IAC-23,B6,IP,9,x75673

Question Answering over Knowledge Graphs for Explainable Satellite Scheduling

Cheyenne Powell(1, a) **and Annalisa Riccardi**(1, b)

(1) *Department of Mechanical and Aerospace Engineering, University of Strathclyde, 75 Montrose Street, Glasgow G1 1XJ, United Kingdom.*
(a) *cheyenne.powell@strath.ac.uk.*
(b) *annalisa.riccardi@strath.ac.uk.*

Abstract

Schedules for satellite missions consist of thousands, if not millions, of interconnected activities executing many times across days, months, and years to fulfill mission objectives. The complexity of a schedule can make it difficult for Ground Station Operators (GSO) to understand the relationship between activities as part of a complete mission, especially when schedules have been created by means of an autonomous decision making algorithm. Text-based explanations are helpful in establishing the reasoning behind decisions suggested by algorithms and their impact on the overall execution plan.

A Knowledge Graph (KG) can provide the underlying data structure to record what has happened and what is scheduled, as well as the interconnected elements that are impacted by the scheduled activities. The relationship between satellite components, environmental conditions, operational constraints, and mission objectives is complex and highly dimensional, which is not easy for a single operator to manage concurrently. A system that can gather information from a KG, and infer the information stored within, can assist human operators in building a deeper understanding of the relationships of automatically scheduled decisions. A natural language query interface to the KG is the simplest way for a human to interface and extract knowledge. Additionally, manual access to the KG can be provided alongside textual answers, enabling exploration of schedule branches to understand what else can change throughout the mission's execution. This improves the robustness of a system's responses to queries and allows for greater flexibility.

An overview is therefore examined of how KG and Natural Language Processing (NLP) technologies can be used to facilitate eXplainable Artificial Intelligence (XAI) in satellite scheduling. Namely, how to model the schedule and environment information to be stored in the graph and how to reason over such information by interpreting user queries in natural language. An example is presented demonstrating the capabilities of flexible query interpretation on an Earth Observation (EO) satellite scheduling problem. Finally, the capabilities of KG and NLP technologies to provide more explainable insights into satellite scheduling tasks are discussed in the frame of future possible developments.

1 Introduction

Satellite scheduling is a complex problem due to the vast number of variables involved, such as communication bandwidth [1], power consumption [2], data acquisition [3], and mission priorities [4]. These schedules were traditionally created using rule-based approaches with the need for manual intervention and therefore may not be suitable for modern satellite constellations that require dynamic and adaptive scheduling to handle diverse missions and changing environmental conditions.

Artificial Intelligence (AI) techniques for satellite scheduling have been introduced to improve scheduling efficiency and optimization for mission operations by means of embedding adaptability capabilities, and responsiveness to changing conditions. A review of AI techniques for the next generation of Low Earth Orbit (LEO) satellite networks was conducted. The researchers outlined the fundamentals and unique features of massive satellite networks and presented state-of-the-art AI techniques for channel forecasting, spectrum sensing, signal detection, inter-satellite and access network optimization, and network security. They also discussed future paradigms for AI implementation in next-generation massive satellite networks for fully autonomous and robust systems [5].

Another example of recent developments of autonomous on-board decision-making algorithms is Project for On-Board Autonomy (PROBA) [6]. Operational case studies were presented to demonstrate the applicability of the PROBA framework, which combined AI-based techniques, spacecraft flight data information, and discipline models to address spacecraft anomaly root cause analysis, spacecraft prediction behaviour, and discipline model refinement[7].

Techniques such as Genetic Algorithms (GA) [8], Hybrid Discrete Particle Swarm Optimization (HDPSO) [9], and Reinforcement Learning (RL)[10] have shown promising results in improving satellite scheduling efficiency and flexibility.

XAI has emerged as a key research area to address the transparency and interpretability challenges of AI systems. As AI models become increasingly complex, their decision-making processes can become opaque and difficult to interpret, raising concerns about trust and accountability in high-stakes applications such as healthcare [11], finance [12], autonomous vehicles [13] and satellite operations [14].

There are several explanation methods developed in recent years. Sensitivity Analysis (SA) which explains the prediction of a model locally evaluated gradient, layer-wise Relevance Propagation (LRP) which decomposes the decisions of classifiers [15], Local Interpretable Model-agnostic Explanations (LIME) where local surrogate interpretable models are created around specific instances to approximate the behavior of the complex AI model[16], and SHapley Additive exPlanations (SHAP), a method that based on game theory and computes the contribution of each feature to a prediction by considering all possible feature combinations[17]. XAI approaches provide insights into the factors influencing AI predictions, enabling users to gain confidence in the system's reliability and make informed decisions.

With regard to satellite scheduling, XAI is deemed important, as scheduling errors can have implications for communication, weather monitoring, disaster response, and various other applications. Additionally, where there are no errors, explanations may be required for reasons behind the processes occurring on-board.

Moreover, to effectively communicate the explanations to end-users, various communication techniques are being explored. These range from visualizations through the means of charts [18], textual explanations [19], and KGs [20].

KGs are structured representations of information that use graph-based models to capture and organize data. They consist of nodes (entities) representing real-world objects and edges (relationships) connecting the entities of interest, representing the associations between them, along with any applicable attributes [21, 22, 23].

The versatility in the application of Knowledge Graphs (KGs) has been shown in various industries to enhance existing AI systems and decision-making processes. In healthcare, KGs are used to model patient data facilitating the understanding of drug interactions, clinical notes, and prescriptions. Additionally, they also assist in identifying misinformation in clinical settings [24]. In digital marketing, such as touristic services, the utilization of KGs was applied to online marketing and sales strategy solutions [25]. KGs are used within the NLP domain by providing structured knowledge and context for language understanding, question answering, Recommender System (RS), and text summarization [22]. In additive manufacturing, KGs also enhances automated and autonomous construction and improvements of manufacturing design rules [26]. For power grid management KGs have been tasked with presenting massive amounts of data from a wide array of sources to enable day-ahead scheduling optimization, supporting the use of autonomous systems [27].

NLP has recently gone through a major development with the introduction of Large Language Models (LLMs), which contain significantly more words and phrases to create more accurate and complex responses [28]. The capability of LLMs has proved to be able to complete complicated reasoning, effectively adding an additional method for XAI, in a study exploring the 'least-to-most' prompting technique, that breaks down queries into a series of smaller component problems. By doing so the Large Language Model (LLM) is able to become increasingly accurate in its responses as the number of steps was increased, outperforming other methods, such as 'chain-of-thought' [29]. In another study, LLMs were examined for their performance in reasoning based on zero-shot learning (where no prior prompting is conducted). By adding a simple instructive phrase to the answer ('Let's think step by step') the researchers were able to produce results that were more accurate than standard zero-shot, or even few-shot, methods. This underlined the importance of how prompts are structured as even slight alterations can lead to vastly different results [30].

In the application of satellite resources, nodes within a KG may be used to represent individual satellites with their respective edges representing their orbits, and their capabilities. However, pertaining to ground stations, nodes may represent each ground station with edges representing their locations, communication capabilities, and capacities. With respect to task requirements, allocations, and constraints; nodes may also represent the respective tasks, with edges representing their requirements, dependencies, constraints such as temporal, and spatial, as well as priorities and execution times. Overall, KGs can store historical scheduling data, allowing for trend analysis and learning from past schedules.

Integrating knowledge graphs into satellite scheduling AI systems allows for the incorporation of domain-specific knowledge and context, providing a more comprehensive view of scheduling decisions. By leveraging the power of knowledge graphs, AI models can be equipped with the ability to explain their reasoning, ensuring that GSO can understand and validate the scheduling outcomes. This transparency is crucial in ensuring the reliability and safety of satellite operations, where even subtle errors can have significant consequences.

Examples of questions that a KGs model, designed to support satellite scheduling on board of an EO mission, can answer are:

- How many times is location *X* accessible?
- When is land visible during time range *X*-*Y*?
- What are the environmental conditions during time *T*?
- Show all instances of ground stations being visible during week *W* and the scheduled actions.
- Which actions are scheduled when current capacity is above *A* Mb?
- What happened between times *X* and *Y*?

With the support of LLMs, the following questions can also be answered.

- Can action *A* replace *B* at time *T*?
- Where can action *A* replace action *B*?

This paper provides a novel approach to how KGs with LLMs may be used to contribute to the development of XAI for satellite scheduling for task execution.

Following this introduction, Section 2 briefly introduces the satellite scheduling problem, then considers how KGs and LLMs may be used. Section 3 outlines the definition of the schema and how NLPs techniques can be used in conjunction with KGs. The results produced for a specific case study are shown and discussed in Section 4 followed by the conclusion of this paper in Section 5.

2 Background and Previous Work

2.1 Satellite Scheduling Problem

A satellite schedule was previously created using Google OR-Tools Constraint Programming (CP) solver [31]. The problem has been defined with three actions to be scheduled at any time: image taking, image processing, and data downlinking while idle when no other actions can occur. These actions were scheduled by means of a CP solver considering onboard memory and environmental conditions as constraints for the problem. This means the use of memory for action execution, must not exceed the maximum onboard memory, images can only take pictures when in sunlight and have a visibility of land, and downlinking can only be executed when there is communication access to a ground station with no overlapping of actions as there can only be 1 action scheduled for each time interval.

2.2 Knowledge Graph Construction and Tools

To the author's knowledge, there are currently limited research findings exploring the use of KGs for satellite scheduling. As a result, this background will cover the application of KGs for other scheduling problems and consider how these principles can be applied to satellite scheduling.

As outlined in the introduction, KGs are graph-based models for representing connected data. The graphs are typically constructed using three types of elements: Entity, Relation and Attribute - These are connected by associations called edges that define how the nodes are joined (e.g. ownership, usage, access, etc). There are no functional limitations on how many nodes of any type may exist, nor on the number of connections they may have, so are well-suited to display the relational information of data sets and can scale to accommodate very large numbers of data points [32]. These principles are followed in this paper for the construction of KGs.

Though KGs have only grown in popularity relatively recently, several tools facilitating the creation of KGs are available to researchers and developers. Considering the comparative analysis conducted in recent studies, the requirements of a KG construction tool in the paper as are follows:

- Compatible with Python
- High scalability
- Built-in visualisation engine
- Easy to adopt and use
- Ideally Open-Source

Vaticle TypeDB tool, has been selected as the preferred choice as it meets all of the requirements, including being open-source [33, 34].

2.3 Knowledge Graphs for Scheduling

A recent survey was conducted to review the domain-specific uses for KGs. The domains assessed were broad, including education, healthcare, cyber-security, various engineering disciplines, and finance - however, the survey found that only the education sector was currently making use of KGs, by assisting with resource planning and allocation [35]. While using KGs to help improve scheduling may not be considered the key use-case in some domains, the author believes this study is a symptom of the lack of published research exploring the potential for KGs in scheduling problems. In spite of this, there have been several recent examples of how KGs can solve scheduling issues as examined below.

As much of the digital world has become procedural, the ability to automatically maintain and deliver has become not just valuable but necessary. A recent study has shown the capability that KGs can enable, by creating a system that autonomously manages the scheduling of tasks for a computing power network based on KGs presenting the architecture of the network. Their self-coded method completed the activities of knowledge extraction, representation and graph construction, building a graph around the identified entities with associated relationships and attributes. Their approach was also theorised to become increasingly efficient and effective through reinforcement learning, which will continue to adapt as the system goes and is exposed to more scenarios [36]. This principle aligns with the critical issue that satellites have in managing the limited power and memory on-board and could assist with the efficient management of resources, improving the effectiveness and lifetime of the satellite.

One of the barriers organisations and industries have to adopting AI systems is the disparate and unstructured nature of data. A study on train maintenance explored this very issue, which they overcame by utilising a knowledge collection pipeline that then presented the information in a KG. Though the scheduling of maintenance would still need to be performed manually, the KG allowed access to previously obscured data enabling more informed decisions and reducing the time for detecting areas of concern [37]. Unstructured or legacy data is unlikely to be a problem on-board a satellite, however, the ground station operators may be working with aged or sub-optimal systems that could greatly benefit from a clearer understanding of the data on the ground to better inform their decisions prior to, and during, a satellites mission.

In the situation where an organisation has many large data sets stored in disassociated environments, an investigation was performed to prove how KGs can be used to combine these sources into a single, referable presentation. Considering the problem of knowledge being held across multiple sources the researchers developed an end-to-end process for constructing KGs, as shown in Figure 1, which focused on the primary principles of KGs as well as completing disambiguation and merging of similar or identical data. A Neural Network (NN) was implemented, due to the high-performance capability and to handle the massive amounts of data collected, aiding the development of the KG, which resulted in the ability to create and manage KGs in real-time. This minimises any delay in responding to new data being added or updated across the environment, allowing operational teams to make informed decisions instantaneously [38]. The capability to gather useful information into a KG in real-time from a wide variety of sources has considerable potential for adoption in satellite scheduling, as events such as debris detection or instrument failure require immediate responses - however, it must be noted that the computing resource required to run a NN may exceed on-board capacity.

2.4 Language Modelling

NLP has become a critical part of developing autonomous systems to allow for interaction between the system and users. Language Models (LMs) have been developed to utilise NLP technology and facilitate more human-like responses to language-based responses, which has been expanded further with the introduction of LLMs that have been trained on an even greater variety of words and phrases leading to evermore capable responses and reasoning capabilities. This improved capability enables increasingly complex interactions that can be highly technical and evidence-based. One of the emergent leaders in LLMs is OpenAI's ChatGPT, which has proven to be highly proficient at knowledge-intensive interactions [39]. Because of these capabilities, ChatGPT (specifically GPT3.5) will be used in this paper for the generation of explanatory responses to questions asked over the KG system.

3 Method

Utilizing the satellite scheduling problem stated in section 2.1, in combination with Vaticle TypeDB and GPT3.5, the chosen LLM, this section explains how the schema for the KG was defined. This is followed by the presentation of how the graph database can be populated with the data, as well as how the data can be accessed and retrieved from the KG. Lastly, how GPT3.5 can be used to retrieve data from the Knowledge Graph, interpret user queries, and generate corresponding explanations, is presented. Figure 2 represents an overview of the approach taken.

3.1 Defining the Schema

The schema for the KG was defined using the actions and constraints from the satellite scheduling problem defined in Section 2, as entities, combined with their respective attributes and relations. The list below is a representation of how the entities were defined, together with their attributes:

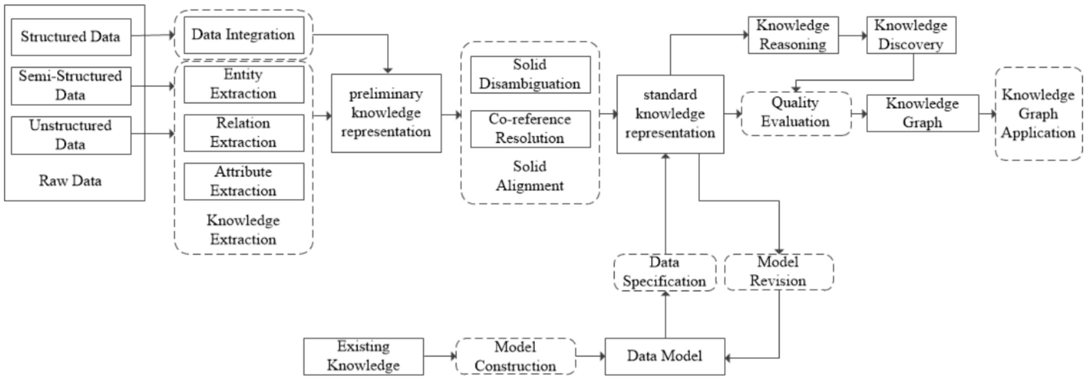

Figure 1: End-to-end process for constructing a KG including the iterative steps of knowledge extraction [38].

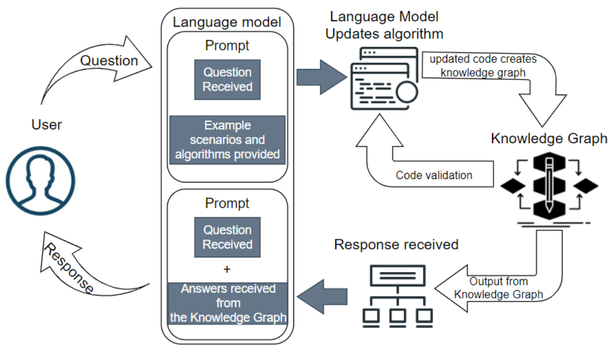

Figure 2: The use of NLP with the generation of KGs and extraction of data.

- Image - represents the '*Image taking*' action, contains '*im_size*' which represents the size of the image; '*im_in_memory*' contains the number of images in memory that have not been downlinked; and '*im_total_taken*' that represents the total images taken up to time *t*.
- Processed - contains '*pr_size*' that represents how much memory is used for each processed action; '*pr_in_memory*' representing how many processed images are in memory at the specified time *t*; and '*total_processed*' is the total processed images in memory up to that point in time *t*.
- Downlink - contains '*dl_size*' representing the memory released following each executed action; and '*total_sent*' that means the total images transmitted up to that point in time *t*.
- Idle - created when no other action occurs, containing '*total_idle*' which represents the total idle instances up to time *t*.

Each of these actions (entities) has the '*contents*' relation that links to the entity '*memory_unit*'.

- Memory_unit - contains two attributes '*maximum_capacity*' which has a fixed value based on the definition of the constraints and '*current_capacity*' that represents the current memory occupied at time *t*.

Additional defined entities are

- Action - Represents the action scheduled and the time of execution. This contains 3 attributes; '*name*' of the action, '*memory_requirements*' that aligns with the memory consumed (or released) by the executed action, and '*a_timestamp*' to represent when the actions are scheduled to be executed.
- Satellite - The main entity, containing an ID number that is traceable with respect to time, is connected to all the other entities through their respective relations:
  - installation - A relation connected to the '*memory_unit*' and '*Satellite*'.
  - schedule - A relation connected to both the '*action*' and '*Satellite*' entities, additionally has it's own '*start*' and '*end*' attributes.
  - localisation - a relation connected to both the '*environment*' and '*Satellite*' entities, also linked to the '*start*' and '*end*' attributes stated for '*schedule*'.
- Environment - this entity, as previously stated, is connected to the '*localization*' relation as well as the '*station_access*' relation. It also contains the '*land_visibility*' status attribute,'*daylight*' status attribute, and the '*latitude*', and '*longitude*' attributes.

- Ground_station - this entity is required for determining if there is '*access*' to the ground_station while is also linked to '*ID*', '*latitude*', and '*longitude*' attributes and also connected to the '*station_access*' relation with '*environment*'.

Figure 3, contains a representation of the KG schema in TypeDB studio that was created for the satellite scheduling problem with the entities shown as pink squares, attributes as blue ovals, and relations as orange diamonds.

Following the definition of the schema, the satellite data, representing a schedule instance for the EO problem solved in Section 2.1, were migrated into the database to complete the creation of the knowledge base used for information retrieval to generate explanations.

3.2 Prompting the Large Language Model

LLMs have a duplex application in the methods here proposed as described in Figure 2

- Interpret user queries and retrieve information from the KG: LLMs are used to automatically translate the user query into the graph query language
- Generate response and explanation: LLMs are used to generate a summary of the response with corresponding explanation from the relevant information extracted from the KG and the corresponding user query

GPT 3.5-turbo-16k was the chosen LLM, to allow for sufficent token generation, and the few-shot learning principle was chosen as the prompting strategy for both applications. Connecting with OpenAI, the hyper-parameters used for the LLM are reported in Table 1. These parameters were used when prompting the model in both applications.

Table 1: hyper-parameters used for GPT 3.5

hyper-parameters	Values
temperature	1.5
max_tokens	13000
top	1
frequency_penalty	0
presence_penalty	0

3.2.1 User Query Translation

To ensure the user query is understood, prompts were used to guide the LLM to generate an appropriate algorithm for the problem. The prompt used is as follows:

- *This is a template code using TypeDB to extract and get all data from a dataset created for a schedule to create a knowledge graph.*
- *If there is a question asking for an explanation, only create an algorithm for the Knowledge Graph.*
- *This template is fixed and only the variable definition and the 'get' statement can be amended but must maintain the same variable labels.*
- *All variables pertaining to the questions should be defined at the beginning, after the "match" clause:*
  - *For example. What would happen if time was 26 seconds? Time would be written like this 2020-12-03T00:00:26;*
  - *Using the same year-month-day and time format.*
  - *Note: $st should only be used if the question specifies it, otherwise it should be removed. Additionally, if there are any questions with a range of variables (you should use the start time ($st)), the same variable should be used and written like this: $st >= a; $st <= b;*
- *Also if there is a question stating how many times is it possible for 'a', the "count" statement should be written after the "get" statement for example, how many times is ground station accessible?*
  - *$g isa ground_station, has access $ac; get $ac; count;*
- *Additionally when asked about MB, multiply by 1000000*
- *Create a knowledge graph based on the question and the code template provided.*

In addition to the prompts stated, example algorithms are provided as shown in Table 2. Following this, the KGs developed are retrieved.

3.2.2 Response and Explanation generation

Following the retrieval of the data from the KGs created, the constraints were provided as prompts in tandem with the output.

- *This is a problem for a satellite schedule. That has several constraints for executing actions on a satellite.*
- *There are 4 actions:*
  - *1. Image taking, can only occur when there is enough available memory onboard, when land is visible, and when there is sunlight exposure.*

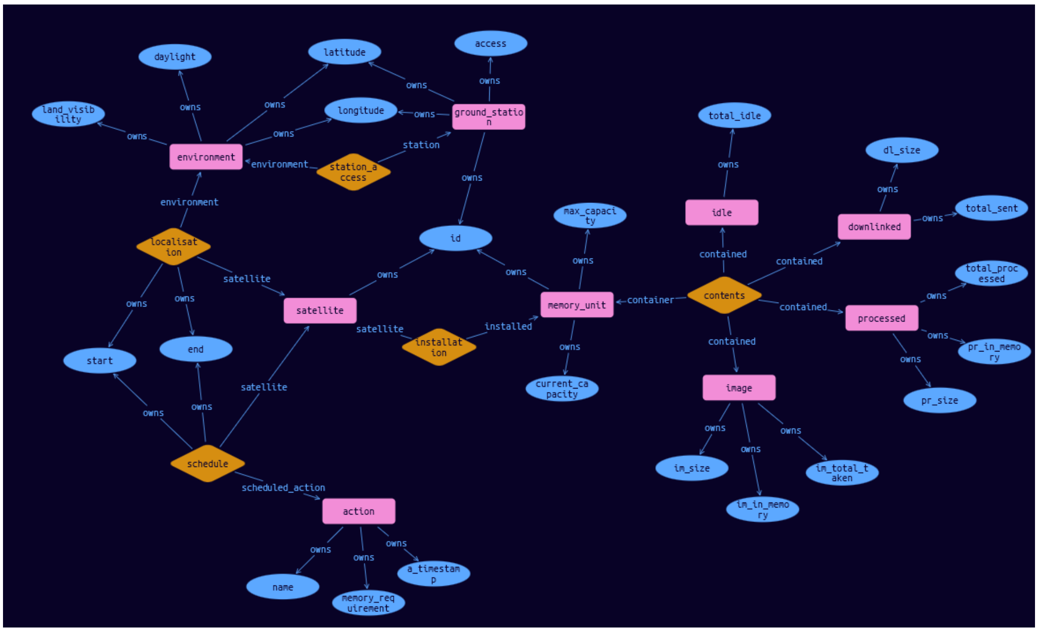

Figure 3: KG representing an overview of the satellite scheduling problem in TypeDB Studio.

- *2. Processing, can only happen when there is enough available memory and when there are unprocessed images in memory.*
- *3. Downlinking can only happen when there is access to a ground station and there is at least 1 processed image in memory.*
- *4. Idle, only occurs when no other action can take place.*

- *Only 1 action can be executed at each start time.*
- *Each action only lasts for 1 time interval.*
- *Maximum memory cannot be exceeded at any point in time which is 1920000.*

After this prompt, the query is provided followed by a list of responses retrieved from the generated KG to create an explanation to the user.

4 Results and Discussion

Once the generation of the KG database using the schema and data migration is completed, the verification and validation of the KGs were executed using three questions aligned with the ones stated in section 1. Namely:

- What actions are possible between time 41 and 56 and why?
- what actions are scheduled when current capacity is above 1.5805 Mb within times 16 seconds and 36 seconds and explain why?
- During time 51, is downlinking possible? If not, I need an explanation.

Following these questions, the LLM, with prompts, was used to guide the generation of the necessary database queries, retrieve the response, and generate the corresponding explanation, as presented in Section 3.

4.1 Queries Translation

Using the 3 queries of interest, Table 3 reports the results of the LLM when prompted as explained in Section 3.2. The tables include the natural language queries and the executable graph queries for TypeDB generated by the LLM. Due to the closed loop cycle of code regeneration, the proposed attempt with few shot learning using only 2 coded examples, required 4-8 attempts before creating an executable code based on the query. This may be improved by altering the hyper-parameters and including more prompts and example questions with their respective code. For example the generated algorithm for question 1 retrieves data containing the name of the action *'$n'*, the action execution time stamp *'$at'*, the scheduled action for retrieving memory *'$ret'*, ground station access *'$ac'*, daylight conditions *'$d'*, the schedule relation *'$t'* (to help visualize the connected components in the graph, the installation relation *'$l'*, the contents relation *'$pt'*, the memory's current capacity *'$cc'*, and land visibility *'$lv'*. This means the LLM deduced the attributes and relations required to generate a graph that can be represented in TypeDB studio. It should be

Table 2: Few-shot data sample used for GPT 3.5

Sample Questions	Sample Answers
What would happen if time was 26 seconds?	match $st = 2020-12-03T00:00:26; $a isa action, has name $n, has a_timestamp $at; $ret isa $ret-type; {$n contains 'processed'; $x isa processed, has $ret; $ret-type type pr_in_memory;} or {$n contains 'image'; $x isa image, has $ret; $ret-type type im_in_memory; } or {$n contains 'downlink'; $x isa downlinked, has $ret; $ret-type type total_sent;}; $pt($x,$mem) isa contents; $g isa ground_station, has access $ac, has $id; $env isa environment, has land_visibility $lv, has daylight $d, has latitude $lat, has longitude $lon; $sat isa satellite; $mem isa memory_unit, has current_capacity $cc, has $id, has max_capacity $max; $t($a,$sat) isa schedule; $l($mem,$sat) isa installation; $sa($g,$env) isa station_access; $loc($sat,$env) isa localisation, has start $st, has end $en; get $n, $at, $ret, $pt, $ac, $lv, $d, $lat, $lon, $l, $sa, $t, $loc, $id, $mem, $g, $cc, $max, $st, $en;
What would happen if ID was 7?	match $id = 7; $a isa action, has name $n, has a_timestamp $at; $ret isa $ret-type; {$n contains 'processed'; $x isa processed, has $ret; $ret-type type pr_in_memory;} or {$n contains 'image'; $x isa image, has $ret; $ret-type type im_in_memory; } or {$n contains 'downlink'; $x isa downlinked, has $ret; $ret-type type total_sent;}; $pt($x,$mem) isa contents; $g isa ground_station, has access $ac, has $id; $env isa environment, has land_visibility $lv, has daylight $d, has latitude $lat, has longitude $lon; $sat isa satellite; $mem isa memory_unit, has current_capacity $cc, has $id, has max_capacity $max; $t($a,$sat) isa schedule; $l($mem,$sat) isa installation; $sa($g,$env) isa station_access; $loc($sat,$env) isa localisation, has start $st, has end $en; get $n, $at, $ret, $pt, $ac, $lv, $d, $lat, $lon, $l, $sa, $t, $loc, $id, $mem, $g, $cc, $max, $st, $en;

noted that the coordinates were not extracted.

Similarly, looking at the generated algorithm for question 2, the LLM again understood the time range to select between, as well as accurately converted the units with the memory's capacity to be above 1.5805MB. This implies the prompts were understood, including the request to retrieve the names of the actions, the respective relations and entities. Likewise, for question 3, the algorithm was successfully created, while calling the relevant fields to generate the data.

On the success of a code generation, the results of the graph queries are then stored to be used in the next stage of explanation generation.

4.2 KG data visualisation

In the following figures, the reported data stored in the KG for the three queries were retrieved. Namely, Figure 4 provides a view of the data for the first query, showing all the connected components in the KG, for the time range 41-56 seconds. In this case, only the *processed* action is scheduled as the time interval is every 5 seconds based on the definition of the satellite scheduling problem in Section 2.1.

Figure 5 shows the KG generated representing the actions scheduled and additional factors such as how many processed images are in memory, the day light conditions, the scheduled times for action execution and the respective *contents* relation when the onboard memory status is greater than 1.58Mb. as a representation of the information to be retrieved by the second query.

Lastly, Figure 6, is a visualization of the satellite schedule data stored in the KG when time is 51 seconds, with the environmental conditions required for the inquired action. In this case, as stated in Section 2.1, downlinking requires ground station access but based on the question as to 'Why', the other factors, such as day and land visibility, were requested by the LLM to

Table 3: Queries and generated code

	Question	Generated code

1	What actions are possible between time 41 and 56 and why?	match $st >= 2020-12-03T00:00:41; # Start time is 41 seconds
		$st <= 2020-12-03T00:00:56; # End time is 56 seconds
		$a isa action, has name $n, has a_timestamp $at;
		$ret isa $ret-type; {$n contains 'processed';
		$x isa processed, has $ret; $ret-type
		type pr_in_memory;} or {$n contains 'image';
		$x isa image, has $ret; $ret-type type im_in_memory; } or
		{$n contains 'downlink'; $x isa downlinked, has $ret; $ret-type
		type total_sent;}; $pt($x,$mem) isa contents;
		$g isa ground_station, has access $ac, has $id;
		$env isa environment, has land_visibility $lv,
		has daylight $d; $sat isa
		satellite; $mem isa memory_unit, has current_capacity $cc,
		has $id, has max_capacity $max; $t($a,$sat) isa schedule;
		$l($mem,$sat) isa installation; $sa($g,$env) isa station_access;
		$loc($sat,$env) isa localisation, has start $st, has end $en;
		get $n, $at, $ret, $ac, $d, $t, $l, $pt, $cc, $lv;

2	What actions are scheduled when current capacity is above 1.5805Mb within times 16 seconds and 36 seconds and explain why?	match $mem isa memory_unit, has current_capacity $cc; $a isa action, has name$n, has a_timestamp $at; $ret isa $ret-type; {$n contains 'processed'; $x isa processed, has $ret; $ret-type type pr_in_memory;} or {$n contains 'image'; $x isa image, has $ret; $ret-type type im_in_memory; } or {$n contains 'downlink'; $x isa downlinked, has $ret; $ret-type type total_sent;}; $pt($x, $mem) isa contents; $g isa ground_station, has access $ac; $env isa environment, has daylight $d; $t($a, $sat) isa schedule; $l($mem, $sat) isa installation; $sa($g, $env) isa station_access; $cc > 1580500; $at >= 2020-12-03T00:00:16; $at <= 2020-12-03T00:00:36; get $n, $at, $ret, $pt, $ac, $d, $cc, $a;

3	During time 51, is downlinking possible. If not, I need an explanation.	match $st = 2020-12-03T00:00:51;
		$a isa action, has name $n, has a_timestamp $at;
		$ret isa $ret-type;
		{$n contains 'processed'; $x isa processed, has $ret;
		$ret-type type pr_in_memory;} or
		{$n contains 'image'; $x isa image, has $ret; $ret-type
		type im_in_memory; } or
		{$n contains 'downlink'; $x isa downlinked,
		has $ret; $ret-type type total_sent;};
		$pt($x,$mem) isa contents;
		$g isa ground_station, has access $ac, has $id;
		$env isa environment, has land_visibility $lv, has daylight $d,
		has latitude $lat, has longitude $lon; $sat isa satellite;
		$mem isa memory_unit, has current_capacity $cc, has $id,
		has max_capacity $max; $t($a,$sat) isa schedule;
		$l($mem,$sat) isa installation; $sa($g,$env) isa station_access;
		$loc($sat,$env) isa localisation, has start $st,
		has end $en; get $n, $ac, $lv, $d, $loc, $id, $g;

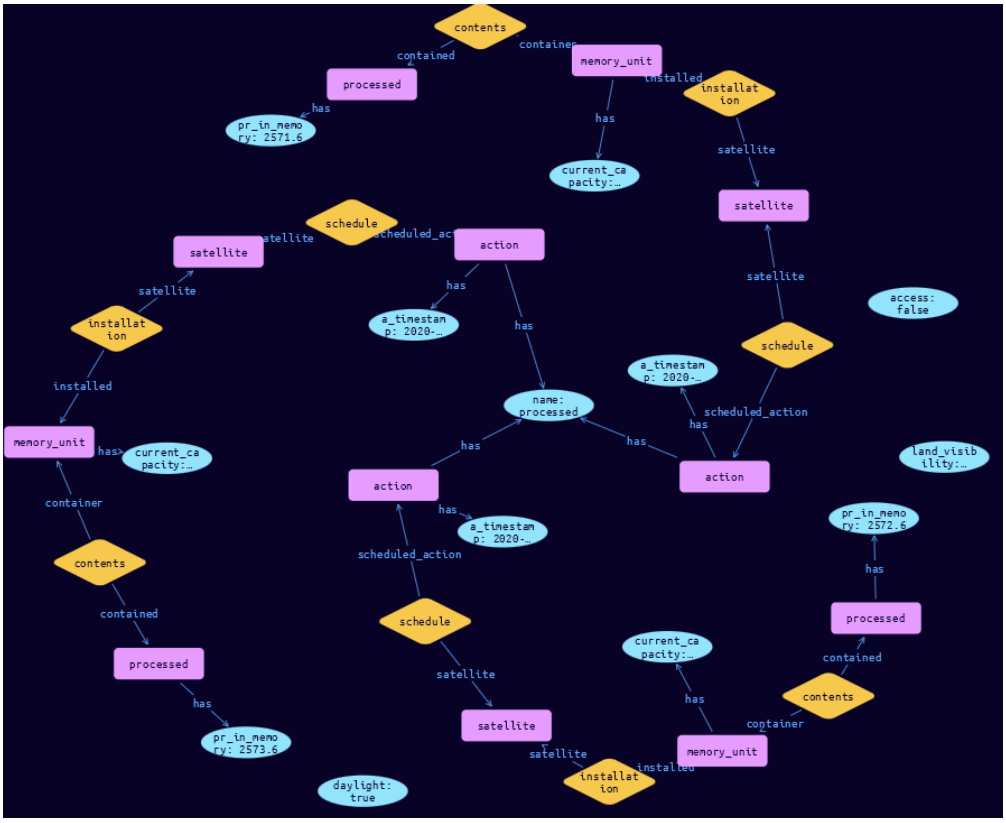

Figure 4: KG results representing the data between times 41 and 56 seconds created by the LLM from Question 1.

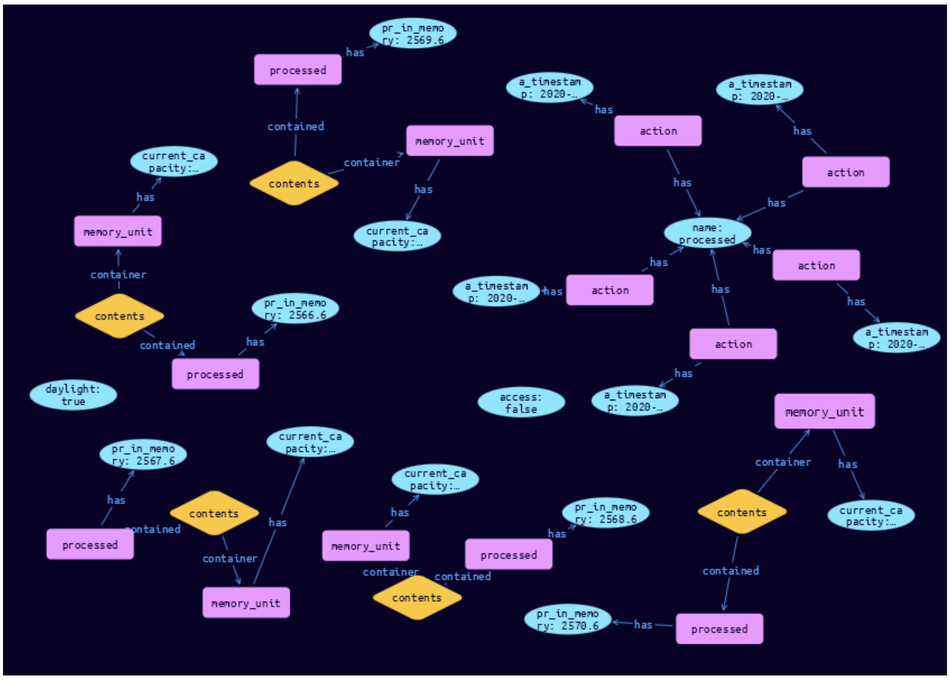

Figure 5: KG results representing the response for question 2 showing the connected fields where the current capacity is above 1.5805MB within 16 and 36 seconds.

represent the information necessary to answer the third query.

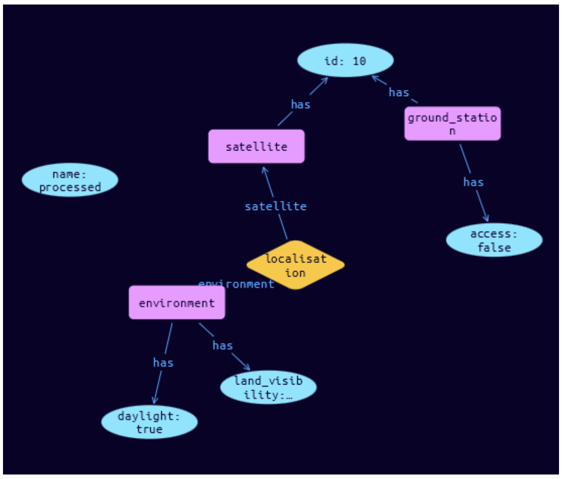

Figure 6: KG results representing data during time 51 seconds.

4.3 Response and Explanation

Following the code execution, the results from TypeDB were prompted to the LLM in combination with the corresponding question to create a response and explanation. The results are shown in Table 4, 5, and 6 respectively. The response to Question 1 in Table 4, initially mentions what possible actions can be scheduled, and it correctly explains that there are processed images in memory that would be ready for downlinking. Furthermore, on the condition that there is enough memory available onboard, the *processing* action may proceed. On the other hand, idle is possible in the case processing doesn't occur and may remain in idle mode until other actions are possible to be executed. The LLM additionally talks about the requirement for downlinking the processed images in memory, however, is not possible as there is no access to the ground station within the requested times.

It can be noted that '*Image_taking*' was not stated, meaning there is potential to improve the model since no example explanations were prompted to the LLM. Therefore, if examples were provided and test cases, could improve the overall performance.

The explanation provided for question 2 in Table 5, accurately stated what actions were scheduled, in this case, '*Processing*'. With the correct explanations, including the memory calculations, memory availability, and ground station accessibility to check for downlinking potential, the LLM used the constraints from the prompts provided and data to explain the satellite's scenario for the 3 different time intervals.

Table 6 represents the explanations in response to question 3. These results provide a direct and succinct explanation reiterating the time from the query to further explain that the ground station isn't accessible, with an implication that it may not be compatible or isn't within range and therefore not possible to downlink to the station. The LLM also stated the satellite may not have an established communication link and thus the processed data cannot be transmitted.

Table 4: Response and explanation provided to the User from the LLM Question 1

	Question	output to user
1	What actions are possible between time 41 and 56 and why?	During the time interval between 41 and 56, there are three possible actions that can occur: processed, idle, and downlinking. 1. Processed: According to the given data, there are three processed images in memory at different timestamps: 2020-12-03 00:00:41, 2020-12-03 00:00:51, and 2020-12-03 00:00:46. This means that the satellite has processed images ready to be downlinked. As long as there is enough available memory, the satellite can continue performing the processing action during this time interval. 2. Idle: The idle action is considered when no other action is possible based on the given constraints. In this case, if there are no processed images in memory and there is spare memory available, the satellite can remain in the idle state during this time interval until new actions become possible. 3. Downlinking: From the given data, access to a ground station is not available at any time between 41 and 56. Therefore, downlinking action is not possible during this time interval. However, if there had been access to a ground station and at least one processed image in memory, the satellite could have performed the downlinking action. Therefore, between time 41 and 56, the possible actions are processed and idle, but downlinking is not possible due to the lack of ground station access.

5 Conclusion and Future Work

The aim of the paper was to explore how KGs could be useful in XAI for satellite scheduling, while detailing how KGs store information, how that information can be accessed by a natural language interface, and what can be inferred based on the response. The component parts of the satellite scheduling problem data were outlined using the principles of KG construction, followed by the formulation of a few-shot learning prompting approach to the chosen LLM (GPT3.5) which primed the model to reply with an algorithm that could be executed for generating the KG. In the results, it was discovered that the LLM was very capable of generating correct responses that would produce KGs displaying relevant and related explanations, based on prompting and the LLM interpretation, although this did often take multiple attempts to produce a fully accurate response leaves room for improvements. It is also worth noting that separate prompts for code generation and subsequent explanations was required, which could potentially be improved through future work. The overall outcome has proved the capability of KGs to present both simple and complex graphs for satellite scheduling, with the ability to answer user queries effectively. The KGs produced are sufficiently flexible to show a small number of fields as well as hundreds of connected fields, or even single-field responses. This takes into account the complexity of the query and allows users to expand or narrow queries quickly, for targeted insight that will enable informed decisions in updating the schedule. Combined with the explanations generated by the LLM, both graphical and textual knowledge is made available to users providing a thorough understanding of how and why the satellite schedule was generated this particular way.

5.1 Future work

This area of research would greatly benefit from further investigation with a key area for exploration being expanding the dataset, increasing the data complexity, and/or introducing unstructured data to test how KGs and LMs can bridge or overcome any gaps for users. The use Argumentation Framework (AF)s should also be explored as the KGs have the potential to present alternative feasible options for schedule alterations based on the results from the AF. Additionally, utilizing a NN to facilitate the autonomous reading and interpretation of KGs, could allow experimenting with automated reasoning and response in scheduling tasks. Finally, investigation into the capability of a multigraph comparative analysis system to visualize decision changes in areas of interest prior to execution to further enhance the information available to users.

Table 5: Response and explanation provided to the User from the LLM Question 2

	Question	output to user
2	What actions are scheduled when current capacity is above 1.5805 Mb within times 16 seconds and 36 seconds and explain why?	Based on the information provided, the actions that are scheduled when the current capacity is above 1.5805 MB within times 16 seconds and 36 seconds are "processed" actions. Calculating the difference in current_capacity between the timestamps 2020-12-03 00:00:16 and 2020-12-03 00:00:36, we find the increase in capacity: 1581546 - 1580546 = 1000 bytes Since the maximum memory capacity is not exceeded at any point in time (1920000 bytes), and the available memory increases by 1000 bytes within this time interval, it indicates that there is enough available memory to schedule a "processed" action. Based on the timestamps and other data, we can see the following details for the "processed" actions within the specified time range: - Action 1: - Timestamp: 2020-12-03 00:00:16 - Available Processed Images in Memory: 2566.6 - Relevant Content for Image Processing: - No Access to Ground Station (False) - Daylight Availability (True) - Action 2: - Timestamp: 2020-12-03 00:00:31 - Available Processed Images in Memory: 2569.6 - Relevant Content for Image Processing: - No Access to Ground Station (False) - Daylight Availability (True) - Action 3: - Timestamp: 2020-12-03 00:00:36 - Available Processed Images in Memory: 2570.6 - Relevant Content for Image Processing: - No Access to Ground Station (False) - Daylight Availability (True) From the provided information, it appears that the "processed" actions are scheduled at the available timestamps within the specified time range because the necessary conditions for image processing are satisfied: there is enough available memory, no access to the ground station (indicating that downlinking is not possible), and daylight availability. It's important to note that although the passage of time indicates a gradual increase in available processed images in memory, the specific constraints mentioned (sufficient memory, no access to the ground station, and daylight availability) determine the scheduling of "processed" actions.

Table 6: Response and explanation provided to the User from the LLM Question 3

	Question	output to user
3	During time 51, is downlinking possible. If not, I need an explanation.	Based on the given data, downlinking is not possible during time 51. The access data is set to False, indicating that there is no access to a ground station. This means that the satellite cannot establish a communication link to transmit the data. Without access to a ground station, the processed data cannot be downlinked. It is also mentioned that the ground station entity has an ID of 10, implying that the only available ground station may be incompatible or out of range for downlinking at this particular time interval. As a result, downlinking cannot occur during time 51 moments.

Acknowledgements

This study was half-funded by ESA under the OSIP Co-Sponsored PhD activity: "Robust and Explainable Mission Planning and Scheduling (REMPS)" No. 4000132894/20/NL/MH/hm. The authors would also like to acknowledge the support of ESA through the Research Fellow program. Finally, they wish to warmly thank the team from Vaticle TypeDB for the support and advice in creating the Knowledge Graph used in this paper.

References

[1] S. S. Bhat, R. M. Narayanan, and M. Rangaswamy, "Bandwidth sharing and scan scheduling in multimodal radar with communications and tracking," *IETE Journal of Research*, vol. 59, no. 5, pp. 551–562, 2013. [Online]. Available: https://www.tandfonline.com/doi/abs/10.4103/0377-2063.123761

[2] C. A. Rigo, L. O. Seman, E. Camponogara, E. Morsch Filho, and E. A. Bezerra, "Task scheduling for optimal power management and quality-of-service assurance in cubesats," *Acta Astronautica*, vol. 179, pp. 550–560, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094576520306858

[3] J. Zhang and L. Xing, "An improved genetic algorithm for the integrated satellite imaging and data transmission scheduling problem," *Computers Operations Research*, vol. 139, p. 105626, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054821003397

[4] F. Xhafa and A. W. Ip, "Optimisation problems and resolution methods in satellite scheduling and space-craft operation: a survey," *Enterprise Information Systems*, vol. 15, no. 8, pp. 1022–1045, 2021. [Online]. Available: https://doi.org/10.1080/17517575.2019.1593508

[5] B. A. Homssi, K. Dakic, K. Wang, T. Alpcan, B. Allen, S. Kandeepan, A. Al-Hourani, and W. Saad, "Artificial intelligence techniques for next-generation mega satellite networks," *arXiv preprint arXiv:2207.00414*, 2022.

[6] J. Bermyn and C. Dorn, "Proba spacecraft family: small mission solutions for emerging applications," in *Small Satellites for Earth Observation*. Springer, 2008, pp. 67–76.

[7] M. Tipaldi, L. Feruglio, P. Denis, and G. D'Angelo, "On applying ai-driven flight data analysis for operational spacecraft model-based diagnostics," *Annual Reviews in Control*, vol. 49, pp. 197–211, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1367578820300213

[8] F. Xhafa, J. Sun, A. Barolli, A. Biberaj, and L. Barolli, "Genetic algorithms for satellite scheduling problems," *Mobile Information Systems*, vol. 8, no. 4, pp. 351–377, 2012.

[9] D. Zhang, L. Guo, B. Cai, N. Sun, and Q. Wang, "A hybrid discrete particle swarm optimization for satellite scheduling problem," in *IEEE Conference Anthology*, 2013, pp. 1–5.

[10] Y. Song, L. Wei, Q. Yang, J. Wu, L. Xing, and Y. Chen, "Rl-ga: A reinforcement learning-based genetic algorithm for electromagnetic detection satellite scheduling problem," *Swarm and Evolutionary Computation*, vol. 77, p. 101236, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S221065022300010X

[11] H. W. Loh, C. P. Ooi, S. Seoni, P. D. Barua, F. Molinari, and U. R. Acharya, "Application of explainable artificial intelligence for healthcare: A systematic review of the last decade (2011–2022)," *Computer Methods and Programs in Biomedicine*, vol. 226, p. 107161, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169260722005429

[12] P. Weber, K. V. Carl, and O. Hinz, "Applications of explainable artificial intelligence in finance—a systematic review of finance, information systems, and computer science literature," *Management Review Quarterly*, pp. 1–41, 2023.

[13] H. Mankodiya, D. Jadav, R. Gupta, S. Tanwar, W.-C. Hong, and R. Sharma, "Od-xai: Explainable ai-based semantic object detection for autonomous vehicles," *Applied Sciences*, vol. 12, no. 11, p. 5310, 2022.

[14] M. Ashoori and J. D. Weisz, "In ai we trust? factors that influence trustworthiness of ai-infused decision-making processes," *arXiv preprint arXiv:1912.02675*, 2019.

[15] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models," *arXiv preprint arXiv:1708.08296*, 2017.

[16] J. Dieber and S. Kirrane, "Why model why? assessing the strengths and limitations of lime," *arXiv preprint arXiv:2012.00093*, 2020.

[17] A. Holzinger, A. Saranti, C. Molnar, P. Biecek, and W. Samek, "Explainable ai methods-a brief overview," in *International Workshop on Extending Explainable AI Beyond Deep Models and Classifiers*. Springer, 2020, pp. 13–38.

[18] H. Heberle, L. Zhao, S. Schmidt, T. Wolf, and J. Heinrich, "Xsmiles: interactive visualization for molecules, smiles and xai attribution scores," *Journal of Cheminformatics*, vol. 15, no. 1, pp. 1–12, 2023.

[19] J.-P. Poli, W. Ouerdane, and R. Pierrard, "Generation of textual explanations in xai: the case of semantic annotation," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2021, pp. 1–6.

[20] J. M. Rožanec, P. Zajec, K. Kenda, I. Novalija, B. Fortuna, and D. Mladenić, "Xai-kg: Knowledge graph to support xai and decision-making in manufacturing," in *International Conference on Advanced Information Systems Engineering*. Springer, 2021, pp. 167–172.

[21] A. Hogan, E. Blomqvist, M. Cochez, C. D'amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A.-C. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann, "Knowledge graphs," *ACM Comput. Surv.*, vol. 54, no. 4, jul 2021. [Online]. Available: https://doi.org/10.1145/3447772

[22] C. Peng, F. Xia, M. Naseriparsa, and F. Osborne, "Knowledge graphs: Opportunities and challenges," *Artificial Intelligence Review*, pp. 1–32, 2023.

[23] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 2, pp. 494–514, 2021.

[24] E. Rajabi and S. Kafaie, "Knowledge graphs and explainable ai in healthcare," *Information*, vol. 13, no. 10, 2022. [Online]. Available: https://www.mdpi.com/2078-2489/13/10/459

[25] A. Fensel, Z. Akbar, E. Kärle, C. Blank, P. Pixner, and A. Gruber, "Knowledge graphs for online marketing and sales of touristic services," *Information*, vol. 11, no. 5, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/5/253

[26] H. Ko, P. Witherell, Y. Lu, S. Kim, and D. W. Rosen, "Machine learning and knowledge graph based design rule construction for additive manufacturing," *Additive Manufacturing*, vol. 37, p. 101620, 2021.

[27] Q. Yan, M. Ding, J. Lu, J. Lu, H. Yang, and F. Xie, "Construction of dynamic knowledge graph for grid day-ahead scheduling optimization decision," in *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, vol. 10. IEEE, 2022, pp. 1821–1826.

[28] B. Min, H. Ross, E. Sulem, A. P. B. Veyseh, T. H. Nguyen, O. Sainz, E. Agirre, I. Heintz, and D. Roth, "Recent advances in natural language processing via large pre-trained language models: A survey," *ACM Computing Surveys*, 2021.

[29] D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. Le *et al.*, "Least-to-most prompting enables complex reasoning in large language models," *arXiv preprint arXiv:2205.10625*, 2022.

[30] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in neural information processing systems*, vol. 35, pp. 22 199–22 213, 2022.

[31] C. Powell and A. Riccardi, "Towards explainability of on-board satellite scheduling for end user interactions," Oct. 2021, 72nd International Astronautical Congress, IAC 2021 ; Conference date: 25-10-2021 Through 29-10-2021. [Online]. Available: https://iac2021.org/,https://www.iafastro.org/events/iac/iac-2021/

[32] Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, and Z. Duan, "Knowledge graph completion: A review," *Ieee Access*, vol. 8, pp. 192 435–192 456, 2020.

[33] A. Berquand, "Text mining and natural language processing for the early stages of space mission design," Ph.D. dissertation, University of Strathclyde, 2022.

[34] A. Jarabo Peñas, "Digital twin knowledge graphs for iot platforms: Towards a virtual model for real-time knowledge representation in iot platforms," 2023.

[35] B. Abu-Salih, "Domain-specific knowledge graphs: A survey," *Journal of Network and Computer Applications*, vol. 185, p. 103076, 2021.

[36] Y. Bi, Y. Long, Y. Jin, S. Zheng, H. Liu, and H. Wang, "Automatic scheduling technology of computing power network driven by knowledge graph," in *2022 International Conference on Service Science (ICSS)*. IEEE, 2022, pp. 154–160.

[37] E. Papadakis, L. McCluskey, H. Louadah, and G. Tucker, "Ontology-guided knowledge graph construction to support scheduling in a train maintenance depot," in *CEUR Workshop Proceedings*. CEUR Workshop Proceedings, 2023.

[38] J. Liu, H. Lu, Y. Yuan, H. Gu, K. Xie, D. Li, C. Luo, G. Wu, and Y. Song, "Intelligent operation and maintenance schedule system based on knowledge graph: Construction and applications," in *Proceedings of the 2023 2nd International Conference on Networks, Communications and Information Technology*, 2023, pp. 177–181.

[39] J. Yang, H. Jin, R. Tang, X. Han, Q. Feng, H. Jiang, B. Yin, and X. Hu, "Harnessing the power of llms in practice: A survey on chatgpt and beyond," *arXiv preprint arXiv:2304.13712*, 2023.