MANTECH Publications

Journal of Data Mining and Knowledge Engineering
Volume 5, Issue 1, January-June 2020

Scalable Construction of Dynamic Knowledge Graphs

Sanjeet S Kulkarni
Associate Professor
Department of Computational Knowledge Engineering
Mount Carmel College, Bengaluru, India
Email: *Sanjeets41kulkarni@rediffmail.com*

Abstract

Knowledge graphs (KGs) have become fundamental in representing and organizing complex relationships among entities across diverse domains. As the volume and velocity of data increase, the need for dynamic and scalable knowledge graph construction becomes critical. Traditional static KGs fail to capture the evolving nature of real-world knowledge, limiting their applicability in domains such as recommendation systems, biomedical research, and social networks. This paper presents a comprehensive review of scalable methods for constructing dynamic knowledge graphs, highlighting both architecture-level and algorithmic solutions. We explore data ingestion strategies, incremental updating mechanisms, distributed processing frameworks, and emerging machine learning techniques for entity and relation extraction. Moreover, we analyze current challenges including graph sparsity, temporal reasoning, and real-time updates, and discuss evaluation metrics suitable for dynamic KGs. Finally, this work provides insights into future directions for building robust, scalable, and adaptive knowledge graph systems.

Keywords: *Knowledge Graphs, Dynamic Graphs, Scalability, Graph Construction, Entity Extraction, Temporal Reasoning, Distributed Computing.*

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

INTRODUCTION

Knowledge graphs (KGs) are structured representations of real-world entities and their interrelations, widely applied in search engines, recommendation systems, question answering, and domain-specific analytics. The traditional approach focuses on building static KGs derived from curated or batch-processed datasets. However, the rapid growth of streaming data and evolving domain knowledge necessitates **dynamic knowledge graphs** that adapt continuously.

Dynamic KGs require mechanisms for:

- Incremental integration of new data
- Efficient update of existing entities and relationships
- Scalability to handle large-scale heterogeneous datasets

This paper reviews the current state-of-the-art methods for **scalable dynamic knowledge graph construction**, focusing on system architectures, data processing pipelines, and advanced machine learning techniques.

FUNDAMENTALS OF KNOWLEDGE GRAPHS

Knowledge Graphs (KGs) are structured representations of knowledge where entities and the relationships between them are explicitly modeled. They allow machines and humans to understand complex, interrelated information in a graph-based format. KGs have become essential in applications like semantic search, recommendation systems, question answering, and natural language understanding.

A fundamental understanding of KGs involves their **structure, representation, and semantics**.

1. Structure of Knowledge Graphs

A typical knowledge graph is formally defined as:

$G=(V,E)G = (V, E)G=(V,E)$

Where:

- VVV is the set of **entities** (or nodes), representing real-world objects, concepts, or events.
- EEE is the set of **relationships** (or edges), representing directed links between entities.

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

Each relationship is typically represented as a **triple**:

(h,r,t)(h, r, t)(h,r,t)

where:

- hhh (**head entity**) is the source or subject of the relationship.
- rrr (**relation type**) is the predicate describing how the head entity relates to the tail entity.
- ttt (**tail entity**) is the target or object of the relationship.

For example, consider a simple KG in the academic domain:

Head Entity	Relation	Tail Entity
Albert Einstein	bornIn	Ulm
Albert Einstein	studiedAt	ETH Zurich
ETH Zurich	locatedIn	Switzerland

Here, each triple captures a factual statement about entities and their relationships.

TYPES OF ENTITIES AND RELATIONS

Entities can represent:

- **People:** Scientists, authors, engineers
- **Organizations:** Universities, companies
- **Locations:** Cities, countries
- **Concepts:** Scientific theories, products, events

Relations can be:

- **Hierarchical:** isPartOf, subClassOf
- **Associative:** worksWith, collaboratesWith
- **Temporal/Contextual:** bornIn, foundedIn

1. Semantic Representation

Beyond triples, knowledge graphs often use semantic models to represent knowledge meaningfully:

- **Ontology-based KGs:** Use formal ontologies like OWL or RDF Schema to define entity types, properties, and constraints.

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

- **Property Graphs:** Each node and edge can have attributes (key-value pairs), allowing richer representation.

Example: Representing an author with properties:

Entity (Node)	Properties
Albert Einstein	{birthDate: 1879, nationality: German, field: Physics}

Edges can also carry properties:

Relation (Edge)	Properties
studiedAt	{from: 1896, to: 1900}

This allows dynamic reasoning over entities and relationships, such as querying who studied at a university in a given timeframe.

GRAPH CONNECTIVITY AND PATHS

Knowledge graphs are **connected structures**, where relationships form paths between entities. Path-based reasoning enables:

- **Inference:** Predicting new relations, e.g., if Einstein studied at ETH Zurich, and ETH Zurich is in Switzerland, then Einstein's education took place in Switzerland.
- **Graph Traversal:** Retrieving connected entities efficiently using algorithms like BFS, DFS, or shortest-path queries.

Table 1: Example of a Knowledge Graph Triple

Head Entity	Relation	Tail Entity
Albert Einstein	bornIn	Ulm
Ulm	locatedIn	Germany
Germany	partOf	Europe

2. Dynamic vs Static Knowledge Graphs

Knowledge graphs can broadly be classified into **static** and **dynamic** graphs based on how they

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

handle data updates and temporal changes. Understanding this distinction is essential for designing scalable graph systems and choosing the right construction and maintenance strategies.

1. Static Knowledge Graphs

Definition:

Static KGs are constructed at a given point in time and remain largely unchanged. They typically integrate well-curated, authoritative sources and undergo periodic updates rather than continuous modifications.

Characteristics of Static KGs:

1. **Batch Construction:** Data is processed in bulk from structured sources like Wikipedia, DBpedia, or scientific datasets.
2. **Low Update Frequency:** Updates occur occasionally, often manually curated or via scheduled scripts.
3. **High Data Quality:** Being curated, they are less prone to noisy or inconsistent data.
4. **Limited Temporal Reasoning:** Historical changes or evolving facts are not captured dynamically.

Examples:

- **DBpedia**: Extracted from Wikipedia infoboxes; updated periodically.
- **YAGO**: Combines Wikipedia and WordNet; primarily static.
- **Scientific Ontologies**: Gene Ontology (GO), which updates in batches.

Use Cases:

- Encyclopedic knowledge retrieval
- Academic research and domain-specific queries
- Question answering systems relying on stable facts

Limitation:

Static KGs cannot reflect real-time changes, making them unsuitable for applications that require up-to-date information.

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

2. Dynamic Knowledge Graphs

Definition:

Dynamic KGs continuously evolve by integrating new data, updating existing nodes and edges, and removing outdated information. They are designed to handle high-velocity and heterogeneous data streams, reflecting the latest state of knowledge.

Characteristics of Dynamic KGs:

1. **Incremental Updates:** New entities, relationships, or properties are added as data arrives.
2. **Temporal Awareness:** Capture changes over time; can represent when facts were true.
3. **High Scalability:** Must handle large-scale data efficiently using distributed frameworks or graph databases.
4. **Real-Time Query Support:** Users and applications can query the most recent graph state.

Examples:

- **Social Networks:** Graph of users, connections, and interactions (e.g., Facebook, LinkedIn) evolves continuously.
- **Financial Market KGs:** Entities include companies, stock trades, and news; relationships and properties change in real-time.
- **Industrial IoT KGs:** Sensor data from equipment or machines continuously updates the state of the system for predictive maintenance.

Use Cases:

- Real-time recommendation systems
- Fraud detection in financial transactions
- Predictive maintenance in manufacturing or energy networks

Challenges of Dynamic KGs:

1. **Temporal Changes:** Events or entity properties may change, requiring versioning or timestamping of triples.
2. **Incremental Graph Updates:** Efficiently integrating changes without rebuilding the entire graph is critical.

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

3. **Query Efficiency:** Maintaining fast query response times on a constantly evolving graph is challenging.

SCALABLE ARCHITECTURE FOR DYNAMIC KNOWLEDGE GRAPHS

Dynamic knowledge graphs (KGs) handle rapidly evolving information from heterogeneous sources. To maintain performance and accuracy at scale, they require a robust architecture that integrates **data ingestion, preprocessing, graph storage, incremental updating, and query processing**. A well-designed architecture ensures **real-time responsiveness**, **temporal awareness**, and **scalability** to millions of entities and relationships.

1. Data Ingestion and Preprocessing

Data ingestion is the first and most critical stage of dynamic KG construction. Dynamic KGs often receive data from multiple sources:

1. **Structured Data:** Databases, CSV/Excel files, knowledge bases.
2. **Semi-structured Data:** JSON, XML, logs, web APIs.
3. **Unstructured Data:** Text documents, news articles, social media posts, sensor readings.

Before integrating this data into a KG, **preprocessing** is essential to ensure quality, consistency, and usability:

KEY PREPROCESSING STEPS

Step	Description	Example
Entity Recognition	Identify entities (nodes) in the dataset	Recognize “Albert Einstein” as a person
Entity Disambiguation	Resolve ambiguity among entities	Differentiate “Apple” the company from “apple” the fruit
Relation Extraction	Identify relations (edges) between entities	Extract “studiedAt” between “Albert Einstein” and “ETH Zurich”

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

Step	Description	Example
Normalization & Canonicalization	Standardize entity names and relation types	Map "U.S.A.", "USA", and "United States" to a single entity

Example Workflow:

Unstructured text:

"Albert Einstein studied at ETH Zurich and later worked at Princeton University."

After preprocessing:

- Entities: {Albert Einstein, ETH Zurich, Princeton University}
- Relations: {(Albert Einstein, studiedAt, ETH Zurich), (Albert Einstein, worksAt, Princeton University)}

Tools & Techniques:

- Named Entity Recognition (NER) using **spaCy**, **Stanford NER**, or transformer-based models (BERT, RoBERTa)
- Relation extraction with **OpenIE**, dependency parsing, or deep learning methods
- Data normalization via dictionaries, ontologies, or knowledge bases

2. Incremental Graph Updating

Static KGs are unsuitable for real-time or evolving data. **Incremental updating** allows dynamic KGs to:

- Add new entities and relationships
- Modify existing nodes or edges
- Remove outdated information

TECHNIQUES FOR INCREMENTAL UPDATES

Technique	Description	Use Case
Delta-Based Updates	Only process changes (deltas) rather than the entire graph	Adding new Twitter interactions to a social network KG

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

Technique	Description	Use Case
Versioned Graphs	Maintain multiple versions for temporal reasoning and historical queries	Tracking employee positions over time in a corporate KG
Streaming Pipelines	Integrate data from real-time streams using frameworks like **Apache Kafka** or **Apache Flink**	Industrial IoT KGs that monitor equipment states

Example:

- At **T1**, the KG has the triple:
  (Albert Einstein, worksAt, Princeton University)
- New information arrives at **T2**:
  (Albert Einstein, awarded, Nobel Prize)
- Delta-based update adds only the new triple without rebuilding the graph

Benefits:

- Reduces computation time
- Enables real-time KG updates
- Supports temporal and historical queries efficiently

3. Distributed Graph Storage

Dynamic KGs can contain **millions of nodes and billions of edges**, making traditional single-node databases insufficient. **Distributed graph storage** allows parallel processing, high availability, and fault tolerance.

POPULAR DISTRIBUTED GRAPH SOLUTIONS

Database	Description	Strengths
Neo4j Fabric	Clustered version of Neo4j for distributed graph queries	Optimized for large-scale graph analytics and Cypher queries

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

Database	Description	Strengths
JanusGraph + HBase/Cassandra	Graph database using scalable NoSQL backends	High scalability, supports billions of nodes, strong consistency
TigerGraph	Native parallel graph computation engine	Extremely fast analytics, supports real-time streaming updates

Architecture Concept:

Advantages:

- Efficient handling of large graphs
- Fault tolerance and redundancy
- Parallel computation of graph algorithms (e.g., shortest path, link prediction)

4. Query Processing

Dynamic KGs are often queried **in real-time**, requiring efficient query mechanisms. Unlike static KGs, dynamic graphs may change during query execution, so traditional query engines must adapt.

Query Techniques

Technique	Description	Example
Incremental Graph Pattern Matching	Detect subgraph patterns efficiently by focusing only on updated portions	Detect fraud patterns in financial transactions
Continuous SPARQL Queries	SPARQL queries executed over streaming data with incremental updates	Retrieve all users who liked a post in the last 10 minutes
Graph Embedding-Based Searches	Approximate similarity search using node embeddings	Recommend new collaborators in a research KG using cosine similarity

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

Example:

- Query: "Find all scientists who won Nobel prizes after 1920 and worked at Princeton University."
- Dynamic KG ensures newly added information (e.g., new award winners) is included immediately in the results.

TECHNIQUES FOR SCALABLE KNOWLEDGE GRAPH CONSTRUCTION

Constructing dynamic knowledge graphs (KGs) at scale requires robust methods that can efficiently extract, integrate, and maintain entities and relationships from vast and evolving datasets. Techniques can be broadly categorized into **rule-based extraction**, **machine learning approaches**, and **temporal modeling**. Each has its advantages and limitations depending on the domain, data type, and scalability requirements.

1. Rule-Based Extraction

Definition:

Rule-based extraction uses predefined domain knowledge, ontologies, or linguistic patterns to identify entities and relationships in text or structured data. These rules define how data should be interpreted and mapped into graph triples.

Advantages:

- High precision for well-understood domains
- Easy to interpret and debug
- Compatible with ontologies and controlled vocabularies

Limitations:

- Poor adaptability to new domains or evolving data
- Requires significant manual effort to define rules
- Cannot handle noisy or unstructured data effectively

Example:

- Text: "Albert Einstein received the Nobel Prize in 1921."
- Rule-based extraction: Identify patterns like [Person] received [Award] in [Year]
- Output triple: (Albert Einstein, awarded, Nobel Prize)

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

Applications:

- Scientific databases
- Legal document analysis
- Regulatory compliance knowledge graphs

Rule-based systems are often combined with machine learning for hybrid approaches that balance precision and adaptability.

2. Machine Learning Approaches

Machine learning methods provide scalability and adaptability for extracting entities and relations, especially from unstructured and noisy data.

SUPERVISED AND SEMI-SUPERVISED LEARNING

Supervised Learning:

- Models are trained on labeled datasets where entities and relationships are annotated.
- Algorithms include Conditional Random Fields (CRFs), Support Vector Machines (SVMs), and modern transformer-based models like BERT or RoBERTa.

Semi-Supervised Learning:

- Uses a small labeled dataset and large unlabeled data to improve model performance.
- Reduces the high cost of manual annotation while maintaining reasonable accuracy.

Example:

- Labeled dataset: Sentences annotated with entity/relation triples
- Model predicts new triples from incoming text streams
- Use case: Extracting company partnerships from financial news

Advantages:

- Can generalize to new contexts with enough training data
- Handles unstructured or semi-structured data better than rule-based systems

GRAPH NEURAL NETWORKS (GNNS)

Definition:

GNNs learn representations of nodes and edges by aggregating information from their neighbors.

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

They are particularly suitable for link prediction, entity classification, and graph completion in dynamic KGs.

Scalable Variants:

- **GraphSAGE:** Samples neighborhoods to generate embeddings efficiently for large graphs
- **Cluster-GCN:** Clusters nodes to enable mini-batch training on massive graphs

Applications:

- Predicting new collaborations in research networks
- Detecting anomalies in social network graphs
- Recommender systems

Example:

- Input: KG with nodes representing researchers and edges representing collaborations
- GNN predicts potential collaborations based on network structure
- Output: New predicted links that can be added to the dynamic KG

KNOWLEDGE GRAPH EMBEDDINGS

Definition:

KG embeddings map entities and relations to continuous vector spaces while preserving structural and semantic properties. These embeddings enable scalable reasoning, link prediction, and similarity searches.

Popular Models:

- **TransE:** Models relations as translations in vector space; simple and efficient
- **RotatE:** Represents relations as rotations in complex vector space; captures symmetric and antisymmetric relations
- **DistMult:** Factorizes triple relationships; suitable for multi-relational data

Advantages:

- Efficient for very large graphs
- Supports approximate reasoning for link prediction and entity inference
- Reduces memory and computational requirements for dynamic updates

Example:

- KG triple: (Albert Einstein, worksAt, Princeton University)

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

- Embedding maps entities and relation to vectors: vec(Albert Einstein) + vec(worksAt) ≈ vec(Princeton University)
- Enables prediction of missing triples or new links in the KG

TEMPORAL KNOWLEDGE GRAPHS

Definition:

Dynamic KGs must account for temporal changes, i.e., the fact that relationships and entity properties may evolve over time. Temporal KGs extend standard triples by adding a timestamp: (h,r,t,timestamp)(h, r, t, timestamp)(h,r,t,timestamp)

Temporal KG Models:

Model	Description	Strengths
TTransE	Extends TransE with temporal embeddings	Captures temporal evolution of relationships
HyTE	Projects entities and relations into time-specific hyperplanes	Models temporal dynamics and link prediction over time

Applications:

- Event prediction (e.g., financial market changes, disease outbreaks)
- Tracking historical changes (e.g., company mergers, academic collaborations)
- Real-time recommendation systems

Example:

- Triple with timestamp: (Albert Einstein, worksAt, Princeton University, 1933-1955)
- Triple added later: (Albert Einstein, awarded, Nobel Prize, 1921)
- Temporal reasoning allows queries like: “Where did Einstein work during the 1940s?”

Benefits:

- Enables dynamic and temporal queries
- Supports trend analysis and predictive modeling in evolving domains

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

CHALLENGES IN DYNAMIC KNOWLEDGE GRAPH CONSTRUCTION

1. Data Heterogeneity

- Data from diverse sources may have inconsistent formats, labels, and granularities.
- Integration requires standardization and schema alignment.

2. Graph Sparsity and Incompleteness

- Many real-world KGs are incomplete.
- Predictive models and embeddings help infer missing links.

3. Scalability and Performance

- Real-time updates on large graphs are computationally intensive.
- Distributed frameworks and approximate algorithms are often required.

4. Evaluation Metrics

- **Precision/Recall/F1** for extracted triples
- **Graph consistency and coverage**
- **Update latency and throughput** for dynamic processing

CASE STUDIES

1. Biomedical Knowledge Graphs

Dynamic KGs in genomics and drug discovery integrate literature, experimental results, and clinical trials. GNNs and embeddings help infer novel interactions between genes, proteins, and diseases.

2. Social Network Graphs

Platforms like Twitter or LinkedIn require real-time graph updates to model user connections, trends, and content recommendation.

3. Industrial IoT

IoT sensor networks generate streaming data representing equipment states. Dynamic KGs facilitate predictive maintenance and anomaly detection.

FUTURE DIRECTIONS

1. **Integration of Large Language Models:** LLMs can enhance entity and relation extraction from unstructured data.

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

2. **Hybrid Graph Construction Pipelines:** Combining rule-based and ML-based methods for higher precision.
3. **Self-evolving Graphs:** Fully autonomous systems that update with minimal human supervision.
4. **Explainable KG Analytics:** Better interpretability of link predictions and embeddings.
5. **Edge Computing for Graph Updates:** Processing near data sources for latency reduction in IoT applications.

CONCLUSION

The construction of scalable dynamic knowledge graphs is a cornerstone for next-generation information systems. This paper reviewed the methods, architectures, and algorithms enabling the continuous growth and adaptability of KGs. Dynamic KGs address the limitations of static knowledge graphs by incorporating temporal reasoning, incremental updates, and distributed processing. Despite advances in GNNs, embeddings, and distributed storage, challenges remain in handling heterogeneity, sparsity, and real-time scalability. Future research focusing on LLM-driven extraction, autonomous graph evolution, and edge-based updates will likely define the next frontier of dynamic knowledge graph technologies.

REFERENCES

1. Hogan, A., Blomqvist, E., Cochez, M., et al. (2021). Knowledge Graphs. *ACM Computing Surveys*, 54(4), 1–37.
2. Wang, Q., Mao, Z., Wang, B., & Guo, L. (2017). Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12), 2724–2743.
3. Rossi, R. A., & Ahmed, N. K. (2015). Role Discovery in Networks. *IEEE Transactions on Knowledge and Data Engineering*, 27(4), 1112–1131.
4. Hamilton, W., Ying, R., & Leskovec, J. (2017). Inductive Representation Learning on Large Graphs. *NeurIPS*.
5. Tran, T., Pham, M., Nguyen, T., & Collier, N. (2020). Dynamic Knowledge Graph Construction from Streaming Data. *Journal of Web Semantics*, 64, 100583.

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved

6. Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Computing Surveys*, 52(1), 1–38.
7. Ji, S., Pan, S., Cambria, E., et al. (2021). A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Transactions on Neural Networks and Learning Systems*, 32(11), 4942–4967.
8. Nickel, M., Murphy, K., Tresp, V., & Gabrilovich, E. (2016). A Review of Relational Machine Learning for Knowledge Graphs. *Proceedings of the IEEE*, 104(1), 11–33.
9. Chen, Y., Zheng, H., Xie, X., et al. (2022). Real-Time Knowledge Graph Construction for Industrial IoT Systems. *IEEE Internet of Things Journal*, 9(12), 11023–11035.
10. Dettmers, T., Minervini, P., Stenetorp, P., & Riedel, S. (2018). Convolutional 2D Knowledge Graph Embeddings. *AAAI Conference on Artificial Intelligence*.

Page 56-72 © MANTECH PUBLICATIONS 2020. All Rights Reserved