Graph Databases: A Survey on Models, Data Modeling, and Applications

www.surveyx.cn

Abstract

Graph databases, designed to efficiently manage and query interconnected data structures, are increasingly vital in scenarios where relationships between data points are as crucial as the data itself. This survey paper systematically examines the concept, importance, and applications of graph databases across various domains. It highlights their advantages over traditional relational databases, particularly in handling complex, non-Euclidean data structures and dynamic graph streams. The paper explores diverse graph database models, including Property Graphs and RDF Graphs, and innovative architectures that optimize data processing and query execution. Key applications are discussed, such as social network analysis, recommendation systems, scientific research, and industrial real-time data processing. Challenges related to scalability, performance, query optimization, security, and privacy are identified, alongside future directions for integration with emerging technologies and enhancing computational efficiency. The survey concludes by emphasizing the transformative impact of graph databases in modern data-intensive environments, underscoring their role in advancing data management and analytics capabilities. By leveraging their structural advantages and adaptability, graph databases continue to evolve, meeting the growing demands of complex data scenarios.

1 Introduction

Graph databases have emerged as a pivotal technology in the management of complex, interconnected data structures. As the volume of data generated across various domains continues to increase exponentially, traditional relational database management systems (RDBMSs) often struggle to efficiently handle the intricate relationships inherent in this data. Graph databases, characterized by their unique data representation involving nodes, edges, and properties, offer a robust framework for modeling and querying these relationships. This review paper aims to explore the multifaceted landscape of graph databases, examining their concepts, motivations for adoption, relevance in data-intensive scenarios, and the overarching structure of the survey. By providing a comprehensive analysis of these aspects, the paper seeks to elucidate the significance of graph databases in contemporary data management and their potential to address the challenges posed by traditional database systems.

1.1 Concept and Importance of Graph Databases

Graph databases are specialized systems designed to efficiently store, manage, and query data structured as nodes, edges, and properties, representing entities, relationships, and attributes. They play a critical role in managing interconnected data, offering expressive query semantics that provide detailed insights into complex relationships embedded within graph data [1]. This capability is increasingly important due to the rapid growth of graph data from diverse real-world applications, making graph querying a critical area of research and application [2]. The unique architecture of graph databases allows for the representation of data in a way that mirrors real-world relationships,

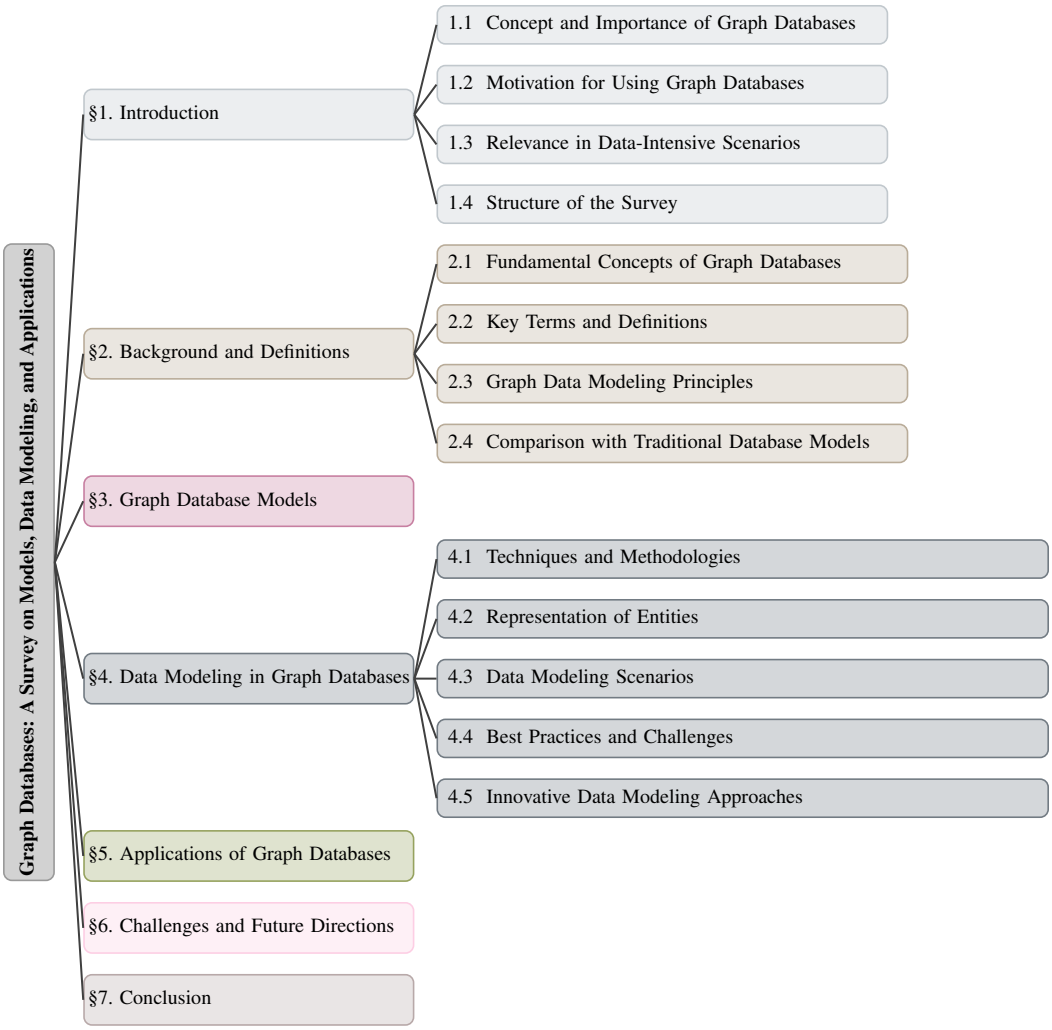


Figure 1: chapter structure

making them particularly effective for applications that require a nuanced understanding of data interconnectivity.

Graph databases are particularly advantageous in handling complex and non-Euclidean data structures, efficiently managing interconnected relationships prevalent in such datasets [3]. This efficiency is crucial in the realm of big data, where managing complex relationships among vast amounts of data points is required [4]. The surge in popularity of graph data underscores its ideal structure for capturing interactions across various domains, enhancing the utility of graph databases in diverse applications [5]. As organizations increasingly rely on data-driven decision-making, the ability to navigate and query complex relationships becomes paramount, further solidifying the role of graph databases in modern data management.

In scenarios involving dynamic graph streams, where edges are frequently added or removed, graph databases are indispensable for robust management of interconnected data [6]. As the volume of graph data continues to grow, challenges in graph management emerge, further emphasizing the essential role of graph databases [7]. Their importance is highlighted by the increasing popularity across application domains such as social networks and the semantic web, where relational perspectives of graph querying are crucial [8]. The adaptability of graph databases to various contexts, whether it be real-time analytics or historical data analysis, showcases their versatility and relevance in today's data-centric landscape.

Graph databases facilitate robust querying mechanisms capable of handling both data attributes and structural relationships, thereby enhancing their expressiveness and applicability in managing

AI-generated, for reference only.

interconnected data [9]. They are utilized in the proposed hybrid model to enable domain knowledge acquisition by intelligent systems, showcasing their adaptability in various contexts [10]. Graph databases are also pivotal in the context of data science lifecycle provenance, where managing interconnected data is essential [11]. Furthermore, they play a significant role in visualizing complex relationships within scholarly data, as demonstrated by the use of Neo4j. The integration of graph databases into various applications not only enhances data management but also fosters innovative approaches to problem-solving across disciplines.

Graph laws are significant for modern graph representation learning, particularly in relation to the development of foundation models [12]. The integration of heterogeneous information for better decision-making in industries such as wind energy further exemplifies the importance of graph databases [13]. Additionally, graph databases are proposed to integrate graph capabilities into traditional RDBMSs to enhance their effectiveness [14]. Graph Neural Networks (GNNs) address data isolation issues, where high-quality graph data is held by multiple data holders, highlighting the importance of graph databases in managing interconnected data [15]. Finally, graph databases represent relationships in data more effectively and are widely adopted in causal inference using causal directed acyclic graphs (cDAGs) [16]. The advancements in graph database technologies continue to push the boundaries of what is possible in data management and analysis, making them an essential tool in the modern data ecosystem.

Moreover, graph databases are crucial for efficiently managing complex relationships, particularly in the context of subgraph matching, which is an NP-Complete problem [17]. They are essential for managing complex datasets such as those involved in phylogenetic analyses, particularly in understanding the evolution of pathogens [18]. Thus, graph databases are a pivotal component in modern data-intensive environments, where understanding and leveraging relationships between data points is as crucial as the data itself [19]. This multifaceted capability of graph databases positions them as a cornerstone technology in addressing the challenges of data complexity and interconnectivity.

1.2 Motivation for Using Graph Databases

The adoption of graph databases is driven by their superior ability to manage and analyze complex, interconnected data structures, which traditional relational databases often struggle with. Graph databases are particularly advantageous in scenarios where relationships between data points are as crucial as the data itself, such as in causal analysis, where property graphs serve as foundational models to enhance scalability and efficiency [16]. The limitations of existing RDBMSs in handling complex relationships, especially many-to-many relationships and recursive joins, further highlight the motivation for transitioning to graph databases [14]. As data continues to grow in complexity and volume, the need for a more flexible and efficient data management solution becomes increasingly apparent.

In the realm of scholarly data visualization, the need for graph databases is underscored by the challenges of visualizing and analyzing vast amounts of data, where traditional tabular methods fall short [20]. This necessity is echoed in the domain of wind energy, where existing condition-based monitoring methods fail to provide contextualized information and effective decision support, thus motivating the use of knowledge graph databases [13]. Moreover, graph databases offer a robust solution for privacy concerns, employing mechanisms like the kt-safe graph to ensure protection against attacks exploiting both attributes and structures [21]. This multifaceted approach to data privacy and security further enhances the appeal of graph databases in sensitive applications.

The optimization of subgraph isomorphism solutions, as demonstrated by the l2Match algorithm, further illustrates the need for graph databases to enhance efficiency in complex data operations [17]. In large-scale phylogenetic analyses, graph databases like Neo4j are utilized to overcome the limitations of traditional relational databases, providing modular frameworks such as phyloDB [18]. Additionally, the development of graph laws addresses the under-explored area of graph parametric representation, facilitating a deeper understanding of graph properties and motivating their use in various applications [12]. The continuous evolution of graph database technologies indicates a growing recognition of their capabilities in addressing contemporary data challenges.

Graph databases are instrumental in effectively learning representations of graph-structured data, addressing the complexities and scale of modern graph datasets [19]. These capabilities make graph

databases indispensable in environments where interconnected data structures are prevalent, providing enhanced decision-making support and efficient data management. As the landscape of data continues to evolve, the motivation for adopting graph databases will likely grow, driven by their ability to offer innovative solutions to the challenges posed by traditional data management systems.

1.3 Relevance in Data-Intensive Scenarios

Graph databases demonstrate significant utility in data-intensive scenarios by efficiently managing complex relationships and large-scale datasets. In environments such as Wireless Multimedia Sensor Networks (WMSNs), they excel in real-time processing of diverse and voluminous data formats, offering robust solutions for dynamic data management [22]. Their integration with Large Language Models (LLMs) further enhances their capability to handle complex reasoning tasks, providing sophisticated semantic understanding and overcoming limitations of traditional models [23]. However, recent advancements in LLMs have shown limitations in effectively processing and utilizing the structural information inherent in Topological Attribute Graphs (TAGs), highlighting the need for graph databases in such contexts [24]. This intersection of technologies illustrates the growing importance of graph databases in facilitating advanced data processing capabilities.

In the domain of cosmology, graph databases are indispensable for analyzing petabytes of data generated from extensive optical surveys, providing efficient algorithms to extract meaningful insights from vast datasets [25]. Similarly, in the financial sector, they facilitate computationally efficient handling of large equity datasets, enabling intricate queries related to Environmental, Social, and Governance (ESG) integration for informed decision-making [26]. The ability to manage and analyze large datasets with complex relationships is crucial in these fields, where timely and accurate insights can significantly impact decision-making processes.

Graph databases also play a crucial role in bioinformatics and social networks, where property graph models are utilized to represent intricate data structures and relationships, underscoring their relevance in data-intensive contexts [27]. In phylogenetic analyses, they integrate genomic and epidemiological data, proving particularly useful in managing complex relationships [18]. Moreover, in session-based recommendation systems, graph databases provide real-time, relevant recommendations by efficiently managing session data and eliminating the need for periodic retraining [28]. These applications exemplify the versatility and effectiveness of graph databases in addressing diverse data management challenges.

The potential of graph databases in broadening the scope of topological data analysis (TDA) across scientific fields further illustrates their importance in enhancing the understanding and representation of complex data structures [29]. They address challenges faced by LLMs in managing entire code repositories, offering solutions for scenarios requiring handling long-context inputs and complex reasoning tasks [30]. The increasing reliance on graph databases in various domains signifies their growing importance as a foundational technology in contemporary data management practices.

Finally, the benchmarking of graph databases in simulating real-world social network interactions highlights their performance in executing complex queries and analytics, reinforcing their applicability in scenarios demanding high computational efficiency and data management capabilities [31]. Additionally, graph databases facilitate the automation of data storytelling in intelligence analysis by enabling semantic analysis and entity disambiguation, which are critical for managing complex relationships [32]. The ability to effectively navigate and analyze intricate data landscapes positions graph databases as a key player in the future of data-driven decision-making processes.

1.4 Structure of the Survey

This survey is systematically organized into seven primary sections, each focusing on critical aspects of graph databases. Section 1 introduces graph databases, emphasizing their importance in managing complex datasets and outlining the motivation for their utilization. It further discusses their relevance in data-intensive scenarios and provides an overview of the survey's structure. Section 2 delves into the background and definitions, offering a comprehensive overview of graph databases, key terms, and the principles of graph data modeling, while contrasting them with traditional database models. Section 3 explores various graph database models, comparing property graphs and RDF graphs, and discussing innovative models and architectures. Section 4 focuses on data modeling in graph databases, detailing techniques and methodologies, representation of data elements, and

providing examples of data modeling scenarios, alongside best practices and challenges. Section 5 examines the applications of graph databases across various domains, highlighting their role in social networks, recommendation systems, knowledge graphs, scientific research, and industrial applications. Section 6 identifies current challenges and future directions in the field, addressing scalability, performance, query optimization, security, and integration with emerging technologies. Finally, Section 7 concludes the survey by summarizing the key points discussed and emphasizing the significance and future potential of graph databases. This structured approach aims to provide readers with a comprehensive understanding of the evolving landscape of graph databases and their implications for future research and application.The following sections are organized as shown in Figure 1.

2 Background and Definitions

2.1 Fundamental Concepts of Graph Databases

Graph databases are structured around nodes, edges, and properties, forming a flexible framework for modeling complex datasets. Nodes represent entities, edges define their connections, and properties provide detailed attributes, facilitating comprehensive graph data representation [5]. This architecture effectively manages complex relationships, addressing challenges that traditional data models face [33]. Their flexibility makes them ideal for applications in social networks, recommendation systems, and knowledge graphs, where relationships are as significant as the entities themselves.

Graph databases adeptly handle dynamic datasets, requiring continuous updates to maintain accurate embeddings in dynamic knowledge graphs [34]. This is crucial in distributed environments, necessitating strategies for managing isolated graph data [15]. Their dynamic nature makes them suitable for scenarios with voluminous and frequently changing data.

Advanced query languages like Cypher enable complex querying and manipulation of graph structures [9], while graph transformation languages like UnCAL support recursive transformations [35]. The integration of Graph Neural Networks (GNNs) in Graph Transformer models enhances performance across applications [36]. These capabilities allow sophisticated analyses often unattainable with traditional databases, expanding graph databases' potential in machine learning and AI.

Visualization tools, such as those in Neo4j, manage massive datasets by offering rich relational data representation. Systems like GRainDB emphasize defining node and edge labels, modeling relations as nodes, and visualizing results in tables and graphs [14]. These tools enhance understanding of complex data structures and facilitate exploratory data analysis, crucial for data-driven decision-making.

Graph databases address privacy concerns with mechanisms like kt-safe graph release, protecting against attribute and structure-based attacks [21]. They also optimize subgraph isomorphism solutions, critical in graph database optimization [17]. This focus on privacy and performance enhances their robustness, making them suitable for sensitive applications in domains like healthcare and finance.

The architecture of graph databases, exemplified by phyloDB, supports efficient querying and management of complex data structures [18]. These core concepts underscore their pivotal role in modern data management, enabling efficient handling and analysis of complex relational data structures across diverse domains. As demand for sophisticated data management solutions rises, graph databases are increasingly recognized as essential tools for data scientists and analysts.

2.2 Key Terms and Definitions

Key elements of graph databases include nodes, edges, and properties. Nodes represent entities, edges depict relationships, and properties offer additional context, forming the basis for complex queries and analyses [37, 12]. Understanding these terms is crucial for grasping graph databases' operation and utility.

Data in graph databases is organized into models like RDF graphs and Labeled Property Graphs (LPGs). RDF graphs emphasize semantic representation, enabling complex queries through SPARQL, while LPGs focus on property-based queries using languages such as Cypher [37]. This versatility accommodates various data representation needs and querying methodologies, influencing performance and efficiency.

Graph laws and concepts, such as macroscopic and microscopic views, and static versus dynamic graphs, are crucial for understanding graph data's diverse nature [12]. These concepts aid in analyzing graph structure and behavior, facilitating tasks like classification and regression, as seen in datasets like TUDataset [37]. Mastery of these patterns enhances query execution efficiency and empowers nuanced data analyses.

Graph databases employ graph patterns and navigational expressions to explore complex relationships, addressing issues like Conjunctive Regular Path Queries (CRPQs) containment [12]. These techniques improve query optimization and information retrieval, essential for user experience and performance enhancement.

Understanding these terms and definitions is vital for leveraging graph databases' full potential in applications from data analysis to machine learning. As the field evolves, a solid grasp of these concepts enables practitioners to harness graph databases' capabilities, driving innovation across domains.

2.3 Graph Data Modeling Principles

Graph data modeling principles are crucial for representing complex interactions within datasets, diverging from traditional models that focus on static data points. A pivotal aspect is specifying transformations within causal property graphs, incorporating hypernodes and causal semantics [16]. This integration facilitates causal analysis, enriching graph models' expressiveness.

Multimodal knowledge graphs, exemplified by XAI4Wind, interlink diverse data types for enhanced decision support, showcasing graph data models' flexibility [13]. This multimodal approach synthesizes information from various sources, providing a holistic data view, valuable in complex domains requiring integrated information.

Techniques for deep graph embeddings and relational structures capture intricate graph data relationships [19]. These techniques enhance data representation and analysis, paving the way for advanced machine learning applications leveraging graph structures. The development of graph embeddings facilitates meaningful feature extraction, enabling effective model learning and generalization.

Graph data modeling also tackles schema-driven generation of synthetic graph instances and query workloads, as highlighted by benchmarks like GMARK [33]. This approach enables rigorous testing and validation of graph database systems, advancing understanding of performance and capabilities.

The complexity of graph data, involving sensitive attributes and vertex relationships, necessitates advanced modeling techniques for privacy and security, as demonstrated by kt-safe graph release mechanisms [21]. Ongoing development of privacy-preserving methodologies reflects data security's growing importance. Implementing robust privacy mechanisms fosters trust among users.

These principles underscore innovative, scalable methodologies designed for graph data's distinct attributes. They offer frameworks for representing, querying, and analyzing interconnected datasets, setting graph data models apart from traditional approaches. This differentiation is evident in graph summarization techniques, condensing data into insights, and graph query operators enhancing provenance information understanding. Tools like GrapAL exemplify graph databases' application in scientific literature exploration, illustrating graph-based approaches' versatility [11, 38, 39].

2.4 Comparison with Traditional Database Models

Graph databases offer advantages over traditional relational models, particularly in managing complex, interconnected structures. Traditional databases use tabular formats and fixed schemas, which can be rigid and inefficient for dynamic datasets [40]. Graph databases, using nodes, edges, and properties, naturally represent and query interconnected data, offering a more flexible approach [41].

Traditional models struggle with large graph workloads due to value-based joins, unlike the predefined joins in graph database management systems (GDBMSs) [40]. Graph databases, especially native ones, optimize storage and support massively parallel processing, as seen in TigerGraph [4]. These optimizations are crucial for scalability and efficient graph data visualization, positioning graph databases as superior for real-time analysis and insights.

Graph databases are categorized into native versus hybrid and graph-only versus converged, highlighting their strengths and weaknesses. Native databases optimize graph data processing, while hybrids integrate graph capabilities into relational systems, balancing flexibility and performance [10]. Innovations like hash-based messaging mechanisms enable scalable computation of graph summaries, enhancing processing efficiency [42].

Graph databases excel in handling multimodal information, as VGStore demonstrates with SPARQL extensions for visual and spatial components [43]. This contrasts with traditional RDBMSs, which struggle to integrate structured and unstructured data effectively [44]. Graph databases' proficiency in this area enhances their applicability across domains, from social media analysis to scientific research.

They also provide robust semantic query solutions, overcoming challenges like CONSTRUCT queries' semantics and Regular Path Queries (RDPQs) containment. Systems like GRainDB exemplify graph databases' enhanced performance and scalability for complex queries [14]. These capabilities are essential for addressing modern graphs' computational restrictions, often exceeding traditional systems' capacities [45].

Graph databases offer flexibility, query capabilities, and scalability for modern applications, distinguishing them from traditional models. Their proficiency in managing intricate relationships, integrating data types, and facilitating advanced query languages makes them essential in contemporary data ecosystems, enabling non-technical users to navigate complex databases with natural language interfaces, enhancing accessibility in an era of unstructured big data and sophisticated analytics [9, 46, 47, 39].

3 Graph Database Models

To effectively explore the intricacies of graph database models, it is essential to first delineate the foundational concepts that underpin their functionality and structure. This section will elucidate the various types of graph database models, highlighting their distinct characteristics and applications. Understanding these models serves as a critical precursor to analyzing their comparative strengths and weaknesses, particularly in relation to their performance and interoperability. Therefore, we commence with an examination of the different types of graph database models, which include Property Graphs and RDF Graphs, each offering unique advantages tailored to specific data management needs.

As illustrated in Figure 2, this figure shows a hierarchical classification of graph database models, effectively categorizing them into Property Graphs and RDF Graphs. It contrasts their strengths and explores cutting-edge models and architectures that enhance functionality and efficiency in handling complex datasets. The exploration of these models not only provides insights into their operational mechanisms but also sets the stage for a deeper investigation into their practical implications and future developments in the field of data management.

3.1 Types of Graph Database Models

Graph database models are integral to the efficient management and analysis of complex datasets, offering a variety of structures and functionalities tailored to specific application needs. Among the most prominent models are Property Graphs and RDF (Resource Description Framework) Graphs. Property Graphs are characterized by their ability to attach properties to both nodes and edges, allowing for detailed data representation and complex querying. This model is particularly advantageous in domains such as social networks and recommendation systems, where capturing intricate relationships is essential [18]. The versatility of Property Graphs is further enhanced by their compatibility with advanced graph processing frameworks like Gremlin, which supports both OLTP and OLAP graph processing, thereby broadening their applicability across various graph database systems [17].

RDF Graphs, on the other hand, focus on semantic data representation, facilitating interoperability across diverse data sources through standardized vocabularies and ontologies. This model is particularly suited for applications requiring semantic queries and the integration of heterogeneous data types, as it supports complex queries through languages like SPARQL. The RDF model's emphasis on semantic richness is complemented by innovative database models like VGStore, which enhance

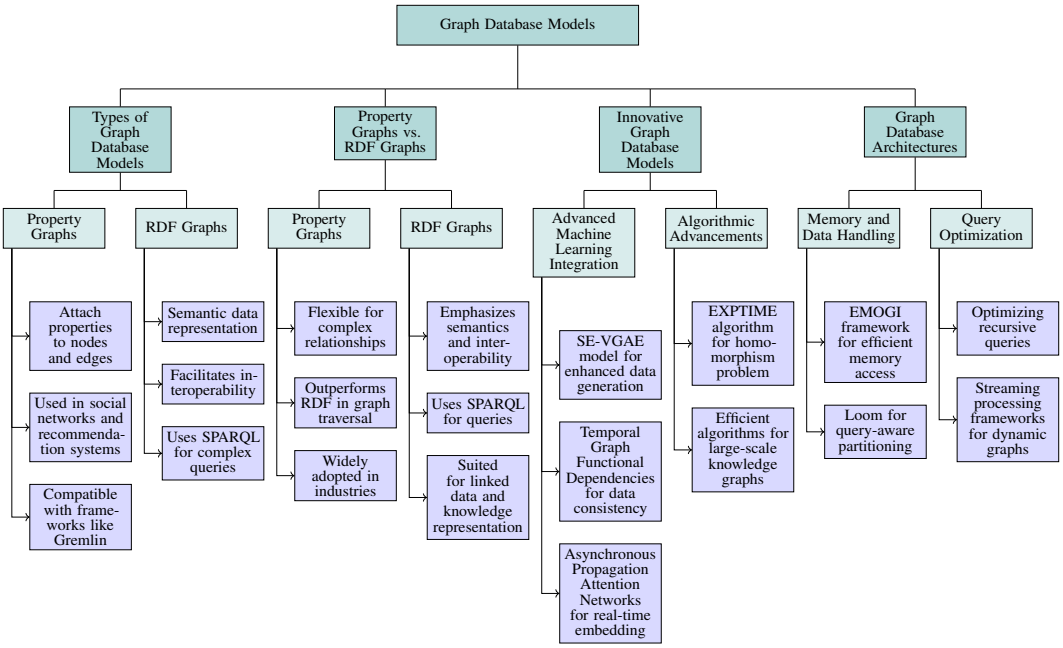


Figure 2: This figure shows a hierarchical classification of graph database models, highlighting the types, comparative analysis, innovative advancements, and architectural designs. It categorizes graph database models into Property Graphs and RDF Graphs, contrasts their strengths, and explores cutting-edge models and architectures that enhance functionality and efficiency in handling complex datasets.

SPARQL queries by integrating custom predicates for multimodal querying capabilities [33]. The ability of RDF Graphs to interlink disparate data sources underscores their significance in the era of big data, where the integration of varied datasets is paramount for comprehensive analysis.

Emerging graph database models continue to expand the capabilities of graph databases. Dynamic knowledge graphs necessitate efficient updating methods to accurately reflect alterations in their structure, highlighting the adaptability of graph database models to continually evolving datasets. For example, embedding techniques used in machine learning require frequent updates to maintain the relevance of node representations, as full retraining can be time-consuming and inefficient. Recent research has explored methods for updating embeddings without complete retraining, allowing for faster maintenance of dynamic knowledge graphs. Additionally, optimizing retrieval algorithms on large-scale knowledge graphs has proven essential for effective knowledge mining and discovery, particularly in fields like bioinformatics, where performance improvements have been demonstrated through innovative optimization approaches. This adaptability and efficiency are crucial for applications such as the GrapAL database, which leverages graph technology to facilitate diverse research needs and enhance the exploration of scientific literature [34, 39, 48]. Furthermore, federated learning paradigms such as VFGNN are specifically designed for privacy-preserving node classification tasks in scenarios where data is vertically partitioned across multiple parties, highlighting the innovative approaches in graph database model development.

The integration of graph database models with advanced architectural designs, particularly Graph Transformer models, significantly enhances query and analytics performance by leveraging innovative techniques such as improved positional embeddings and attention mechanisms tailored for graph-structured data. This synergy not only facilitates more efficient data retrieval and processing but also addresses the complexities of graph tasks across various domains, as demonstrated by comprehensive evaluations of these models on benchmark datasets [36, 23, 49, 50]. These models incorporate graph information into traditional Transformer architectures, improving their ability to handle complex graph data. Moreover, specialized models for specific applications, such as frameworks leveraging Graph Pattern Calculus (GPC) to express transformations, demonstrate the adaptability of graph databases to various domains.

The diverse models and methodologies discussed in this subsection underscore the scalability and versatility of graph databases, making them indispensable tools for managing complex, interconnected datasets across various domains. Continuous advancements and the incorporation of innovative models and querying techniques highlight the inherent flexibility of graph database architectures, enabling them to effectively address the dynamic requirements of contemporary data-intensive applications. This adaptability is exemplified by the integration of diverse metadata through property graphs and the development of user-friendly query languages, as seen in tools like GrapAL, which facilitate the exploration of scientific literature [50, 38, 49, 39].

As illustrated in Figure 3, the primary categories of graph database models, including Property Graphs, RDF Graphs, and Emerging Models, highlight their key features and applications in complex data management and analysis.

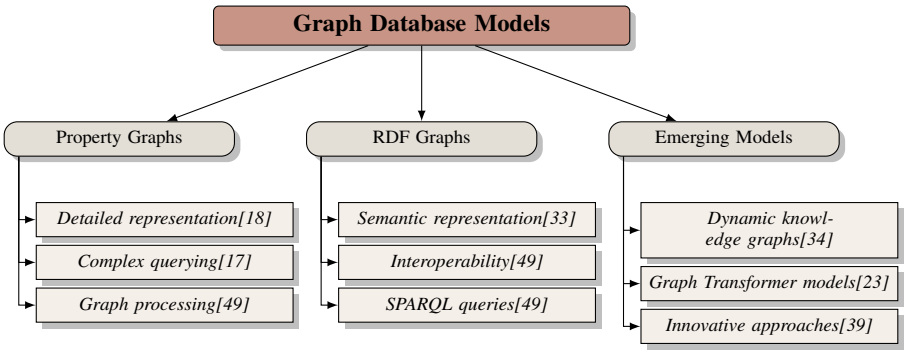


Figure 3: This figure illustrates the primary categories of graph database models, including Property Graphs, RDF Graphs, and Emerging Models, highlighting their key features and applications in complex data management and analysis.

3.2 Property Graphs vs. RDF Graphs

Property Graphs and RDF (Resource Description Framework) Graphs represent two distinct paradigms in graph database models, each with unique structures and applications. Property Graphs are characterized by their ability to attach properties to both nodes and edges, offering a flexible model for representing complex relationships and attributes. This model is particularly advantageous in scenarios requiring intricate graph traversal and querying, as it generally outperforms RDF databases in terms of graph traversal query performance [51]. The industry-wide adoption of the property graph model underscores its effectiveness in handling diverse applications, from social networks to recommendation systems [52]. The practical implications of Property Graphs are evident in their widespread use in various industries, where the ability to model complex relationships is crucial for deriving insights from interconnected data.

In contrast, RDF Graphs focus on semantic data representation, emphasizing interoperability and standardization through vocabularies and ontologies. RDF Graphs leverage SPARQL for querying, facilitating complex semantic queries across heterogeneous data sources [53]. The semantic richness of RDF models is further enhanced by mappings that ensure semantics and information preservation, demonstrating that property graph models can subsume the information capacity of RDF models [53]. Declarative languages like G2GML are employed to specify mappings between RDF and property graph patterns, highlighting the adaptability of RDF models in various contexts [54]. The emphasis on semantics in RDF Graphs makes them particularly suited for applications in fields such as linked data and knowledge representation, where the meaning of data is as important as the data itself.

The absence of a common algebra for path queries remains a significant obstacle in graph query engines, affecting both RDF and property graph models [52]. However, approaches such as extensions of relational calculus have been proposed to study graph database querying, providing optimal data complexity bounds and enhancing the expressive power of graph query engines [55]. This ongoing research into query optimization and algebraic foundations is critical for advancing the state of graph databases, as it addresses the fundamental challenges that hinder their performance and usability.

Innovative methods like FAST, which accelerates subgraph matching through optimized task parallelism on FPGAs, further illustrate the potential of property graphs in computational efficiency [56]. Similarly, the development of index structures like WKRI for efficient k-step weighted reachability query processing exemplifies the advancements in graph database models to support complex queries [57]. These advancements not only enhance the performance of property graphs but also contribute to the overall evolution of graph database architectures, ensuring that they remain relevant in an increasingly data-driven world.

The comparison between Property Graphs and RDF Graphs reveals the strengths and limitations inherent in each model. While Property Graphs excel in performance and flexibility, RDF Graphs provide robust semantic capabilities and interoperability. This nuanced understanding of their respective advantages is essential for practitioners and researchers when selecting the appropriate model for specific applications. The ongoing development of hybrid approaches that combine the strengths of both models may pave the way for future innovations in graph database technology, ultimately enhancing data management practices across various domains.

3.3 Innovative Graph Database Models

Innovative graph database models have been developed to address the complexities of dynamic and temporal datasets, providing advanced methodologies for real-time data processing and sophisticated query handling. Among these, the SE-VGAE model stands out by integrating a transformer-based encoder with a style-based decoder, significantly enhancing the fidelity and diversity of generated architectural layouts [58]. This approach exemplifies the potential of leveraging advanced machine learning techniques in graph database models to improve data representation and generation. The integration of such cutting-edge technologies not only enhances model performance but also expands the applicability of graph databases in various fields, including architecture and design.

Temporal Graph Functional Dependencies (TGFDs) represent another notable advancement in graph database models, offering the capability to model temporal data consistency by enforcing value dependencies conditioned on both topological and temporal constraints [59]. This innovation is particularly useful in applications where the temporal dimension of data is crucial, ensuring that the integrity and consistency of data are maintained over time. The ability to address temporal aspects in graph databases is increasingly important, especially in domains such as social media analysis and event tracking, where the timing of interactions can significantly influence outcomes.

Dynamic graph models have also seen significant progress, particularly in the context of real-time data environments. Techniques such as Asynchronous Propagation Attention Networks (APAN) significantly improve the processing capabilities of graph databases by utilizing asynchronous attention mechanisms that enable real-time temporal graph embedding. This approach effectively addresses the limitations of traditional graph algorithms, which often struggle with high time complexity during k-hop neighbor queries, thereby hindering their deployment in time-sensitive applications like financial fraud detection. By decoupling graph computation from model inference, APAN enhances the speed of model inference—achieving up to an 8.7 times improvement—while maintaining competitive performance across various tasks. This makes it particularly suitable for scenarios involving continuously evolving data that demands rapid updates and efficient query responses [60, 61, 62, 63]. These models are instrumental in environments that demand real-time insights and adaptive data management strategies.

Recent advancements in graph theory include the introduction of efficient algorithms specifically designed for addressing complex graph-related problems, such as the EXPTIME algorithm for the homomorphism problem. This algorithm significantly enhances solution efficiency compared to traditional methods, which is crucial for optimizing performance in applications like knowledge graph retrieval and data summarization. For instance, in large-scale knowledge graphs, innovative optimization techniques have demonstrated impressive speedups ranging from 44 to 3839 times faster than naive querying approaches, highlighting the importance of algorithmic efficiency in managing vast interconnected data [38, 48]. These advancements highlight the ongoing efforts to optimize computational efficiency in graph databases, addressing the challenges posed by complex query requirements and large datasets.

The advancement of temporal graph models, particularly the Time-Varying Graph (TVG) model, facilitates the effective representation and analysis of networks that undergo changes over time. This

evolution allows researchers to gain deeper insights into the temporal dynamics of data, such as the intricate relationships between the topological structure of transit networks and the mobility patterns of vehicles, as demonstrated in studies utilizing TVGs for transit management. By employing network metrics like temporal shortest paths and various centralities, these models reveal significant dynamics in evolving networks, enhancing our understanding of complex systems across diverse applications, from urban transit to biological networks [64, 23, 65, 66, 67]. This capability is crucial for applications that require an understanding of how relationships and structures evolve, providing a more comprehensive view of the data landscape.

The emergence of these innovative models and methodologies underscores the dynamic nature of graph database development, emphasizing their adaptability and effectiveness in managing complex, interconnected datasets across various domains. The ongoing integration of cutting-edge techniques, such as advanced graph summarization methods and efficient historical data management systems, ensures that graph databases not only maintain their competitive edge in data management solutions but also effectively address the increasing complexity and demands of modern data-intensive applications, including social networks and scientific literature analysis [49, 10, 68, 39, 38].

3.4 Graph Database Architectures

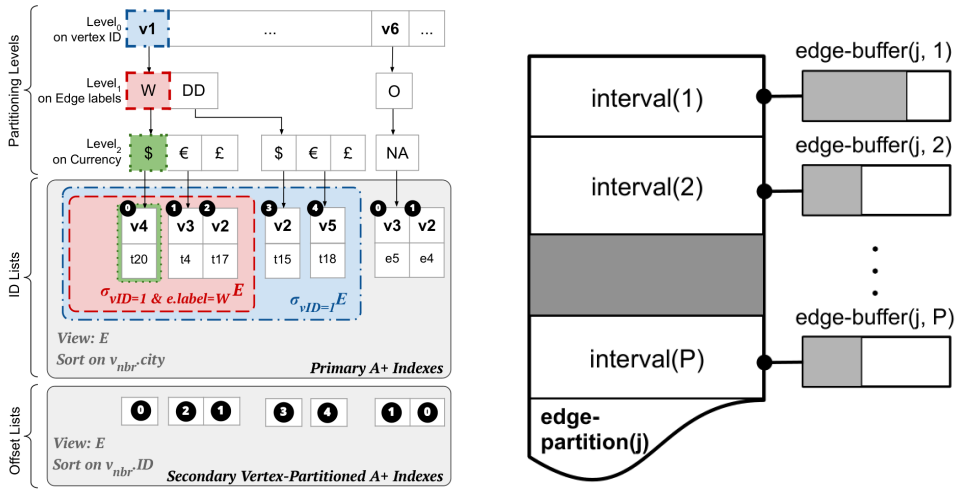
Graph database architectures are designed to efficiently manage and process complex, interconnected datasets by leveraging various structural and computational optimizations. One of the key innovations in this domain is the EMOGI framework, which introduces merged memory access and memory access alignment. These optimizations significantly enhance bandwidth utilization compared to existing Unified Virtual Memory (UVM)-based methods, enabling more efficient data handling and processing in graph databases [69]. The impact of such architectural advancements is profound, as they facilitate the management of increasingly large datasets, ensuring that performance remains robust even as data complexity grows.

Another notable architecture is Loom, which excels in query-aware partitioning, particularly beneficial in dynamic graph environments. Loom's ability to produce partitionings tailored to specific query workloads results in improved performance, addressing the challenges of dynamic data processing [70]. This approach is crucial for optimizing query execution and resource allocation in environments where data is continuously evolving. The flexibility of Loom in adapting to varying query patterns exemplifies the importance of architecture in enhancing the usability and efficiency of graph databases.

The architecture of graph databases also involves optimizing recursive queries, a common requirement in many applications. The proposed optimization strategy involves differentially maintaining recursive queries by selectively dropping differences and recomputing them as needed. This method reduces memory overhead while maintaining performance, offering a scalable solution for processing complex recursive queries [71]. Such optimization techniques are essential for applications that rely on complex relationships and require efficient data retrieval methods, ensuring that graph databases can meet the demands of modern applications effectively.

Furthermore, the taxonomy of streaming processing frameworks for dynamic graphs provides a comprehensive classification based on various aspects such as update ingestion, historical data maintenance, dynamic graph representation, incremental changes, and programming APIs. This taxonomy aids in understanding the diverse architectural approaches adopted by different frameworks to manage dynamic graph data efficiently [72]. The classification of these frameworks not only enhances our understanding of their operational mechanisms but also guides future developments in the field by identifying best practices and potential areas for improvement.

The architectural innovations in graph databases, such as those exemplified by GrapAL and DeltaGraph, highlight their remarkable adaptability and efficiency in managing complex data environments. These systems effectively support diverse use cases, from enabling researchers to explore scientific literature and uncover connections between biomedical entities to facilitating efficient retrieval of historical data in evolving networks. Additionally, advanced query operators for provenance graphs enhance the ability to analyze and summarize complex relationships within data, further demonstrating the capabilities of graph databases in addressing intricate data challenges [11, 68, 39]. By integrating advanced memory management techniques, query-aware partitioning, and optimized recursive query processing, graph database architectures continue to evolve, meeting the demands of modern data-intensive applications.



(a) Partitioning Levels and ID Lists for Efficient Indexing in Graph Data Structures[73]

(b) An Image of a Graph Representation[74]

Figure 4: Examples of Graph Database Architectures

As shown in Figure 4, understanding the intricacies of data organization and representation is crucial for efficient data retrieval and manipulation in the realm of graph database models and architectures. The provided examples offer insights into two distinct approaches to structuring graph data. The first example, "Partitioning Levels and ID Lists for Efficient Indexing in Graph Data Structures," showcases a multi-level partitioning strategy that enhances indexing efficiency. This model employs three levels of partitioning, with vertex IDs at the topmost level, followed by edge labels, demonstrating a hierarchical approach to data segmentation. Such a structure facilitates quicker access and management of graph data by organizing it into manageable, color-coded sections. The second example, "An Image of a Graph Representation," illustrates a more interconnected approach, where a central node labeled "interval(P)" serves as a hub, linking to various "edge-partition(j)" nodes. These nodes, in turn, connect to "edge-buffer(j, 1)" nodes, forming a complex network of relationships. This representation highlights the dynamic nature of graph databases, where nodes and edges are intricately linked, allowing for flexible data traversal and querying. Together, these examples underscore the diversity in graph database architectures, each tailored to optimize performance and scalability in handling complex data relationships [73, 74].

4 Data Modeling in Graph Databases

Graph databases are integral to managing complex datasets, providing frameworks that excel in depicting relationships and interactions among data points. They are particularly advantageous in fields such as social networks, biological data analysis, and recommendation systems. Table 3 offers a detailed summary of the methods and features utilized in graph databases, illustrating the diverse techniques and innovative approaches that enhance data modeling and analysis. By examining various techniques and methodologies within these databases, we gain insights into their effectiveness in modeling and analyzing intricate datasets, setting the stage for a deeper exploration of specific techniques.

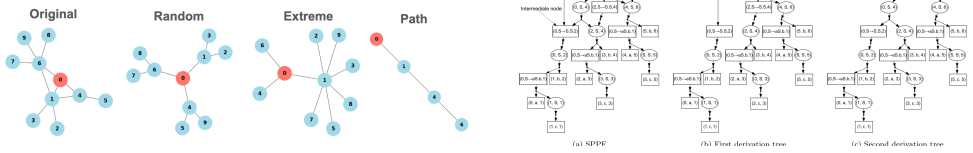
4.1 Techniques and Methodologies

Graph databases employ diverse methodologies to optimize data modeling and analysis. Vertex sampling, based on fractional vertex cover, enhances data processing efficiency without sacrificing insight quality [77]. TigerGraph's parallel architecture exemplifies the power of simultaneous query execution, reducing analysis time [4]. GRainDB integrates graph capabilities into RDBMSs using extended SQL syntax, easing adoption for SQL users [14]. Privacy concerns are addressed by the kt-safe graph generation method, ensuring k-anonymity and t-closeness [21]. Dynamic environments benefit from 2-hop labeling for rapid updates and real-time responses [6]. The Declarative Transfor-

Category	Feature	Method
Techniques and Methodologies	Complex Data Handling Graph-Based Techniques Database Efficiency Data Protection and Privacy	GLL[75], DTF[76] SCA[77], CSC[6], HONC[78] TG[4], GRainDB[14], N/A[18] kt-safe[21]
Representation of Entities, Relationships, and Attributes	Graph Structure Representation	PG-S[79]
Data Modeling Scenarios	Robustness and Generalization Feature Separation Hardware Acceleration	ARGA[80] STGD-VAE[65] FAST[56]
Best Practices and Challenges	Performance Optimization	CUT[81], FS+[82]
Innovative Data Modeling Approaches	Robustness Enhancement Complex Relationship Modeling Decentralized Graph Techniques Integrated Graph Structures	AT-GAE/VGAE[83] TGM[84] VFGNN[15] KG[13]

Table 1: This table provides a comprehensive overview of various methods employed in graph databases, categorized into distinct areas such as techniques and methodologies, representation of entities, data modeling scenarios, best practices, and innovative approaches. Each category encompasses specific features and methods, highlighting the diverse strategies used to enhance data handling, graph representation, and performance optimization. The table serves as a valuable resource for understanding the multifaceted nature of graph data modeling and its application in complex data environments.

mation Framework (DTF) streamlines complex data manipulations [76]. Techniques capturing distant neighbors' information are crucial for comprehensive analysis [78]. PhyloDB integrates algorithms directly into Neo4j for efficient phylogenetic analysis [18]. Advanced summarization techniques further enhance data comprehension and insight extraction [11, 38, 39].



(a) Network Analysis: Different Types of Graphs[85]

(b) Derivation Trees for a Grammar[75]

Figure 5: Examples of Techniques and Methodologies

Figure 5 illustrates the dual aspects of network analysis and grammatical derivation, showcasing the versatility of graph databases in modeling complex systems. Diverse graph types highlight network topology variations, while derivation trees emphasize language structure hierarchy [85, 75].

4.2 Representation of Entities, Relationships, and Attributes

Graph databases represent entities, relationships, and attributes through nodes, edges, and properties. Nodes encapsulate entities, as seen in Maven Dependency Graph where artifacts are nodes [2]. TGM uses hyper-nodes for complex structures while maintaining schema constraints [84]. Edges define connections, illustrating dependencies in Maven Dependency Graph and enabling complex relational modeling with hyper-edges in TGM [2, 84]. Properties add context, with PG-Schema ensuring coherent data representation [79]. Advanced methodologies like RWR-GAE and T2-GNN refine entity and relationship representation through embeddings and dual-model guidance [86, 87]. Adversarial perturbations enhance robustness in graph autoencoders [83]. These components demonstrate graph databases' effectiveness in modeling intricate datasets.

4.3 Data Modeling Scenarios

Graph data modeling is employed across diverse scenarios, each leveraging graph structures' capabilities. Table 2 provides a comprehensive comparison of graph data modeling methods, detailing their application areas, techniques, and data types, thereby illustrating the diverse scenarios where graph structures are effectively utilized. Scientific research benefits from modeling citation networks, as

AI-generated, for reference only.

Method Name	Application Domains	Modeling Techniques	Data Types
ARGA[80]	Scientific Research	Adversarial Training	Citation Networks
STGD-VAE[65]	Protein Folding	Bayesian Model	Traffic Data
FAST[56]	Subgraph Matching	Fpga-based Algorithms	Large Graphs

Table 2: This table presents a comparative overview of different graph data modeling methods, highlighting their application domains, modeling techniques, and data types. The methods ARGA, STGD-VAE, and FAST demonstrate the versatility and specificity of graph modeling in scientific research, protein folding, and subgraph matching, respectively. This comparison underscores the adaptability of these methods to various complex data scenarios.

seen in datasets like Cora and PubMed, enhancing graph neural networks' robustness [80]. Spatiotemporal analysis uses models like STGD-VAE for disentangling spatial and temporal features, crucial in traffic and biological data [65]. Large-scale graph processing, exemplified by LDBC benchmarks, showcases FPGA-based algorithms for efficient subgraph matching [56]. These scenarios illustrate graph data modeling's adaptability and effectiveness in summarizing and exploring complex datasets, facilitating insights across various domains [23, 88, 39, 11, 38].

4.4 Best Practices and Challenges

Best practices in graph data modeling optimize performance and scalability. Cuttana framework enhances partitioning quality, reducing network overhead in distributed processing [81]. Sparse learning methods offer faster convergence and feature learning, crucial for dynamic graphs [89, 77]. Challenges include clone detection and GGD validation [90, 91]. TGM provides enhanced semantic expressiveness but requires careful architectural tuning [84, 92]. Efficient recursive query maintenance and pattern search tools like GraphQ improve usability [71, 93]. Privacy concerns are addressed by FedSpectral+, reducing communication costs while maintaining clustering performance [82]. Benchmarks may not cover all use cases, affecting comprehensive evaluation [94, 37]. These practices and challenges highlight graph data modeling's dynamic nature, necessitating ongoing innovation.

4.5 Innovative Data Modeling Approaches

Innovative approaches in graph data modeling continue to evolve, addressing modern datasets' complexities. Graph neural networks (GNNs) capture relationships and dependencies, improving node classification and link prediction [19]. GNNs manage isolated graph data across distributed holders [15]. Causal semantics enhance graph models' expressiveness, aiding causal analysis [16]. Multimodal knowledge graphs integrate diverse data types for decision support [13]. Hypergraph models capture higher-order relationships, offering nuanced analyses [84]. Adversarial regularization in graph autoencoders enhances robustness [83]. These approaches underscore the need for continuous advancement in graph data modeling, facilitating insights from interconnected datasets. Tools like GrapAL exemplify practical applications, enhancing literature exploration and citation metrics analysis [38, 39].

Feature	Vertex Sampling	TigerGraph	GRainDB
Optimization Technique	Fractional Vertex Cover	Parallel Architecture	Extended Sql Syntax
Application Area	Data Processing	Query Execution	Rdbms Integration
Unique Feature	Efficiency Enhancement	Simultaneous Execution	Sql User Adoption

Table 3: This table provides a comparative analysis of three distinct graph database methodologies: Vertex Sampling, TigerGraph, and GRainDB. It highlights the optimization techniques, application areas, and unique features of each method, showcasing their contributions to enhancing data processing, query execution, and integration with relational database management systems (RDBMS). The table underscores the diverse approaches employed in graph databases to optimize data modeling and analysis.

5 Applications of Graph Databases

Graph databases transcend their foundational capabilities, demonstrating adaptability and effectiveness across diverse applications. A prominent area of impact is the analysis of social networks and community structures. This section explores how graph databases enhance understanding of social dynamics, allowing researchers to uncover complex relationships and community behaviors often hidden in traditional relational databases. Their ability to model intricate interconnections renders them invaluable across fields such as sociology, marketing, and organizational studies, especially as the digital landscape evolves and the analysis of social interactions becomes more critical.

5.1 Social Networks and Community Analysis

Graph databases are crucial in analyzing social networks and community structures, offering robust methodologies for managing complex relational data. Tools like Gremlin exemplify their flexibility, supporting both imperative and declarative querying, which is essential for diverse social network analysis applications [95]. This capability aids in exploring intricate community structures and relationships, providing insights into social dynamics. Representing entities as nodes and relationships as edges allows intuitive visualizations and analyses, revealing patterns and trends often obscured in traditional data structures. These insights are valuable in fields such as marketing, public health, and community engagement.

In community analysis, graph databases facilitate the detection of outlier communities, indicative of unique social phenomena. Community-based outlier detection techniques use edge-attributed graphs to analyze community structures, offering insights into underlying social interactions [96]. These methodologies are applicable in organizational graphs, where understanding community dynamics is vital for effective management. Identifying outlier communities helps organizations understand their workforce and improve team dynamics, enhancing organizational effectiveness.

Privacy-preserving methods like GraphSE2 extend graph databases' applicability in social network analysis by ensuring user privacy while maintaining analytical capabilities [97]. This is crucial in today's data-driven world, where privacy concerns are paramount. As organizations increasingly rely on social network data, the ability to analyze such data without compromising privacy is essential. Privacy-preserving techniques enhance user trust and broaden research possibilities in sensitive areas like healthcare and social services.

Graph Reinforcement Learning (GRL) techniques in graph databases improve task performance and efficiency, particularly in complex environments [98]. These advancements enhance analytical processes in social network analysis, providing accurate and efficient solutions for understanding social interactions. GRL optimizes models to better capture social dynamics, paving the way for sophisticated analyses and applications, highlighting the evolving relevance of graph databases.

Graph prompting methods, which enhance model performance in graph learning and natural language processing, are also relevant in social network analysis [99]. These methods enable nuanced analyses of social media interactions, contributing to a comprehensive understanding of social dynamics. Advanced prompting techniques extract richer contextual information from social media data, enhancing result interpretability. This is valuable in rapidly changing social landscapes, where timely insights inform policy decisions and community interventions.

Multimodal graph learning (MGL) techniques, integral to social media analysis, leverage diverse data types to offer a holistic view of social interactions, enabling effective community analysis [100]. Integrating multiple data modalities enhances analysis depth and breadth, providing richer insights into community structures. By analyzing various data streams, researchers develop a comprehensive understanding of social phenomena, leading to effective engagement strategies.

5.2 Recommendation Systems and Personalized Services

Graph databases enhance recommendation systems and personalized services by modeling complex relationships within data. The graph-based recommendation method by Delianidi et al. demonstrates efficiency in session-based recommendations, utilizing real-time data without extensive training [28]. This approach exemplifies graph databases' potential to provide timely, relevant recommendations, adapting to user preferences and behaviors. As demand for personalized experiences rises, graph

databases' ability to deliver contextually relevant suggestions becomes increasingly valuable in industries like e-commerce, streaming services, and social media.

Deep learning methods applied to graphs have advanced recommendation systems by capturing intricate patterns within user-item interactions [101]. The integration of Graph Attention LLM (GALLM) in recommendation systems emphasizes understanding relationships in graph data to improve personalization [102]. Graph structures enable accurate, tailored recommendations, aligning with individual user needs, enhancing satisfaction, engagement, and loyalty—critical for modern digital platforms.

Future research in graph-based recommendation systems includes advanced methodologies like GPN and HAT-GAE. GPN, discussed by Ding et al., promises improvements in structural learning and user engagement [103]. HAT-GAE exploration could further enhance personalized services, reflecting ongoing recommendation system evolution [104]. These innovative approaches highlight the need for continuous improvement to meet dynamic user demands.

Graph databases provide a robust framework for recommendation systems, integrating complex data relationships to enable personalized services that resonate with user preferences. Advanced models' ability to represent complex interactions and adapt to changing behaviors is crucial for effective recommendation systems, leveraging graph data for relationship mining and contextual understanding, leading to relevant, accurate recommendations. Advancements in graph database technologies and methodologies promise to enhance recommendation systems, driving innovation and improving user experiences across sectors [105, 20, 102].

5.3 Knowledge Graphs and Semantic Data Management

Graph databases are pivotal in managing semantic data and knowledge graphs, offering advanced capabilities for integrating and querying complex datasets. The extension of SPARQL to handle multimodal queries, as demonstrated by VGStore, significantly advances knowledge base question answering (KBQA) systems working with multimodal datasets [43]. This adaptability supports diverse applications, from semantic web technologies to enterprise data management, enabling intelligent systems to derive insights from varied data sources.

Domain-specific languages (DSLs) for causal graph analysis illustrate graph databases' potential in semantic data management. These DSLs facilitate causal inference using graph neural networks and transformer-based approaches, enhancing insights from complex data structures [16]. This integration highlights graph databases' role in supporting sophisticated semantic reasoning and inference tasks. As organizations seek actionable insights, graph databases become essential for decision-making and strategic planning.

Graph databases excel in visualizing and interacting with knowledge graphs, enabling intuitive data exploration and user-friendly interfaces. This capability is crucial for applications requiring seamless interaction with intricate datasets, enhancing user engagement and data management efficiency. Tools like GrapAL leverage graph databases for intuitive queries, revealing connections between biomedical entities, while comprehensive data from sources like Google Scholar can be analyzed to detect trends and predict emerging research domains [20, 39]. Visualizing complex data relationships empowers researchers and decision-makers to draw meaningful conclusions.

Ontology-based approaches in graph databases improve semantic data management, particularly in dialogue systems where accurate responses are critical. Structuring knowledge in a graph format enhances understanding of user queries, providing contextually appropriate answers, improving user experience and satisfaction. The flexibility and scalability of graph databases make them ideal for managing evolving semantic data, allowing adaptation to new information and insights. As demand for sophisticated data management grows, graph databases' capabilities will shape the future of semantic data management.

5.4 Scientific Research and Data Analysis

Graph databases are vital in scientific research and complex data analysis, offering robust frameworks for managing intricate datasets. Their application spans numerous scientific domains, benefiting from their strengths in handling large-scale and dynamic data. Graph representation learning advancements enhance applications in scientific research and data analysis [19]. By integrating diverse data types

and sources, graph databases empower researchers to conduct comprehensive analyses, leading to richer insights and discoveries.

In phylogenetic research, graph databases like phyloDB facilitate large-scale analyses by integrating complex data types and enabling real-time analysis, supporting comprehensive studies [18]. This capability streamlines and enhances research processes. Visualization capabilities, demonstrated in scholarly data analysis, reveal important patterns and insights, enhancing research across scientific domains [20]. Visualizing relationships and trends enhances interpretability, fostering collaboration and knowledge sharing.

The GRainDB system exemplifies graph databases' application in scientific research, particularly in complex data analysis scenarios like COVID-19 contact tracing, highlighting adaptability in addressing real-world challenges [14]. Integrating graph databases in knowledge graph frameworks, such as those in the wind industry, provides explainable decision support, offering actionable insights for maintenance strategies based on operational data [13]. These applications illustrate graph databases' transformative potential in enhancing research methodologies and fostering innovation.

Privacy preservation in scientific data analysis is another area where graph databases excel. The kt-safe graph generation approach addresses privacy challenges by ensuring structural and attribute information protection, facilitating secure data analysis [21]. This is crucial in fields like healthcare and social sciences, where sensitive data must be handled carefully. Robust privacy-preserving mechanisms enable analyses without compromising privacy, fostering trust and collaboration in data sharing initiatives.

The method for visiting distant neighbors in graph structures demonstrates superior performance compared to traditional Graph Convolutional Networks (GCNs), especially in limited labeled data scenarios, illustrating graph databases' potential in enhancing scientific research methodologies [78]. These advancements emphasize graph databases' critical role in advancing scientific research, offering robust methodologies for managing and analyzing complex datasets across domains. Their ability to integrate diverse data types and support sophisticated analytical processes makes them indispensable in scientific discovery and innovation.

5.5 Industrial Applications and Real-Time Data Processing

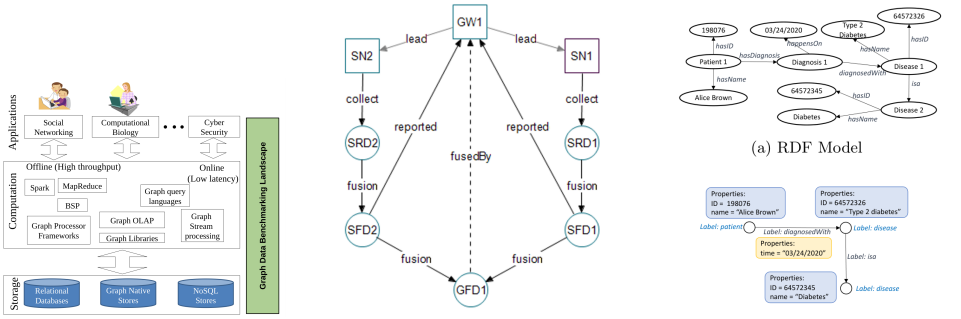
Graph databases are increasingly employed in industrial applications and real-time data processing due to their ability to efficiently manage complex, interconnected datasets. A primary advantage of graph databases, like System G, is handling high insertion rates while supporting numerous concurrent queries, making them suitable for real-time applications [40]. This efficiency is crucial in industries where rapid data ingestion and processing maintain operational effectiveness and competitiveness. Real-time analytics enable informed decision-making, enhancing responsiveness to market changes and operational challenges.

In finance, graph databases manage and analyze intricate networks of transactions and relationships, providing insights essential for risk management, fraud detection, and compliance. The LDBC FinBench serves as a standardized framework for testing and comparing graph database technologies in financial scenarios, ensuring chosen systems handle specific data workload demands [106]. This benchmarking is vital for financial institutions adopting graph databases as part of data management strategies, enhancing analytical capabilities for effective risk assessments and fraud prevention.

Graph databases play a crucial role in real-time data processing across industrial applications, particularly in supply chain management. They facilitate seamless tracking and analysis of goods and materials through intricate networks, enabling swift responses to demand changes and operational optimization. Advanced graph technologies provide deeper insights into supply chain dynamics, enhancing efficiency and decision-making [107, 38, 10]. Modeling and querying networks in real-time optimize logistics, reduce costs, and improve service delivery. In energy, graph databases support diverse data stream integration from sensors and IoT devices, facilitating real-time monitoring and decision-making for operational efficiency and sustainability.

Graph databases' application in industrial and real-time settings underscores their effectiveness in managing dynamic, interconnected data environments. Their ability to deliver timely insights and address complex queries makes them essential tools for industries seeking strategic data advantages. Advanced visualization techniques and NoSQL databases enable organizations to extract and represent

intricate data related to authors, affiliations, citations, and publication trends graphically. This facilitates knowledge discovery, interpretation, and identification of domain shifts, emerging subfields, and citation cartel behaviors at journal and author levels, supporting informed decision-making and professional growth across academic and technological landscapes [20, 47].



(a) Graph Data Benchmarking Landscape[108]

(b) A Network Diagram[22]

(a) RDF Model

(c) RDF Model for a Patient and a Disease[10]

Figure 6: Examples of Industrial Applications and Real-Time Data Processing

As shown in Figure 6, the utilization of graph databases in industrial applications and real-time data processing is exemplified through visual representations highlighting this technology's diverse and impactful nature. The "Graph Data Benchmarking Landscape" image provides a comprehensive overview of components and technologies integral to graph data processing, categorizing applications into social networking, computational biology, and cyber security, while distinguishing between offline and online computation modes. This landscape underscores graph databases' versatility and breadth in handling complex data relationships across domains. The "Network Diagram" illustrates data flow within an industrial setting, showcasing information routing between a central node, "Global Warehouse 1," and connected nodes, exemplifying graph databases' efficient data management capabilities. The "RDF Model for a Patient and a Disease" offers a glimpse into healthcare, where graph databases facilitate representing and managing intricate relationships between entities like patients and diseases, as demonstrated by Alice Brown's Type 2 Diabetes diagnosis. Collectively, these examples convey graph databases' significant role in enhancing data processing and management across industrial sectors [108, 22, 10].

6 Challenges and Future Directions

The evolution of graph databases is accompanied by various challenges, particularly in scalability and performance. As data complexity rises, recognizing the limitations of current methodologies is crucial. The following subsection examines scalability and performance issues, emphasizing inefficiencies in large dataset management and innovative solutions to these challenges. This analysis lays the groundwork for understanding the broader implications within the field. Additionally, technological advancements and the growing demand for efficient data management systems necessitate a thorough review of existing literature to pinpoint research gaps and opportunities.

6.1 Scalability and Performance

Scalability and performance are pivotal challenges in graph databases, especially with the increasing size and complexity of datasets. Inefficiencies in managing large graphs lead to computational bottlenecks, as seen with traditional methods that struggle with scalability [17]. Systems like phyloDB have been developed to efficiently handle large-scale datasets [18]. Maintaining high-quality embeddings during updates in dynamic knowledge graphs further complicates scalability [16]. Advanced strategies are needed to ensure consistency and scalability across evolving datasets.

Graph representation methods face scalability issues due to the need for adaptive learning techniques [19]. The computational complexity of ensuring privacy through kt-safe graph anonymization presents significant hurdles [21]. Moreover, web scraping can limit dataset comprehensiveness [47]. Alternative data collection methodologies are necessary to enhance data quality and breadth.

AI-generated, for reference only.

Implementing systems like VGStore in Python can introduce latency and performance issues, especially with extensive datasets and complex queries [43]. VFGNN addresses scalability by allowing initial node embeddings without sharing sensitive information [15]. Traditional RDBMSs have limitations in scalability, prompting optimized query processing techniques like GRainDB [14]. Addressing these challenges requires bridging academic research with practical applications to ensure graph databases support large-scale, complex data environments.

6.2 Query Optimization and Efficiency

Query optimization and efficiency are crucial in graph databases, directly affecting performance and scalability. The complexity of graph data demands advanced methodologies for efficient data retrieval. Enhancements in query optimization, fault tolerance, and query language capabilities are necessary [109]. The buffered streaming model by Cuttana improves query performance by optimizing vertex assignments in distributed environments [81]. This model enhances resource utilization and reduces costs.

Challenges in query optimization are evident in systems like DataChat, where effective query generation remains a hurdle [110]. Deadlocks during imports necessitate retries, slowing processes [111]. The inefficiencies of functions like shortestPath highlight the need for improved retrieval algorithms [48]. Sparse learning methods struggle with large graphs or complex interactions [89]. Continuous innovation in query optimization is essential for enhancing efficiency and scalability.

Improving query optimization techniques and algorithms can significantly boost performance and reliability in graph databases. This is vital for managing complex data, especially in life sciences and bioinformatics, where large-scale knowledge graphs are common. Innovative optimization approaches can yield significant speedups, ensuring graph databases remain integral to data management and analysis [49, 48, 10].

6.3 Security and Privacy

Security and privacy are critical in graph databases, especially with sensitive data in anomaly detection and molecular property prediction. Ensuring confidentiality and integrity in financial data applications is essential [112]. In cloud environments, protecting graph data during Graph Neural Networks operations is crucial [113]. The VFGNN framework uses differential privacy to secure information between data holders and servers [15]. Balancing security and functionality is vital for user trust and regulatory compliance.

Challenges remain in optimizing security without compromising performance. The complexity of methods like A+ indexes impacts scalability and efficiency, particularly in dense graphs [114]. Effective security frameworks must adapt to evolving threats, as traditional approaches may not suffice. The effectiveness of graph data imputation methods is influenced by security, with lower recall indicating vulnerabilities [115]. The scalability of security measures is constrained by computational demands for accurate query admissions [116]. Future research should explore RRDPQs optimization for acyclic graphs and nested regular expressions to enhance security [117].

Advancements in privacy-preserving techniques and security enhancements are crucial for protecting sensitive data. Methods like GraphSE2 and CryptGraph ensure encrypted analytics, maintaining data confidentiality. Innovations such as kt-safety and adversarial privacy-preserving graph embedding address vulnerabilities, ensuring integrity and performance in various applications.

6.4 Integration with Emerging Technologies

Integrating graph databases with emerging technologies enhances capabilities and broadens applicability. Hybrid approaches combining filtering with machine learning improve performance in dynamic graphs [17]. Integrating graph databases with Transformer models enhances computational efficiency and scalability [19]. These integrations revolutionize data processing, paving the way for sophisticated applications.

GRainDB exemplifies potential integrations, particularly in handling semi-structured data. Future work includes enhancing GRainDB for complex structures and compatibility with other databases [14]. Schema-driven selectivity estimation and recursive queries improve performance and user experience by providing efficient data retrieval [33].

In privacy and security, future research should optimize algorithms for larger datasets and explore additional privacy mechanisms with the kt-safety model [21]. These advancements ensure data security and privacy in cloud environments, leading to a secure and efficient data management ecosystem.

Integrating graph databases with streaming applications and machine learning models expands capabilities in real-time processing [18]. Knowledge graphs combined with Explainable AI models improve decision-making transparency [13]. This technological convergence enhances data management systems, allowing nuanced insights and improved analytical capabilities.

Integrating graph databases with technologies like NLP and data summarization enhances functionality and efficiency across fields such as scientific literature and biomedical research. Tools like GrapAL facilitate knowledge discovery, while summarization methods extract insights from complex data [38, 39]. Leveraging these advancements, graph databases become versatile tools in modern data-intensive environments.

7 Conclusion

Graph databases have established themselves as indispensable tools in modern data management, offering unparalleled capabilities for handling complex and interconnected datasets. Their architecture, which revolves around nodes, edges, and properties, provides a dynamic framework that surpasses traditional databases, particularly in scenarios demanding nuanced relationship management. The integration of graph databases with advanced technologies, such as large language models and automated query optimization systems, marks a significant leap in graph analytics, though there remain challenges in achieving widespread applicability and efficiency across diverse contexts.

The advancement of expressive query languages highlights the potential for developing robust frameworks capable of maintaining decidable model checking, thereby enhancing the querying functionalities of graph databases. Innovative frameworks demonstrate the ability of graph databases to improve data integration and interoperability, especially in specialized domains. By utilizing these systems, researchers can gain deeper insights into complex data relationships, facilitating more informed decision-making. Furthermore, the refinement of property graph models and the development of serialization formats enhance interoperability among graph databases, promoting seamless data exchanges and collaboration.

The transformation of RDF databases into property graph databases, while preserving critical information, showcases the adaptability and versatility of graph database technologies. Establishing formal semantics for query languages is crucial for ensuring consistent behavior across implementations and optimizing graph databases. Practical applications, such as bug detection and privacy-preserving functionalities, illustrate the real-world utility of graph databases. Additionally, methods for query rewriting underscore the relevance of graph databases in overcoming traditional limitations and addressing complex data challenges across various fields. This ongoing evolution underscores the vital role of graph databases in contemporary data management, paving the way for future innovations.

References

[1] Gábor Szárnyas, Brad Bebee, Altan Birler, Alin Deutsch, George Fletcher, Henry A. Gabb, Denise Gosnell, Alastair Green, Zhihui Guo, Keith W. Hare, Jan Hidders, Alexandru Iosup, Atanas Kiryakov, Tomas Kovatchev, Xinsheng Li, Leonid Libkin, Heng Lin, Xiaojian Luo, Arnau Prat-Pérez, David Püroja, Shipeng Qi, Oskar van Rest, Benjamin A. Steer, Dávid Szakállas, Bing Tong, Jack Waudby, Mingxi Wu, Bin Yang, Wenyuan Yu, Chen Zhang, Jason Zhang, Yan Zhou, and Peter Boncz. The linked data benchmark council (ldbc): Driving competition and collaboration in the graph data management space, 2024.

[2] Amine Benelallam, Nicolas Harrand, César Soto Valero, Benoit Baudry, and Olivier Barais. The maven dependency graph: a temporal graph-based representation of maven central, 2019.

[3] Rui Song, Fausto Giunchiglia, Ke Zhao, and Hao Xu. Topological regularization for graph neural networks augmentation, 2021.

[4] Alin Deutsch, Yu Xu, Mingxi Wu, and Victor Lee. Tigergraph: A native mpp graph database, 2019.

[5] Nico Hochgeschwender, Holger Voos, and Gerhard K. Kraetzschmar. Towards persistent storage and retrieval of domain models using graph database technology, 2016.

[6] Qingshuai Feng, You Peng, Wenjie Zhang, Ying Zhang, and Xuemin Lin. Towards real-time counting shortest cycles on dynamic graphs: A hub labeling approach, 2022.

[7] Hanna Abi Akl. The path to autonomous learners, 2022.

[8] Bianca Löhnert, Nikolaus Augsten, Cem Okulmus, and Magdalena Ortiz. Towards practicable algorithms for rewriting graph queries beyond dl-lite, 2024.

[9] Chantal Montgomery, Haruna Isah, and Farhana Zulkernine. Towards a natural language query processing system, 2020.

[10] Yuanyuan Tian. The world of graph databases from an industry perspective, 2022.

[11] Hui Miao and Amol Deshpande. Understanding data science lifecycle provenance via graph segmentation and summarization, 2018.

[12] Dongqi Fu, Liri Fang, Zihao Li, Hanghang Tong, Vetle I. Torvik, and Jingrui He. What do llms need to understand graphs: A survey of parametric representation of graphs, 2025.

[13] Joyjit Chatterjee and Nina Dethlefs. Xai4wind: A multimodal knowledge graph database for explainable decision support in operations maintenance of wind turbines, 2021.

[14] Guodong Jin, Nafisa Anzum, and Semih Salihoglu. Graindb: A relational-core graph-relational dbms. In *CIDR*, volume 2, page 26, 2022.

[15] Chaochao Chen, Jun Zhou, Longfei Zheng, Huiwen Wu, Lingjuan Lyu, Jia Wu, Bingzhe Wu, Ziqi Liu, Li Wang, and Xiaolin Zheng. Vertically federated graph neural network for privacy-preserving node classification, 2022.

[16] Amedeo Pachera, Mattia Palmiotto, Angela Bonifati, and Andrea Mauri. What if: Causal analysis with graph databases, 2024.

[17] C. Q. Cheng, K. S. Wong, and L. K. Soon. l2match: Optimization techniques on subgraph matching algorithm using label pair, neighboring label index, and jump-redo method, 2023.

[18] Bruno Lourenço, Cátia Vaz, Miguel E. Coimbra, and Alexandre P. Francisco. phylodb: A framework for large-scale phylogenetic analysis, 2023.

[19] William L Hamilton. *Graph representation learning*. Morgan & Claypool Publishers, 2020.

[20] Gouri Ginde. Visualisation of massive data from scholarly article and journal database a novel scheme, 2016.

[21] Weilong Ren, Kambiz Ghazinour, and Xiang Lian. kt-safety: Graph release via k-anonymity and t-closeness (technical report), 2022.

[22] Cihan Küçükkeçeci and Adnan Yazıcı. Big data model simulation on a graph database for surveillance in wireless multimedia sensor networks, 2017.

[23] Wenbo Shang and Xin Huang. A survey of large language models on generative graph analytics: Query, learning, and applications, 2024.

[24] Zhengyu Hu, Yichuan Li, Zhengyu Chen, Jingang Wang, Han Liu, Kyumin Lee, and Kaize Ding. Let's ask gnn: Empowering large language model for graph in-context learning, 2025.

[25] Cristiano G. Sabiu, Ben Hoyle, Juhan Kim, and Xiao-Dong Li. Graph database solution for higher order spatial statistics in the era of big data, 2019.

[26] Partha Sen and Sumana Sen. Graph database while computationally efficient filters out quickly the esg integrated equities in investment management, 2024.

[27] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Martin Schuster, Petra Selmer, and Andrés Taylor. Formal semantics of the language cypher, 2018.

[28] Marina Delianidi, Michail Salampasis, Konstantinos Diamantaras, Theodosios Siomos, Alkiviadis Katsalis, and Iphigenia Karaveli. A graph-based method for session-based recommendations, 2021.

[29] Jelena Grbić, Jie Wu, Kelin Xia, and Guo-Wei Wei. A unified topological approach to data science, 2021.

[30] Xiangyan Liu, Bo Lan, Zhiyuan Hu, Yang Liu, Zhicheng Zhang, Fei Wang, Michael Shieh, and Wenmeng Zhou. Codexgraph: Bridging large language models and code repositories via code graph databases, 2024.

[31] Florin Rusu and Zhiyi Huang. In-depth benchmarking of graph database systems with the linked data benchmark council (ldbc) social network benchmark (snb), 2019.

[32] Pranav Addepalli, Eric Wu, Douglas Bossart, Christina Lin, and Allistar Smith. Spider: Selective plotting of interconnected data and entity relations, 2020.

[33] Guillaume Bagan, Angela Bonifati, Radu Ciucanu, George H. L. Fletcher, Aurélien Lemay, and Nicky Advokaat. gmark: Schema-driven generation of graphs and queries, 2016.

[34] Christopher Wewer, Florian Lemmerich, and Michael Cochez. Updating embeddings for dynamic knowledge graphs, 2021.

[35] Harsh Thakkar, Dharmen Punjani, Soeren Auer, and Maria-Esther Vidal. Towards an integrated graph algebra for graph pattern matching with gremlin (extended version), 2019.

[36] Erxue Min, Runfa Chen, Yatao Bian, Tingyang Xu, Kangfei Zhao, Wenbing Huang, Peilin Zhao, Junzhou Huang, Sophia Ananiadou, and Yu Rong. Transformer for graphs: An overview from architecture perspective, 2022.

[37] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

[38] Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. Graph summarization methods and applications: A survey, 2018.

[39] Christine Betts, Joanna Power, and Waleed Ammar. Grapal: Connecting the dots in scientific literature, 2019.

[40] Gabriel Tanase, Toyotaro Suzumura, Jinho Lee, Chun-Fu Chen, Jason Crawford, Hiroki Kanezashi, Song Zhang, and Warut D. Vijitbenjaronk. System g distributed graph database, 2018.

[41] Hirokazu Chiba, Ryota Yamanaka, and Shota Matsumoto. Property graph exchange format, 2019.

[42] Till Blume, Jannik Rau, David Richerby, and Ansgar Scherp. Time and memory efficient parallel algorithm for structural graph summaries and two extensions to incremental summarization and k-bisimulation for long k-chaining, 2022.

[43] Yanzeng Li, Zilong Zheng, Wenjuan Han, and Lei Zou. Vgstore: A multimodal extension to sparql for querying rdf scene graph, 2022.

[44] Zihao Zhao, Zhihong Shen, Mingjie Tang, Chuan Hu, Huajin Wang, and Yuanchun Zhou. Pandadb: Understanding unstructured data in graph database, 2022.

[45] Charalampos E. Tsourakakis. Streaming graph partitioning in the planted partition model, 2014.

[46] Makbule Gulcin Ozsoy, Leila Messallem, Jon Besga, and Gianandrea Minneci. Text2cypher: Bridging natural language and graph databases, 2024.

[47] Gouri Ginde, Snehanshu Saha, Archana Mathur, Harsha Vamsi, Sudeepa Roy Dey, and Swati Sampatrao Gambhire. Use of nosql database and visualization techniques to analyze massive scholarly article data from journals, 2018.

[48] Jens Dörpinghaus and Andreas Stefan. Optimization of retrieval algorithms on large scale knowledge graphs, 2020.

[49] Renzo Angles, Marcelo Arenas, Pablo Barceló, Aidan Hogan, Juan Reutter, and Domagoj Vrgoč. Foundations of modern query languages for graph databases. *ACM Computing Surveys (CSUR)*, 50(5):1–40, 2017.

[50] Malcolm Crowe and Fritz Laux. Graph data models and relational database technology, 2023.

[51] Zihao Zhao, Xiaodong Ge, and Zhihong Shen. S2ctrans: Building a bridge from sparql to cypher, 2023.

[52] Renzo Angles, Angela Bonifati, Roberto García, and Domagoj Vrgoč. Path-based algebraic foundations of graph query languages, 2024.

[53] Renzo Angles, Harsh Thakkar, and Dominik Tomaszuk. Directly mapping rdf databases to property graph databases, 2020.

[54] Shota Matsumoto, Ryota Yamanaka, and Hirokazu Chiba. Mapping rdf graphs to property graphs, 2018.

[55] Diego Figueira, Anthony W. Lin, and Liat Peterfreund. Relational perspective on graph query languages, 2024.

[56] Xin Jin, Zhengyi Yang, Xuemin Lin, Shiyu Yang, Lu Qin, and You Peng. Fast: Fpga-based subgraph matching on massive graphs, 2021.

[57] Congquan Mei, Lian Chen, Junfeng Zhou, Ming Du, Sheng Yu, Xian Tang, and Ziyang Chen. Efficient k-step weighted reachability query processing algorithms, 2024.

[58] Jielin Chen and Rudi Stouffs. Se-vgae: Unsupervised disentangled representation learning for interpretable architectural layout design graph generation, 2024.

[59] Morteza Alipourlangouri, Adam Mansfield, Fei Chiang, and Yinghui Wu. Temporal graph functional dependencies [extended version], 2022.

[60] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[61] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications, 2021.

[62] Xuhong Wang, Ding Lyu, Mengjian Li, Yang Xia, Qi Yang, Xinwen Wang, Xinguang Wang, Ping Cui, Yupu Yang, Bowen Sun, and Zhenyu Guo. Apan: Asynchronous propagation attention network for real-time temporal graph embedding, 2021.

[63] Jianjun Wei, Yue Liu, Xin Huang, Xin Zhang, Wenyi Liu, and Xu Yan. Self-supervised graph neural networks for enhanced feature extraction in heterogeneous information networks, 2024.

[64] Falih Gozi Febrinanto, Feng Xia, Kristen Moore, Chandra Thapa, and Charu Aggarwal. Graph lifelong learning: A survey, 2022.

[65] Yuanqi Du, Xiaojie Guo, Hengning Cao, Yanfang Ye, and Liang Zhao. Disentangled spatiotemporal graph generative models, 2022.

[66] Ikechukwu Maduako, Emerson Cavalheri, and Monica Wachowicz. Exploring the use of time-varying graphs for modelling transit networks, 2018.

[67] Shohei Nakazawa, Yoshiki Sato, Kenji Nakagawa, Sho Tsugawa, and Kohei Watabe. A tunable model for graph generation using lstm and conditional vae, 2021.

[68] Udayan Khurana and Amol Deshpande. Efficient snapshot retrieval over historical graph data, 2012.

[69] Seung Won Min, Vikram Sharma Mailthody, Zaid Qureshi, Jinjun Xiong, Eiman Ebrahimi, and Wen mei Hwu. Emogi: Efficient memory-access for out-of-memory graph-traversal in gpus, 2021.

[70] Hugo Firth, Paolo Missier, and Jack Aiston. Loom: Query-aware partitioning of online graphs, 2017.

[71] Khaled Ammar, Siddhartha Sahu, Semih Salihoglu, and M. Tamer Ozsu. Optimizing differentially-maintained recursive queries on dynamic graphs, 2022.

[72] Maciej Besta, Marc Fischer, Vasiliki Kalavri, Michael Kapralov, and Torsten Hoefler. Practice of streaming processing of dynamic graphs: Concepts, models, and systems, 2021.

[73] Amine Mhedhbi, Pranjal Gupta, Shahid Khaliq, and Semih Salihoglu. A+ indexes: Tunable and space-efficient adjacency lists in graph database management systems, 2021.

[74] Aapo Kyrola and Carlos Guestrin. Graphchi-db: Simple design for a scalable graph database system – on just a pc, 2014.

[75] Semyon Grigorev and Anastasiya Ragozina. Context-free path querying with structural representation of result, 2017.

[76] Angela Bonifati, Filip Murlak, and Yann Ramusat. Transforming property graphs, 2024.

[77] John Kallaugher, Michael Kapralov, and Eric Price. The sketching complexity of graph and hypergraph counting, 2018.

[78] Alireza Hashemi and Hernan Makse. Visiting distant neighbors in graph convolutional networks, 2024.

[79] Renzo Angles, Angela Bonifati, Stefania Dumbrava, George Fletcher, Alastair Green, Jan Hidders, Bei Li, Leonid Libkin, Victor Marsault, Wim Martens, Filip Murlak, Stefan Plantikow, Ognjen Savković, Michael Schmidt, Juan Sequeda, Sławek Staworko, Dominik Tomaszuk, Hannes Voigt, Domagoj Vrgoč, Mingxi Wu, and Dušan Živković. Pg-schema: Schemas for property graphs, 2023.

[80] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding, 2019.

[81] Milad Rezaei Hajidehi, Sraavan Sridhar, and Margo Seltzer. Cuttana: Scalable graph partitioning for faster distributed graph databases and analytics, 2024.

[82] Janvi Thakkar and Devvrat Joshi. Fedspectral+: Spectral clustering using federated learning, 2023.

[83] Tianjin Huang, Yulong Pei, Vlado Menkovski, and Mykola Pechenizkiy. On generalization of graph autoencoders with adversarial training, 2021.

[84] Fritz Laux. The typed graph model – a supermodel for model management and data integration, 2021.

[85] Jin Huang, Xingjian Zhang, Qiaozhu Mei, and Jiaqi Ma. Can llms effectively leverage graph structural information through prompts, and why?, 2024.

[86] Vaibhav, Po-Yao Huang, and Robert Frederking. Rwr-gae: Random walk regularization for graph auto encoders, 2019.

[87] Cuiying Huo, Di Jin, Yawen Li, Dongxiao He, Yu-Bin Yang, and Lingfei Wu. T2-gnn: Graph neural networks for graphs with incomplete features and structure via teacher-student distillation, 2022.

[88] Siddhartha Sahu, Amine Mhedhbi, Semih Salihoglu, Jimmy Lin, and M. Tamer Özsu. The ubiquity of large graphs and surprising challenges of graph processing: Extended survey, 2019.

[89] Ichigaku Takigawa and Hiroshi Mamitsuka. Sparse learning over infinite subgraph features, 2014.

[90] Stephan Doerfel, Tom Hanika, and Gerd Stumme. Clones in graphs, 2018.

[91] Larissa C. Shimomura, George Fletcher, and Nikolay Yakovets. Ggds: Graph generating dependencies, 2020.

[92] Su Hyeong Lee, Qingqi Zhang, and Risi Kondor. Sign rank limitations for inner product graph decoders, 2024.

[93] Huan Song, Zeng Dai, Panpan Xu, and Liu Ren. Interactive visual pattern search on graph data via graph representation learning, 2022.

[94] Pieter Cailliau, Tim Davis, Vijay Gadepally, Jeremy Kepner, Roi Lipman, Jeffrey Lovitz, and Keren Ouaknine. Redisgraph graphblas enabled graph database, 2019.

[95] Marko A. Rodriguez. The gremlin graph traversal machine and language, 2015.

[96] Supriya Pandhre, Manish Gupta, and Vineeth N Balasubramanian. Community-based outlier detection for edge-attributed graphs, 2017.

[97] Shangqi Lai, Xingliang Yuan, Shi-Feng Sun, Joseph K. Liu, Yuhong Liu, and Dongxi Liu. Graphse^2: An encrypted graph database for privacy-preserving social search, 2019.

[98] Mingshuo Nie, Dongming Chen, and Dongqi Wang. Reinforcement learning on graphs: A survey, 2023.

[99] Xuansheng Wu, Kaixiong Zhou, Mingchen Sun, Xin Wang, and Ninghao Liu. A survey of graph prompting methods: Techniques, applications, and challenges, 2023.

[100] Ciyuan Peng, Jiayuan He, and Feng Xia. Learning on multimodal graphs: A survey, 2024.

[101] Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey, 2020.

[102] Xinyuan Wang, Liang Wu, Liangjie Hong, Hao Liu, and Yanjie Fu. Llm-enhanced user-item interactions: Leveraging edge information for optimized recommendations, 2024.

[103] Qianggang Ding, Deheng Ye, Tingyang Xu, and Peilin Zhao. Gpn: A joint structural learning framework for graph neural networks, 2022.

[104] Chengyu Sun. Hat-gae: Self-supervised graph auto-encoders with hierarchical adaptive masking and trainable corruption, 2023.

[105] Radeen Mostafa, Mirza Nihal Baig, Mashaekh Tausif Ehsan, and Jakir Hasan. G-rag: Knowledge expansion in material science, 2024.

[106] Shipeng Qi, Heng Lin, Zhihui Guo, Gábor Szárnyas, Bing Tong, Yan Zhou, Bin Yang, Jiansong Zhang, Zheng Wang, Youren Shen, Changyuan Wang, Parviz Peiravi, Henry Gabb, and Ben Steer. The ldbc financial benchmark, 2023.

[107] Leticia Gómez, Bart Kuijpers, and Alejandro Vaisman. Online analytical processsing on graph data, 2019.

[108] Miyuru Dayarathna and Toyotaro Suzumura. Benchmarking graph data management and processing systems: A survey, 2021.

[109] Chiranjeeb Buragohain, Knut Magne Risvik, Paul Brett, Miguel Castro, Wonhee Cho, Joshua Cowhig, Nikolas Gloy, Karthik Kalyanaraman, Richendra Khanna, John Pao, Matthew Renzelmann, Alex Shamis, Timothy Tan, and Shuheng Zheng. A1: A distributed in-memory graph database, 2020.

[110] Lizhou Fan, Sara Lafia, Lingyao Li, Fangyuan Yang, and Libby Hemphill. Datachat: Prototyping a conversational agent for dataset search and visualization, 2023.

[111] Joshua Porter and Aleks Ontman. Importing relationships into a running graph database using parallel processing, 2020.

[112] Shalin Shah, Srikanth Ryali, and Ramasubbu Venkatesh. Multi-document financial question answering using llms, 2024.

[113] Songlei Wang, Yifeng Zheng, and Xiaohua Jia. Secgnn: Privacy-preserving graph neural network training and inference as a cloud service, 2023.

[114] M. Praveen and B. Srivathsan. Nesting depth of operators in graph database queries: Expressiveness vs. evaluation complexity, 2016.

[115] Jiang Hua, Michael Bewong, Selasi Kwashie, MD Geaur Rahman, Junwei Hu, Xi Guo, and Zaiwen Fen. Gig: Graph data imputation with graph differential dependencies, 2024.

[116] Hao Xu and Juan A. Colmenares. Admission control with response time objectives for low-latency online data systems, 2023.

[117] Zhilin Wu. Regular path queries on graphs with data: A rigid approach, 2014.

Disclaimer:

SurveyX is an AI-powered system designed to automate the generation of surveys. While it aims to produce high-quality, coherent, and comprehensive surveys with accurate citations, the final output is derived from the AI's synthesis of pre-processed materials, which may contain limitations or inaccuracies. As such, the generated content should not be used for academic publication or formal submissions and must be independently reviewed and verified. The developers of SurveyX do not assume responsibility for any errors or consequences arising from the use of the generated surveys.

AI-generated, for reference only.