OB-GRAG: LLM Assisted Graph Creation and Querying via Domain Specific Ontology*

Paulis Barzdins[1,*], Normunds Gruzitis[1]

[1] *Institute of Mathematics and Computer Science, University of Latvia, Raina bulvaris 29, Riga, LV-1459, Latvia*

Abstract

LLMs struggle with long-term memory, particularly in storing, organising, and accessing domain-specific known facts. RAG-based methods show practical promise but lack dynamism (continuous knowledge updates) and structure (interrelated facts following domain-specific logic and rules). Graph RAG addresses some of these issues, but full structural benefits require an accompanying ontology.

This paper introduces ontology-based graph RAG (OB-GRAG), which uses narrow, specific ontologies as a domain focus to create a graph-based RAG that supports dynamic updates and structural reasoning. The method involves defining the ontology, creating "actions" for valid property graph creation, and translating human questions into Cypher queries via LLMs. We demonstrate this using raw TextWorld game observations, which are dynamic data with a structure of interest, fit for complex querying. The result is a detailed methodology for OB-GRAG's initial version, demonstrated on TextWorld data.

Keywords

graph RAG, ontology, long-term memory, LLM, TextWorld

1. Introduction

LLMs excel in short-term memory within their context window and possess extensive general world knowledge embedded in their model weights. However, they struggle with long-term memory, which is crucial for receiving, structuring, and querying known facts. This limitation can lead to hallucinations [21, 22], inconsistent answers, and difficulties in fact attribution.

A real world problem that motivated this work is a project on workforce and human capital. The client has disparate datasets on tax data, education, job vacancies. The answer to the client's questions should be derivable from the entirety of this data, but there isn't a way to, for example, fit all this information into the context window of an LLM, and ask the questions of interest (especially not at the needed scale). The challenge is to make this comprehensive knowledge, "memory", accessible to an LLM, enabling effective data interaction, querying, and reasoning.

There are several approaches to address LLM limitations. Fine-tuning is used in some scenarios, and efforts are underway to significantly expand the context window size. However currently, especially for this style of problem, the most popular is the use of Retrieval Augmented Generation (RAG). RAG retrieves relevant data sections and integrates them into the LLM's context, allowing these knowledge bits to be utilised. But RAG is inadequate when the required knowledge isn't available as a direct fact, and reasoning over the whole is needed [2, 21].

Graph RAG, in its various implementations, addresses this by structuring knowledge into graphs, grouping related items, and providing multi-level summaries [10]. This enables some reasoning and information retrieval from the whole, but these methods often lack dynamism and structure.

Not being dynamic is that the graph, or at the minimum the summaries at different levels, need to be regenerated after any addition of new data. This means the method isn't applicable in uses where new data is constantly coming in.

* *ICTCS'25: Italian Conference on Theoretical Computer Science, September 10–12, 2025, Pescara, Italy*

1,* Corresponding author.

paulis.barzdins@lumii.lv (P. Barzdins); normunds.gruzitis@lu.lv (N. Gruzitis)

0009-0006-7186-9776 (P. Barzdins); 0000-0003-0511-1829 (N. Gruzitis)

Not being structured is that the graph, while being a graph, doesn't have a particularly useful schema. It is usually in no way tailored to the specific data that it stores, thus there is no ability for structural rule-based queries, or easy aggregation and multi-hop reasoning.

For these full benefits an ontology is needed alongside the graph. What use cases benefit from an ontology? Sharma et al [21] say it is when there is a need for fact based reasoning. Where decisions are made following strict rules and procedures. Industrial fields, medical, judicial, farming; but also knowledge work like journalism, research, consulting. These fields filter and organise facts and their relations in proper domain specific structures and systems; this structure is then beneficial to use in analysis. Classical RAG and most graph RAG fail to capture this strict and organised structure.

We propose an ontology-based graph RAG (OB-GRAG). It begins with defining a narrow domain-specific ontology, then transforming data into a property graph using this ontology as a schema (for this step we propose the use of "actions" to ensure valid graphs). The final step involves a *Text to Cypher* task [15, 26], where a small, specific schema enhances result quality [22]. This final step makes data accessible to non-experts without specialised Cypher querying skills.

We use raw TextWorld [8] game instance observations as example input data, which have an extractable underlying structure. The game is dynamic which allows to test that aspect. Also, current TextWorld LLM solutions struggle with navigation [2], which is structure extraction and multi-hop queries – challenges that the proposed OB-GRAG aims to tackle.

2. Related work

Significant efforts have been made to enhance the long-term memory of LLMs by increasing context window length. Approaches like MAMBA [14], RMT [5, 6], ARMT [19], and RWKV [18] have at best achieved up to 80% fact retrieval from a 50 million token context window, though these technologies, at the authors' admission, are not yet ready for practical use.

Numerous RAG and graph RAG implementations exist [1, 10, 20, 25], but they face the aforementioned limitations. Though some approaches recognise the value of ontologies; for instance, OG-RAG [21] uses an ontology but creates a flattened hyper-graph, losing structural query capabilities. CypherBench [12] aligns closely with our work, splitting enormous unmanageable RDF tables like WikiData [24] into smaller property graphs with strict schemas, enabling human questions to be converted into Cypher queries. Its preprint was released in April 2025, concurrent with our work.

In the TextWorld domain, we considered works [2, 7] and our previous experience [3].

Lastly, research on knowledge graph and LLM combinations [9, 11, 16, 17, 22, 23, 27] highlights the benefits LLMs bring to this field (compared to earlier manual approaches [4]), noting that narrow schemas are essential for successful automation.

3. Methodology and implementation

Here we make OB-GRAG on TextWorld data as an example. Creating the ontology that defines the question domain we are interested in. Input is the raw game observations, and we return this data parsed into a property graph that strictly follows our ontology as a schema (strict validity enabled by the use of defined "actions"). Then the graph is queryable by using an LLM to write Cypher [13] queries from human questions.

3.1. Ontology

The ontology defines the specific question domain we are interested in. It will act as a strict schema for the property graph. We want to make it narrow and specific as that improves automatisation, as seen in [22].

In that paper the authors noted an example of an ontology alignment task, with the sizes being 40 classes, 149 object properties, 49 data properties for one, and 156 classes, 124 object properties,

46 data properties for the other. The LLM failed to align them. But when split into twenty naturally occurring modules, and working module by module, the LLM managed to achieve high precision and recall. This gives some estimate of ontology sizes. Though the limits observed in practice will likely be a balance, where the specific LLM model ability and prompt descriptiveness quality will dictate the size of the ontology; giving some flexibility to influence capability, simplicity, and cost.

In this example we choose to keep track of players within the game, rooms, exits, and items. Where the player is on each turn. Where items are on different turns. What items the player has picked up and used. All the rooms and how they are connected. Exits seen from each room, so that unexplored exits can later be found. This is a subset of the information that could be tracked, but it is a useful proof of concept.

Such an ontology would allow us to answer questions like “what is the shortest path to the kitchen?”, “where are all the unexplored exits?”, “in how many games are bedrooms next to toilets?”, “what is the greatest number of items that a player has had in their inventory?”.

An intermediary ontology drawing can be used to make sure the interested parties agree on the question domain, but the final output has to be a property graph schema. Thus the ontology and schema used in this work is the following in Figure 1.

```
NODES                                RELATIONSHIPS
     Player                               player IS_IN room
          game_id                              turn
     Room                                 item IS_IN room
          game_id                              turn
          name                            player HAS item
     Exit                                      turn
          game_id                         player USED item
          name: {room_name}_{dir               turn
     Item                                 room RELATES room
          game_id                              direction
          name                            room HAS_EXIT exit
                                               direction
```

Figure 1: The ontology and schema for the TextWorld data example. Nodes are the players, rooms, exits, items. They have relationships. Important are the properties “turn” that track on which turn the relationship was observed, which enables temporal queries.

3.2. Graph creation

The graph creation has two main components. Defining a set of “actions”, and then a prompt that has the LLM call the appropriate actions for each game observation.

We define these actions to match the gameplay, so that there is a simple high-level API for the LLM to use. These actions then take care of the data entry into the graph, making sure it strictly follows the schema, by asking for a specified list of parameters, then executing predefined Cypher code with those parameters. For our use case the actions are “new_game”, “entered_room”, “found_item”, “item_to_inventory”, “put_down_item”, “use_up_item”. Parameter examples are “room name”, “direction”, “items in room”, though they vary for each action.

Once we have a set of actions, we can write a prompt for the LLM. As input it receives the action taken and the resulting game observation. From this the LLM is instructed to select the appropriate actions and pass to them the relevant parameters. The prompt structure is “intro; game observation; list of actions and their parameters; instructions; examples”.

When the LLM returns a list of actions and their parameters the relevant action Cypher codes are executed to enter the data into the graph.

3.3. Queries

We now have a property graph that strictly follows our schema. The final step is querying this data. For that we again use an LLM to receive human questions, and translate them to Cypher queries. The prompt contains the ontology/schema we defined in Section 3.1. which allows the LLM to write proper queries. The prompt structure is “task; ontology; hints; question”.

Automatic error correction is included, by repeating the prompt upon failure with “previous attempt; error message” appended to the end. These types of failures the LLM can often fix itself. A worse scenario is if a query runs successfully, but its logic is flawed. Currently there is no automatic system to catch these, the user would need to know Cypher or know what answers to expect. To avoid this, it is advised to use the best available LLM (in this paper for this step we used OpenAI *gpt-o3-mini*). Eventual solutions could involve developing more sophisticated prompts or integrating additional layers of validation to ensure query accuracy and thus system robustness.

4. Results

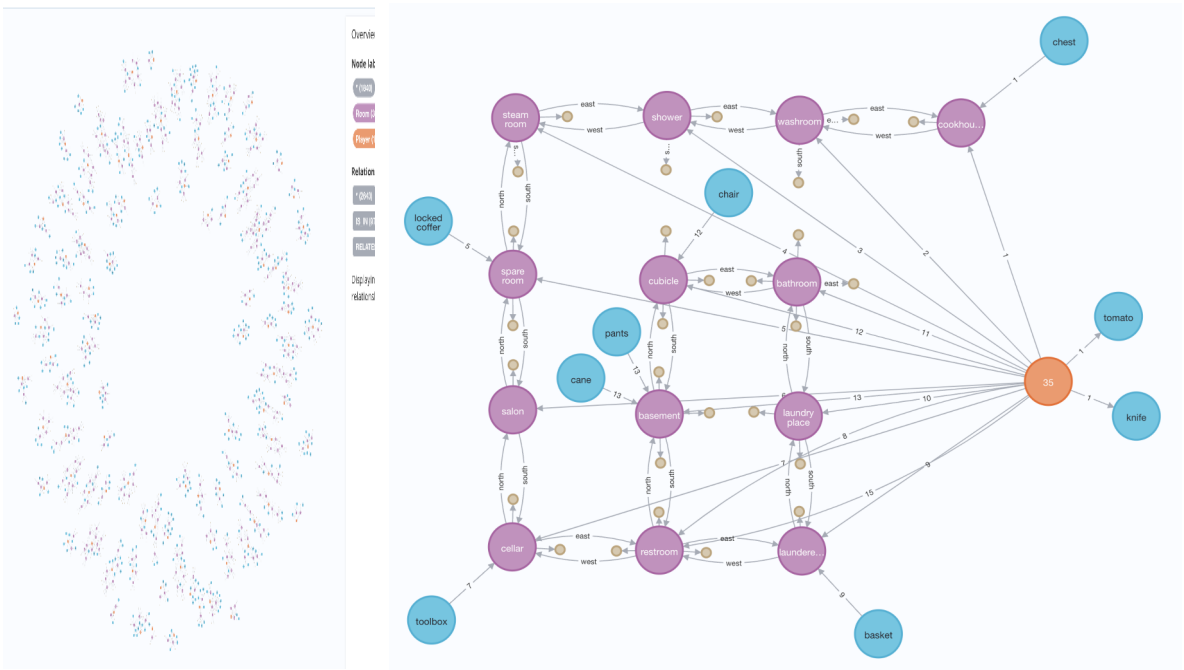

Figure 2: (left) All 100 game instance graphs. (right) One example graph of the largest game instance by node count, showing the player (orange), items (blue), rooms (purple), exits (brown); where what was on each turn, and how they are connected.

4.1. Graph creation results and discussion

For the input data a 100 game walkthrough instances were created, of different sizes, quest lengths, item counts. For each game that is a sequence of textual observations for each turn, as well as the action performed by the agent. Figure 2 shows an overview of all the instances, as well as a close up of the largest game instance by node count.

There isn’t a trivial way to check the correctness of the whole graph, thus a combination of systemic sanity checks and manual examinations was used.

There are the expected 100 separate instance graphs. No rooms have more than the expected single connection between them (no room is both, say, east and south of another room). One mislabeled room was found, that caused pathways to merge incorrectly (a pantry was mistakenly labeled as a kitchen, thus merging their exits). There are 37 missing exit nodes, where there are room connections (5 of these can be seen missing in Figure 2, right).

The authors' opinion is that these issues can be solved with better prompts. The current prompts are very minimalistic, without explaining what is an exit (whether the direction you entered is also an exit, whether locked doors count, etc.), or that the room name should be taken directly from the observation without any guess work.

4.2. Query results and discussion

The system was asked various questions – the 4 sanity checks on the graph were done using this question function, as well as 11 other questions to check the ability to aggregate information, perform multi-hop reasoning, and to overall test the system.

Thirteen of the questions were answered correctly on the first try. For one question an error was returned about incorrect shortest path command usage, but the system was able to resolve the error on its own. For one question an answer was returned but the query was non-sensical; the same question was asked a second time and then a correct answer was returned.

Overall the system demonstrated impressive capabilities in question answering, and in the authors' opinion show great promise in this approach. Some highlight questions are included here.

Some of the questions answered correctly on the first go:

- "Show me rooms that relate to other rooms in directions, where they don't have exits"
- "Return all rooms that have unexplored exits"
- "In which games was the player in a bedroom after they were in a bathroom?"
- "Does the presence of a knife within the game influence the map size?"

The question where the system successfully error corrected itself:

- "In game 18, return the shortest paths from the players current room, to the rooms with unexplored exits"

The question where non-sensical Cypher was returned and needed a re-run:

- "In what room are knifes usually in?"

5. Conclusions

An early version of OB-GRAG was successfully created and demonstrated on the TextWorld example. With the use of a specific ontology and actions, an LLM can create a property graph that strictly follows the ontology/schema. This narrow specific ontology then allowed for successful *Text to Cypher* creation for the question answering section.

The system is dynamic and structured, able to store and use the specific knowledge structure of the domain. It provided a TextWorld solution that is able to navigate in the game world, as well as answer other difficult questions.

This initial version can now be improved towards a more complete RAG solution. Major improvements are expected with more detailed prompt engineering, taking current failures as guidelines on how to improve the prompt. This method can also now be applied to actual practical client needs and data (as is being done in parallel to this work).

Acknowledgements

This work was funded by the EU Recovery and Resilience Facility's project "Language Technology Initiative" (2.3.1.1.i.0/1/22/I/CFLA/002).

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] Allemang, D., 2024. LLMs, Knowledge Graphs and Property Graphs. https://medium.com/@dallemang/llms-knowledge-graphs-and-property-graphs-5b6fc2cf9f55

[2] Anokhin, P., Semenov, N., Sorokin, A., Evseev, D., Burtsev, M. and Burnaev, E., 2024. Arigraph: Learning knowledge graph world models with episodic memory for llm agents. arXiv preprint arXiv:2407.04363.

[3] Barzdins, G., Gosko, D., Barzdins, P.F., Lavrinovics, U., Bernans, G. and Celms, E., 2019, August. Rdf* graph database as interlingua for the textworld challenge. In 2019 IEEE Conference on Games (CoG) (pp. 1-2). IEEE.

[4] Barzdins, G., Gosko, D., Cerans, K., Barzdins, O.F., Znotins, A., Barzdins, P.F., Gruzitis, N., Grasmanis, M., Barzdins, J., Lavrinovics, U. and Mayer, S.K., 2020. Pini Language and PiniTree Ontology Editor: Annotation and Verbalisation for Atomised Journalism. In The Semantic Web: ESWC 2020 Satellite Events: ESWC 2020 Satellite Events, Heraklion, Crete, Greece, May 31–June 4, 2020, Revised Selected Papers 17 (pp.32-38). Springer International Publishing.

[5] Bulatov, A., Kuratov, Y. and Burtsev, M., 2022. Recurrent memory transformer. Advances in Neural Information Processing Systems, 35, pp.11079-11091.

[6] Bulatov, A., Kuratov, Y., Kapushev, Y. and Burtsev, M.S., 2023. Scaling transformer to 1m tokens and beyond with rmt. arXiv preprint arXiv:2304.11062.

[7] Chen, L., Wang, L., Dong, H., Du, Y., Yan, J., Yang, F., Li, S., Zhao, P., Qin, S., Rajmohan, S. and Lin, Q., 2023. Introspective tips: Large language model for in-context decision making. arXiv preprint arXiv:2305.11598.

[8] Côté, M.A., Kádár, A., Yuan, X., Kybartas, B., Barnes, T., Fine, E., Moore, J., Hausknecht, M., El Asri, L., Adada, M. and Tay, W., 2019. Textworld: A learning environment for text-based games. In Computer Games: 7th Workshop, CGW 2018, Held in Conjunction with the 27th International Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden, July 13, 2018, Revised Selected Papers 7 (pp. 41-75). Springer International Publishing.

[9] Dai, X., Hua, Y., Wu, T., Sheng, Y. and Qi, G., 2024. Counter-intuitive: Large Language Models Can Better Understand Knowledge Graphs Than We Thought. arXiv preprint arXiv:2402.11541.

[10] Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S. and Larson, J., 2024. From local to global: A graph rag approach to query-focused summarization. arXiv preprint arXiv:2404.16130.

[11] Fathallah, N., Das, A., Giorgis, S.D., Poltronieri, A., Haase, P. and Kovriguina, L., 2024, May. Neon-GPT: a large language model-powered pipeline for ontology learning. In European Semantic Web Conference (pp. 36-50). Cham: Springer Nature Switzerland.

[12] Feng, Y., Papicchio, S. and Rahman, S., 2024. CypherBench: Towards Precise Retrieval over Full-scale Modern Knowledge Graphs in the LLM Era. arXiv preprint arXiv:2412.18702.

[13] Francis, N., Green, A., Guagliardo, P., Libkin, L., Lindaaker, T., Marsault, V., Plantikow, S., Rydberg, M., Selmer, P. and Taylor, A., 2018, May. Cypher: An evolving query language for property graphs. In Proceedings of the 2018 international conference on management of data (pp. 1433-1445).

[14] Gu, A. and Dao, T., 2023. Mamba: Linear-time sequence modeling with selective state spaces. arXiv preprint arXiv:2312.00752.57

[15] Hornsteiner, M., Kreussel, M., Steindl, C., Ebner, F., Empl, P. and Schönig, S., 2024. Real-Time Text-to-Cypher Query Generation with Large Language Models for Graph Databases. Future Internet, 16(12), p.438.

[16] Meyer, L.P., Stadler, C., Frey, J., Radtke, N., Junghanns, K., Meissner, R., Dziwis, G., Bulert, K. and Martin, M., 2023, June. Llm-assisted knowledge graph engineering: Experiments with chatgpt. In Working conference on Artificial Intelligence Development for a Resilient and Sustainable Tomorrow (pp. 103-115). Wiesbaden: Springer Fachmedien Wiesbaden.

[17] Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J. and Wu, X., 2024. Unifying large language models and knowledge graphs: A roadmap. IEEE Transactions on Knowledge and Data Engineering.
[18] Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Biderman, S., Cao, H., Cheng, X., Chung, M., Grella, M. and GV, K.K., 2023. Rwkv: Reinventing rnns for the transformer era. arXiv preprint arXiv:2305.13048.
[19] Rodkin, I., Kuratov, Y., Bulatov, A. and Burtsev, M., 2024. Associative recurrent memory transformer. arXiv preprint arXiv:2407.04841.
[20] Shah, S., Ryali, S. and Venkatesh, R., 2024. Multi-Document Financial Question Answering using LLMs. arXiv preprint arXiv:2411.07264.
[21] Sharma, K., Kumar, P. and Li, Y., 2024. OG-RAG: Ontology-Grounded Retrieval-Augmented Generation For Large Language Models. arXiv preprint arXiv:2412.15235.
[22] Shimizu, C. and Hitzler, P., 2025. Accelerating knowledge graph and ontology engineering with large language models. Journal of Web Semantics, p.100862.
[23] Vandemoortele, N., Steenwinckel, B., Hoecke, S.V. and Ongenae, F., 2024. Scalable Table-to-Knowledge Graph Matching from Metadata using LLMs. In Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 23rd International Semantic Web Conference (ISWC 2024) (pp. 54-68)
[24] Vrandečić, D. and Krötzsch, M., 2014. Wikidata: a free collaborative knowledgebase. Communications of the ACM, 57(10), pp.78-85.
[25] Wu, M., Castelló, N.B., Marszałek-Kowalewska, K. and Singh, A., 2024. Transforming Natural Language into Cypher Queries: GPT-4o as a Bridge to Neo4j Graph Data. https://medium.com/@wumingzhu1989/transforming-natural-language-into-cypher-queries-gpt-4o-as-a-bridge-to-neo4j-graph-data-072b6ff1b5ba
[26] Zhong, Z., Zhong, L., Sun, Z., Jin, Q., Qin, Z. and Zhang, X., 2024. Synthet2c: Generating synthetic data for fine-tuning large language models on the text2cypher task. arXiv preprint arXiv:2406.10710.
[27] Zhu, Y., Wang, X., Chen, J., Qiao, S., Ou, Y., Yao, Y., Deng, S., Chen, H. and Zhang, N., 2024. Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities. World Wide Web, 27(5), p.58.