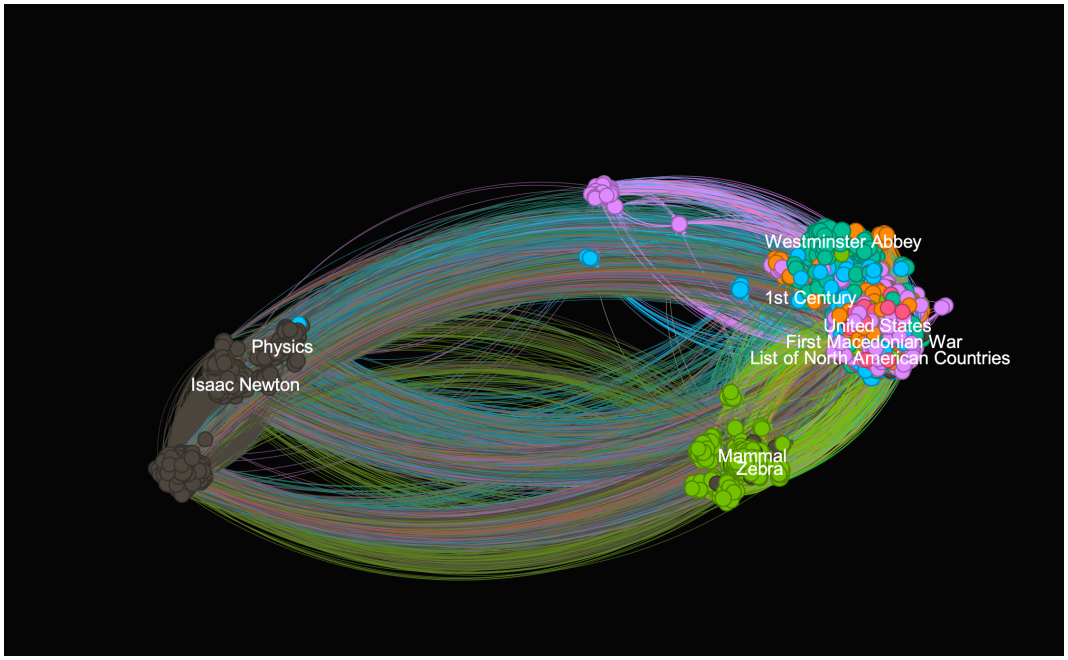

Figure 3. Wikispeedia Graph.

ber, while other articles appear less connected. This difference highlights how the choice of links in the construction process can significantly affect the network's structure.

Figure 3 represents the Wikispeedia graph, characterized by uniformly sized nodes, indicative of a network without a predominant starting article. Clusters within the graph are thematically organized, with 'Isaac Newton' and 'Physics' forming a cluster around scientific inquiry, while 'Westminster Abbey' serves as a node for the cluster concerning England. 'Mammal' and 'Zebra' are central to a cluster on zoology. These labels serve as the focal points for their respective clusters, marking the diverse subjects navigated by users.

Table V showcases examples of how the extracted features are employed to predict specific paths, highlighting the model's ability to deduce the most probable outcomes. Table V includes the conditional probabilities that reflect the model's ability to correctly anticipate the actual path taken. These examples are instrumental in illustrating the practical application of the model and the effectiveness of the features in guiding the model toward the most probable navigational

 ISBN: 978-1-68558-138-1

route. The utilization of hypergraph features results in higher conditional probability compared to the use of `original edge` features. The examples clearly show that when hypergraph features are considered, the model tends to assign a greater likelihood to the true path, suggesting that these features capture more of the complexities inherent in human navigational behaviors on Wikipedia. The examples are drawn from the sparse graph of the WCM dataset.

A. Performance Analysis of Model Across Dense and Sparse Graphs

In the evaluation of the Dual GRETEL model, distinct performances are observed between the sparse and dense graphs. A higher predictive accuracy with respect to *precision top5* metric is measured for the dense graph than the sparse one. This improved performance can be attributed to the vital role of the hyperedges, which enrich the model's contextual framework for more accurate extrapolation.

The sparse graph, despite its lower connectivity, shows commendable results, outperforming the dense graph in terms of target probability and choice accuracy. Dual GRETEL predicts the correct target with a probability of 69.71 ± 0.0038 %. GRETEL accurately chooses the next step with a rate of 51.18 ± 0.0011 %. It's noteworthy that except for the precision top 5, Dual GRETEL maintains a better performance on the sparse graph than the dense one.

This comparison reveals that while the Dual GRETEL model benefits from the rich link structures in dense graphs for precision tasks, it retains substantial predictive strength in sparse settings. This insight may guide further optimization for the model, enhancing its adaptability across varying network densities.

Also, this performance indicates that the model may benefit from the reduced complexity in sparse networks, potentially due to less noise and fewer connections, which can simplify the path prediction process. The comparison suggests that the model might generalize better in sparse environments, avoiding potential overfitting that can occur in dense networks with more intricate connections. Conversely, the specificity that dense networks provide can enhance the model's precision in certain contexts.

B. Model Benchmarking on WCM Dense Graph Versus Wikispeedia Graph

For the WCM dense graph, Dual GRETEL demonstrated a significant improvement, achieving an impressive precision top5 score of $83.8694 \pm 0.0112\%$. On the WIKISPEEDIA graph, Dual GRETEL also showed enhanced performance with a precision top5 score of $30.14 \pm 0.1\%$. This indicates that hypergraph features greatly enhance the model's ability to accurately identify the most likely paths in a dense environment.

Furthermore, GRETEL demonstrated a high choice accuracy of $48.0602 \pm 0.0135\%$ on the dense graph compared to $23.2 \pm 0.1\%$ of Dual GRETEL on the WIKISPEEDIA dataset. Our findings show that model performance on the dense graph improves across all metrics except *choice accuracy* when we use hypergraph features. That is, hypergraph features are particularly effective in densely connected graphs, enhancing the model's predictive accuracy across all metrics we tested. The results indicate the potential of hypergraph features to improve the performance of path prediction models like GRETEL, especially in complex network structures.

The completion rate of paths in the WIKISPEEDIA dataset may introduce additional complexity, given that there is a mixture of successful and abandoned paths. In contrast, the smaller dataset might offer more uniformly successful paths, influencing the ease with which the model can learn and predict.

The analysis of the model's performance, as shown in Tables II-IV, reveals a trend where effectiveness inversely correlates with graph density. This suggests that as graphs become more interconnected, the model encounters greater challenges in path prediction accuracy. These observations emphasize the critical role that graph density plays in the deployment and refinement of path prediction algorithms. A possible explanation for the deterioration of accuracy as density increases could be the rise in potential paths that the model must discern. In denser graphs, the increased interconnectivity results in a greater number of plausible trajectories between nodes, potentially complicating the model's task of pinpointing the most likely path. Furthermore, a dense network may introduce more noise in the form of less relevant or weaker connections, which could mislead the prediction algorithm. These findings indicate that models like GRETEL or Dual GRETEL may require adjustments or enhancements, such as more sophisticated feature extraction or the incorporation of context-aware learning mechanisms, to better handle the complexity introduced by higher-density graphs.

IV. CONCLUSIONS

A detailed analysis of GRETEL and its variant Dual GRETEL has been presented on dense and sparse graphs derived from the WCM dataset, aiming to improve path extrapolation models. Having developed the novel dataset centered on Central Macedonia, Greece, we have provided a resource that captures the complexity of human navigational patterns on Wikipedia.

Our investigation has shown that the density of a graph significantly influences the effectiveness of path prediction methods. Both models have performed better on sparse graphs in various aspects, yet they have achieved higher accuracy with respect to the top five predictions on the dense WCM graph. Furthermore, the incorporation of hypergraph features into the GRETEL model yielding the Dual GRETEL variant has significantly enhanced the accuracy of path predictions, underscoring the importance of feature extraction in graph-based predictive analytics. Comparisons of Dual GRETEL performance on the more extensive WIKISPEEDIA dataset against the WCM dense graph have also shown that the top metrics were measured on the WCM dense graph, despite its smaller size. This indicates that the model's success is

 ISBN: 978-1-68558-138-1

TABLE V. EXAMPLES OF PATH PREDICTION

		$\Pr(s \mid h, p, G)$		$\Pr(s \mid h, p, G)$		$\Pr(s \mid h, p, G)$
prefix	**Naousa, Imathia, History of Macedonia, Craterus**		**Volvi, Egnatia, Thessaloniki, Arethousa**		**Thessaloniki, Greek National Road, Evzonoi, Axioupoli**	
true suffix	**Antigenes, Nearchus, Tlepolemus**		**Nea Madytos, Vrasna**		**Greek Macedonia, Despotate of Epirus**	
original edges	Antigenes, Nearchus, Satraps **Antigenes, Nearchus, Tlepolemus**	0.74 0.26	Stefanina, Thessaloniki **Nea Madytos, Vrasna**	0.75 0.25	Skra, Kilkis **Greek Macedonia, Despotate of Epirus**	0.38 0.01
similarity-hyperedge	Antigenes, Nearchus, Satraps **Antigenes, Nearchus, Tlepolemus**	0.64 0.36	Stefanina, Thessaloniki **Nea Madytos, Vrasna**	0.67 0.33	Skra, Kilkis **Greek Macedonia, Despotate of Epirus**	0.26 **0.03**
DHnode-in-out-degree	Antigenes, Nearchus, Satraps **Antigenes, Nearchus, Tlepolemus**	0.69 0.31	Stefanina, Thessaloniki **Nea Madytos, Vrasna**	0.78 0.22	Skra, Kilkis **Greek Macedonia, Despotate of Epirus**	0.29 0.01
similarity-hyperedge - DHnode-in-out-degree	**Antigenes, Nearchus, Tlepolemus**	**0.6**	Stefanina, Thessaloniki **Nea Madytos, Vrasna**	0.58 **0.42**	Skra, Kilkis	0.46

influenced by the quality of the graph's structure and the features used.

ACKNOWLEDGEMENTS

This research was carried out as part of the project "Optimal Path Recommendation with Multi Criteria" (Project code: KMP6-0078997) under the framework of the Action "Investment Plans of Innovation" of the Operational Program "Central Macedonia 2014-2020" that is co-funded by the European Regional Development Fund and Greece.

REFERENCES

[1] Z. Wu *et al.*, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2020.

[2] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2008.

[3] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 793–803.

[4] J. Zhou *et al.*, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[5] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[6] S. Xiao, S. Wang, Y. Dai, and W. Guo, "Graph neural networks in node classification: survey and evaluation," *Machine Vision and Applications*, vol. 33, pp. 1–19, 2022.

[7] B. Li and D. Pi, "Learning deep neural networks for node classification," *Expert Systems with Applications*, vol. 137, pp. 324–334, 2019.

[8] Y. Rong, W. Huang, T. Xu, and J. Huang, "Dropedge: Towards deep graph convolutional networks on node classification," *arXiv preprint arXiv:1907.10903*, 2019.

[9] A. Kumar, S. S. Singh, K. Singh, and B. Biswas, "Link prediction techniques, applications, and performance: A survey," *Physica A: Statistical Mechanics and its Applications*, vol. 553, p. 124289, 2020.

[10] L. Cai, J. Li, J. Wang, and S. Ji, "Line graph neural networks for link prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5103–5113, 2021.

[11] "Wikispeedia navigation paths - SNAP: Stanford," [retrieved: February 8, 2024]. [Online]. Available: http://snap.stanford.edu/data/wikispeedia.html

[12] M. Sotiroudi, A.-S. Toufa, and C. Kotropoulos, "Central Macedonia Wikipedia dataset," [retrieved: February 8, 2024]. [Online]. Available: https://tinyurl.com/5n7yd94a

[13] "Code for WCM dataset creation," [retrieved: February 9, 2024]. [Online]. Available: https://github.com/MarthaSotiroudi/Wikipedia-Central-Macedonia-Dataset

[14] J.-B. Cordonnier and A. Loukas, "Extrapolating paths with graph neural networks," *arXiv preprint arXiv:1903.07518*, 2019.

[15] J. Jaehyeong *et al.*, "Edge representation learning with hypergraphs," in *Advances in Neural Information Processing Systems*, vol. 34. Virtual Conference: Curran Associates, Inc., 2021, pp. 7534–7546.

[16] A.-S. Toufa, C. Kotropoulos, and I. Tsingalis, "Dual hypergraph features for path inference in wikipedia links," in *Proceedings of the 2023 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2023, pp. 1–7.

[17] A.-S. Toufa, I. Tsingalis, and C. Kotropoulos, "DualGRETEL+: Exploiting dual hypergraphs for path inference applied to navigation data," in *Proceedings of the 27th Pan-Hellenic Conference on Progress in Computing and Informatics with International Participation (PCI)*, 2023, pp. 1–10.

[18] C. Kotropoulos, "Multimedia social search based on hypergraph learning," in *Graph-Based Social Media Analysis*, I. Pitas, Ed. CRC Press, 2016, vol. 39, pp. 215–273.

[19] "Wikipedia Central Macedonia article," [retrieved: February 8, 2024]. [Online]. Available: https://en.wikipedia.org/wiki/Central_Macedonia

[20] S. Auer *et al.*, "Dbpedia: A nucleus for a web of open data," in *International Semantic Web Conference*. Springer, 2007, pp. 722–735.

[21] J. Ramos, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the First Instructional Conference on Machine Learning*. Citeseer, 2003, pp. 1–4.

[22] "Wikispeedia Paths & Dual Hypergraph Features repository," [retrieved: February 9, 2024]. [Online]. Available: https://github.com/asrtoufa/wikispeedia-paths-dual-hypergraph-features/tree/main

[23] M. Kempton, "Non-backtracking random walks and a weighted Ihara's theorem," *arXiv preprint arXiv:1603.05553*, 2016.

[24] "Gephi - The Open Graph Viz Platform," [retrieved: February 8, 2024]. [Online]. Available: https://gephi.org/

 ISBN: 978-1-68558-138-1

Web Components for Database Developers

Andreas Schmidt*‡ and Tobias Münch†
* University of Applied Sciences
Karlsruhe, Germany
Email: andreas.schmidt@h-ka.de
‡ Karlsruhe Institute of Technology
Karlsruhe, Germany
Email: andreas.schmidt@kit.edu
† Münch Ges. für IT Solutions mbH, Germany
Email: to.muench@muench-its.de

***Abstract*—We present a number of web components, which allow the presentation and modification of database content in a browser. The components include an element for displaying complete tables, or a portion thereof, a component for representing Structured Query Language (SQL) queries, and components that offer forms for creating and editing data records. In addition to presenting the functionality of the components, we also show how the components can be embedded in websites.**

***Index Terms*—Web Component; Relational-Database; Interface; Prototyping**

I. Introduction

The main purpose of the components presented here is the visual representation and manipulation of dynamic database content within websites.

Nowadays, web frameworks, such as angular or react are usually used for this purpose. These frameworks provide the experienced developer with a wide range of tools for realizing complex applications. Advantages include shorter development times, a more consistent code base, sophisticated security features and various scaling options [1].

But there are also a number of disadvantages, you have to consider. On the one hand, this concerns the complexity of the frameworks, which require a long familiarization period before they can be used productively. Due to the complexity, developers usually limit themselves to one framework, which makes them dependent to a certain extent on the continued existence of the framework [1].

These frameworks are often simply oversized when it comes to simple applications in the scientific and technical field. Here, it is often a matter of visualizing data in tabular form, searching and possibly manipulating it. In such applications, web components [2] can be a good alternative to web frameworks. Web components are a recommendation of the W3C and are now supported by all common browsers.

The structure of the abstract is as follows: First, web components are introduced in Section II, afterwards the overall architecture is presented in Section III, before examples of the implemented components are given in Section IV. This includes the integration into the HTML pages as well as partly the visual representation. The concluding Section V provides an outlook on further planned work on our components.

II. Web Components

At the heart of the web components are the custom elements, which allow the definition of user-specific HTML tags that bundle their own user interface and the associated logic.

The components inherit from the HTML-element class and implement a series of methods that are then called later when the components are added to the Document Object Model (DOM) tree.

III. Architecture of the Database web components

While the web components run in the browser, they have to communicate with a database server. The developed web components don't communicate directly with the database, but through a thin Representational State Transfer (REST)-based access layer (see Fig. 1). This service maps a logical database identifier to a specific database on the server side.

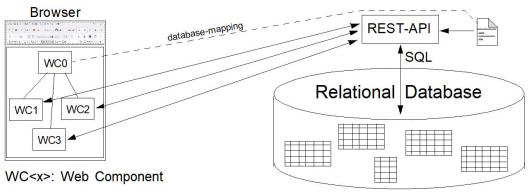

Fig. 1. Architecture of our database web components.

Beside the data, the web components also request metadata about the table from the database. The metadata is used, among other things, to construct the forms for a data set.

IV. Components

This section presents the web components we have developed so far. The Mondial [3] database was used for the examples (screenshots and SQL queries).

A. Table-component

The *db-table* component (Fig. 2) is responsible for displaying the content of a database table. It also allows navigation (scrolling) within the datasets (1), sorting by column values (2) and the formulation of additional conditions on the datasets (3).

 ISBN: 978-1-68558-138-1

First Prev 1 / 2 (15) Next Last ①

② ↑ Area

Name	Code	Capital	Province	Area	Population
					< 100000 ③
Holy See	V	Vatican City	Holy See		840
Monaco	MC	Monaco	Monaco	2	31719
Nauru	NAU	Yaren	Nauru	21	10273
Tuvalu	TUV	Funafuti	Tuvalu	26	10146
San Marino	RSM	San Marino	San Marino	60	24521
Liechtenstein	FL	Vaduz	Liechtenstein	160	31122
Marshall Islands	MH	Majuro	Marshall Islands	181	58363
Saint Kitts and Nevis	KN	Basseterre	Saint Kitts and Nevis	269	41369
Grenada	WG	Saint Georges	Grenada	340	94961
Antigua and Barbudas	AG	Saint Johns	Antigua and Barbuda	440	65647

First Prev 1 / 2 (15) Next Last

Fig. 2. web component *Table*

The corresponding HTML-code is shown in Listing 1.

Listing 1. Embedding of db-table component in a web-page

```
<db-table  table="country"
           connection="mondial"
           pagesize="10">
</db-table>
```

B. Dataset-component

The *db-dataset* component is responsible for displaying a single dataset and optionally editing it. Fig. 3 shows the dataset of "France".

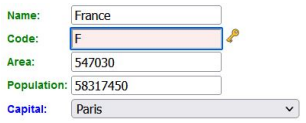

Fig. 3. web component *db-dataset*

Listing 2 shows how the dataset "France" can be embedded in a page. The parameter `key` expects the value of the primary key of the dataset. The name of the primary key attribute does not have to be specified as it is determined from the metadata of the table.

Listing 2. db-dataset component in a web-page

```
<db-dataset  table-name="country"
             key="F">
</db-dataset>
```

C. Selection-component

The *db-select* web component (Fig. 4) shows a selection box, from which values can be selected and searched via a prefix search. The values are specified by an SQL-select statement. The SQL-statement can either be specified directly by the `sql`-attribute, or it is specified using the attributes `code`, `value`, `table` and (optional) `filter`. Listing 3 gives examples of the embedding into a page.

Listing 3. Examples of *db-select* components in a web-page

```
<db-select  table="country"
            key="Code"
            value="Name">
</db-select>
```

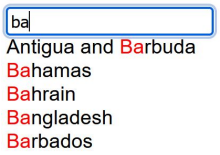

Fig. 4. web component *db-select*

D. Query-Component

The visual representation of the *db-query* component is similar to that of the *db-table* component in Fig. 2. The main difference is that no `table` parameter is specified, but an arbitrary SQL select-statement. Listing 4 shows an example in which the SQL statement determines the number of cities with more than one million inhabitants in the individual countries.

Listing 4. db-query component in a web-page

```
<db-query  sql="
      select  co.name,
              count(*)  num_big_cities
        from  city  ci
        join  country  co
          on  co.code=ci.country
       where  ci.population > 1000000
       group  by  ci.country, co.name
       order  by  2  desc"
           pagesize="10">
</db-query>
```

E. Server-component

The server component is a non-visible component in a page. It is responsible for the mapping to a concrete database on server-side. Listing 5 gives an example, how the web component is integrated inside a HTML page. The *db-server* component communicates with a RESTful service which is specified by the parameter `url`. The service, which is written in PHP, is then also responsible for mapping and accessing the specific database (specified by the parameter `database`).

Listing 5. Specification of a db-server component inside a web-page

```
<db-server
      url="https://dbkda.smiffy.de/mondial/dbwc"
      database="mondial-2010">
</db-server>
```

 ISBN: 978-1-68558-138-1

V. Conclusion and Outlook

We have implemented the first prototype for our web components and are currently in the process of expanding the components so that all meta information available in the database is also evaluated in the components. This can already be seen in Fig. 3, where a selection box with the dereferenced value (Paris) is displayed instead of the foreign key for the capital. We are also planning to extend the *db-dataset* component so that it not only provides its own form, but can also handle external forms. This will make it possible to completely customize the layout to your preferences or requirements. Further work concerns aspects of security, such as authentication and authorization.

References

[1] G. Juste. Exploring the Advantages and Disadvantages of Incorporating Frameworks into Web Development. https://www.linkedin.com/pulse/exploring-advantages-disadvantages-incorporating-frameworks-juste/. Last accessed 06.02.2024.

[2] https://www.webcomponents.org/specs. Last accessed 06.02.2024.

[3] W. May. Information Extraction and Integration with Florid: The Mondial Case Study, Universität Freiburg, Institut für Informatik, 1999, http://dbis.informatik.uni-goettingen.de/Mondial. Last accessed 06.02.2024.

 ISBN: 978-1-68558-138-1