KG-Roar: Interactive Datalog-based Reasoning on Virtual Knowledge Graphs

Luigi Bellomarini
Bank of Italy
luigi.bellomarini@bancaditalia.it

Marco Benedetti
Bank of Italy
marco.benedetti@bancaditalia.it

Andrea Gentili
Bank of Italy
andrea.gentili@bancaditalia.it

Davide Magnanimi
Bank of Italy & Politecnico di Milano
davide.magnanimi@bancaditalia.it

Emanuel Sallinger
TU Wien & University of Oxford
sallinger@dbai.tuwien.ac.at

ABSTRACT

Logic-based Knowledge Graphs (KGs) are gaining momentum in academia and industry thanks to the rise of expressive and efficient languages for Knowledge Representation and Reasoning (KRR). These languages accurately express business rules, through which valuable new knowledge is derived. A versatile and scalable back-end reasoner, like Vadalog, a state-of-the-art system for logic-based KGs—based on an extension of Datalog—executes the reasoning.

In this demo, we present KG-Roar, a web-based interactive development and navigation environment for logical KGs. The system lets the user augment an input graph database with intensional definitions of new nodes and edges and turn it into a KG, via the metaphor of reasoning widgets—user-defined or off-the-shelf code snippets that capture business definitions in the Vadalog language.

Then, the user can seamlessly browse the original and the derived nodes and edges within a "Virtual Knowledge Graph", which is reasoned upon and generated interactively at runtime, thanks to the scalability and responsiveness of Vadalog. KG-Roar is domain-independent but domain aware, as exploration controls are contextually generated based on the intensional definitions.

We walk the audience through KG-Roar showcasing the construction of certain business definitions and putting it into action on a real-world financial KG, from our work with the Bank of Italy.

PVLDB Reference Format:
Luigi Bellomarini, Marco Benedetti, Andrea Gentili, Davide Magnanimi, and Emanuel Sallinger. KG-Roar: Interactive Datalog-based Reasoning on Virtual Knowledge Graphs. PVLDB, 16(12): 4014 - 4017, 2023.
doi:10.14778/3611540.3611609

PVLDB Artifact Availability:
The datasets and an accompanying video have been made available at https://bit.ly/3ZahQC5 and **https://youtu.be/Vo9eIWjc4zA**, respectively.

1 INTRODUCTION

Datalog [1], originally conceived and incubated by the database community, is experiencing growing success in the AI space [10] and in industrial applications, as witnessed by lively dedicated venues [2]. The semantic limitations, which initially reduced the application of Datalog for reasoning purposes, have been overcome in most recent fragments of the Datalog$^{\pm}$ family of languages [8], such as *Warded Datalog*$^{\pm}$ [10]. The potential undecidability and complexity blowup coming from the introduction of existential quantification and its interplay with recursion have been tamed by means of mild syntactic limitations that allow achieving a good tradeoff between expressive power and computational complexity. In particular, Warded Datalog$^{\pm}$ can express languages of the *DL-Lite* family (being therefore suited for ontological reasoning), captures full Datalog, and offers PTIME data complexity [7, 10].

We recently proposed Vadalog [4, 6], a Datalog-based reasoner that exploits the Vadalog language—an extension of the warded fragment with features of practical utility—and rolled out many industrial applications, especially in the financial realm, by modeling complex business domains as *Knowledge Graphs* (KGs) [11].

Prominent proposals [5, 9] do indeed acknowledge that reasoning plays a key role in the definition of a KG, seen as the interplay of: (i) an *extensional component*, the database of originally available facts from enterprise databases, and (ii) an *intensional component*, expressed by reasoning rules that derive new database facts [5].

Despite such a remarkable industrial and academic interest, we still witness the absence of a general-purpose tool to define, navigate, and query a Datalog-based KG. In this work, we fill this gap and present KG-Roar, a web-based development and navigation tool for KGs based on Vadalog.

Scope of the Demo. We introduce many financial business notions expressed in Vadalog. The audience will experience the "virtual flavour" of a KG that is grown dynamically by exerting such business knowledge. Logical KGs, sometimes thought of as mere academic constructs, will be dressed in the familiar-looking navigational interaction of graph databases, without giving up the countless possibilities offered by reasoning rules, which we exploit to feature a sound, domain-independent (but domain-aware), real-time augmentation of the KG. Our demonstration will involve the participants in multiple end-to-end applications of KG-Roar on real-world use cases from our work with the Central Bank of Italy. The audience will (i) play the role of a financial analyst and study the relationships between intermediaries (e.g., control, ultimate owners, generic indirect influence, and so on) searching for *unlawful or otherwise interesting financial patterns*, and (ii) simulate a *creditworthiness assessment*, by checking the absence of conflicts of interest between banks, even in ultra-large-scale settings. Finally, (iii) we will perform an *exploratory analysis* of the KG.

Proceedings of the VLDB Endowment, Vol. 16, No. 12 ISSN 2150-8097.
doi:10.14778/3611540.3611609

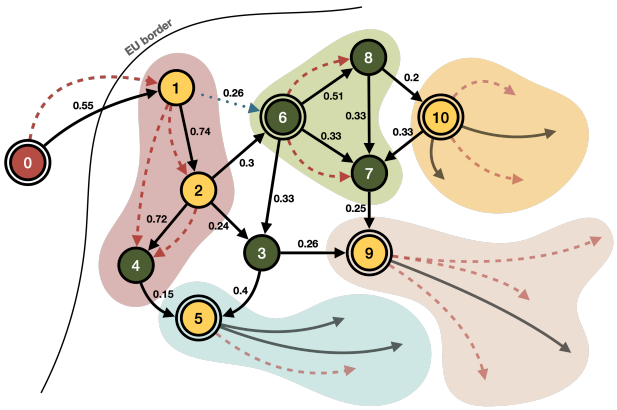

Figure 1: An excerpt of the European Ownership KG. Green nodes are intermediaries; yellow nodes are intermediaries of national relevance; the red node is a non-EU bank. Solid edges represent settled ownership relationships, with the respective share amount; dashed-red edges are control relationships. The dotted-blue edge is a candidate share acquisition. Coloured shapes determine banks and intermediaries within the same control groups. Ultimate controllers are represented as nodes with two circles.

Overview. We lay out the technical background (Section 2), Then, we describe the functionalities of KG-Roar (Section 3) and the demo organization (Section 4). An accompanying video is available.[1]

2 LOGICAL KNOWLEDGE GRAPHS IN KG-ROAR

Figure 1 depicts a small excerpt of the *European Ownership KG* managed by the Bank of Italy. Our demo of KG-Roar will be based on this KG. Nodes denote companies (i.e., banks or financial intermediaries) and individuals (not present in the figure); solid edges represent settled shareholding relationships, known as *ownerships*, with the respective share amount. Nodes and solid edges are part of the extensional component; dashed lines stand for derived elements.

In KG-Roar, we adopt a relational representation of the extensional component. Let $\mathbf{C}$, $\mathbf{N}$, and $\mathbf{V}$ be disjoint countably infinite sets of *constants*, *(labeled) nulls*, and (regular) *variables*, respectively. A *(relational) schema* $\mathbf{S}$ is a finite set of relation symbols (or predicates) with associated arity. A *term* is a constant or a variable. An *atom* over $\mathbf{S}$ is an expression of the form $R(\bar{v})$, where $R \in \mathbf{S}$ is of arity $n > 0$ and $\bar{v}$ is an n-tuple of terms. A *database* over $\mathbf{S}$ associates with each relation symbol in $\mathbf{S}$ a relation of the respective arity over $\mathbf{C}$.

- The **extensional component** of the KG of our demo is encoded as a database $\mathcal{D}$ of facts $\mathsf{Own}(x, y, w)$, where x is a shareholder (company or individual), y is a company, and w is the share percentage. For example, the graph in Figure 1 is encoded as $\mathcal{D} = \{\mathsf{own}(1, 2, 0.74), \mathsf{own}(2, 4, 0.72), \mathsf{own}(2, 3, 0.24), \mathsf{own}(4, 5, 0.15), ..\}$.

Let us now consider the intensional component of the KG at hand. We introduce some technical background first: A Datalog$^{\pm}$ program Σ is a set of *existential rules* $\forall\bar{x}\forall\bar{y}(\varphi(\bar{x}, \bar{y}) \rightarrow \exists\bar{z}\ \psi(\bar{x}, \bar{z}))$, where φ (the *body*) and ψ (the *head*) are conjunctions of atoms. We will omit quantifiers and often denote conjunction by comma.

The semantics of a Datalog$^{\pm}$ program Σ can be defined via chase-based procedures [1]: Loosely speaking, the chase incrementally expands $\mathcal{D}$ by applying the rules of Σ, possibly introducing fresh labeled nulls from $\mathbf{N}$, to satisfy existential quantification. Warded Datalog$^{\pm}$ limits the propagation of labelled nulls and controls computational complexity. Vadalog enriches the warded fragment with equality-generating dependencies [3], algebraic operations, comparisons, aggregations, and other features [10] of practical utility. They will be used in the demo as their semantics is intuitive, although it is not formally introduced in this paper for space reasons.

In our demo, we define multiple business notions as business rules, which will lead to an interactive generation of new facts from the application of Σ on $\mathcal{D}$—technically known as the *reasoning task*. This will go hand in hand with the analyst invoking specific queries or performing exploratory analysis. The first use case will put into action reasoning tasks encoding *company control*, *ultimate controllers*, *mutual influence*, and others in the corporate economics realm; the second will rely on a notion of *close links*, i.e., companies considered too "close" to back one another in lending operations.

To continue our technical introduction and cover the **intensional component**, we start by focussing on the definition of *control edges* and *control groups* in Figure 1. We then conclude the section with an example of a reasoning task.

Control edges are derived relationships with the meaning "*x controls y*" obtained from shareholding edges according to the business definition [12]: *A company x controls a company y iff (i) x directly owns a majority of the shares of y, or (ii) x controls a set of companies that, possibly together with x itself, jointly hold the majority of y.*

- The *control edges* of the intensional component of our KG are defined by the following Vadalog rule. Every company x controls itself (Rule 1). Then, by Rule 8, we have that x controls y if the sum of the shares w of y owned by companies z, over all companies z controlled by x, is above the 50% threshold.

$$\mathsf{company}(x) \rightarrow \mathsf{control}(x, x) \tag{1}$$

$$\mathsf{control}(x, z), \mathsf{own}(z, y, w) \wedge v = \mathit{sum}(w) \wedge v > 0.5 \rightarrow \mathsf{control}(x, y) \tag{2}$$

Then, further intensional definitions can be built upon control.

- For example, a *control group* is defined as follows. By Rule 3, every company y is controlled by some shareholder x (company or individual), which, in turn, is not controlled by anyone else (i.e., the *ultimate controller*). All companies x having the same ultimate controller z belong to the same control group (Rule 4). According to Rule 5 (an equality-generating dependency), we enforce that control groups for the same ultimate controller x have the same identifier, by unifying the corresponding labeled nulls.

$$\mathsf{company}(x), \mathsf{company}(y), \mathsf{company}(z), \mathsf{control}(x, y), \neg\mathsf{control}(z, x) \wedge z \neq x \rightarrow \mathsf{ultimateController}(x, y) \tag{3}$$

$$\mathsf{ultimateController}(x, y) \rightarrow \exists z\ \mathsf{controlGroup}(z, y) \tag{4}$$

$$\mathsf{controlGroup}(z, x), \mathsf{controlGroup}(w, x) \rightarrow z = w \tag{5}$$

[1] https://youtu.be/Vo9eIWjc4zA [last accessed: 28-06-2023].

A Reasoning Use Case. An analyst wants to assess the impact of the prospective acquisition of 26% of Company 6 by Bank 1. KG-Roar would raise an alert, because the acquisition would move Companies 5 and 9 (which are intermediaries of national relevance), into a non-EU control group. Reasoning steps: Company 1, controlled by the non-EU Company 0, already controls 30% of Company 6, and, with a further 26% acquisition would gain the majority. Then, Company 1 would gain control over Company 3, the strategic intermediary 5, as well as 7 and 8, already controlled by 6. Finally, via the controlled companies 3 and 7, Company 1 would take over the strategic intermediary 9 and, with a cascade effect, also other companies controlled by 5, 7 and 9. As a result, Company 0 becomes the ultimate controller of all the aforementioned companies, which would transition into non-EU control.

3 OVERVIEW OF THE SYSTEM

KG-Roar offers a productivity environment for KGs: It allows to handle a graph-based extensional component, augment it via intensional Vadalog rules, seamlessly navigate through original or derived graph items, perform complex analyses, and export results.

For the extensional component, KG-Roar handles *graph databases* as *first-class citizens*: It captures nodes and edges with a relational interface, as facts, and provides standard visualization, navigation, and query capabilities, by interacting with the back-end database in its native query language.

To handle the intensional component in a modular way, KG-Roar adopts a **microkernel architecture**: The core of business knowledge expressed by the extensional component is augmented by *loadable modules*, each representing a cohesive portion of the domain of interest, encoded in Vadalog. The loaded modules are applied as reasoning tasks (see Section 2), executed with a **lazy evaluation** strategy by a back-end instance of the Vadalog System.

KG-Roar provides ready-to-use **libraries** of modules, capturing specific areas of business knowledge, such as detection of controllers and control groups, creditworthiness evaluation, risk propagation, and so on. Many of them will be shown in the demo. Modules can be visually edited and operated by the user in the form of *widgets*. A widget is an embedded micro-IDE environment enabling the specification of a module. Following a *notebook approach* like in well-known data science platforms, a widget allows a textual/visual description of a module behavior, incorporates a productivity area to write, test, and debug the Vadalog code, a mechanism to bind the reasoning rules to the extensional component, and a set of options to customize how the derived knowledge should be rendered.

Visual Organization of KG-Roar. Two working panels of KG-Roar are shown in Figure 2(a): the left-hand one shows the micro-IDE for widget development; the right-hand one shows a rendered navigable portion of the KG.

Main Workflow. The user interaction starts with ① the execution of an *exploratory widget*, which performs a *selection reasoning task* that defines the scope of the analysis; ② the system spawns the corresponding *reasoning task*, that is executed by the Vadalog System in the back-end, and returns and renders a *view on the KG*, which comprises both extensional and freshly generated intensional elements. The user then can follow up along three paths: ③a she *visually explores the graph*, using contextual controls to activate intensional definitions to expand the rendered nodes and include nodes and edges obtained via reasoning; ③b she loads a *pre-defined widget* encoding a portion of business logic, visually binds it to the desired input and runs it to augment the graph with new reasoning results; ③c she defines a new portion of the intensional component by creating a *new widget* and continues with 3a or 3b.

The demo will also cover many non-functional characteristics of the system. KG-Roar is *domain-independent*, in the sense that the KG navigation controls are generated at runtime, depending on the underlying extensional component and, more importantly, intensional definitions. From another perspective, KG-Roar adapts the navigation based on the defined rules and how they augment the currently rendered objects. In this sense, it is also *domain-aware*. KG-Roar offers *seamless navigation*, as one can indistinctly inspect items of either the extensional or intensional component. The intensional component is *dynamically extensible*, in the sense that the user can define new nodes and edges with reasoning rules while exploring an existing KG. Moreover, KG-Roar can work in a "*data science mode*", consisting in the incremental augmentation of the rendered KG with the results of queries or reasoning programs. As far as reasoning is concerned, the system offers *runtime derivation of facts* by invoking Vadalog as a back-end reasoner. Inheriting the scalability properties of the system, it provides high performance on large data volumes, with an *interactive experience*. Compact experimental evidence related to the "creditworthiness assessment" use case (details in Section 4) is shown in Figure 2(b).

4 DEMONSTRATION PLAN

Our presentation script is designed for expert and novel users interested in working with logical KGs. After a very brief focus on the system architecture, the demonstration will start.

We will kick off the session with a tour of the working panels of KG-Roar (Figure 2), to give a high-level overview of the interactions. Then, we will play the role of a financial analyst.

Financial analysis. We will use simple *selection widgets* to single out a portion of the KG to center our analysis on. We will show the extensional component and perform some basic navigation. Then, we will use the widgets to understand and execute a module that augments the KG with *company control* edges, as we have seen with Rules 1 and 2. This will give us the occasion to see a joint use of *full recursion* and *aggregation* in ontological reasoning on KGs. Another pre-built widget, namely, *ultimate controller*, will then be introduced; it will give us the chance to present Rules 3 and 4 in action on the extensional data at hand. They will be enriched with new edges directly connecting our financial intermediaries with their ultimate controllers. In this case, we will experience the use of recursion and *negation*. Finally, one more step will conclude the expansion of the KG with our intensional component: We will enact *existential quantification* and *equality-generating dependencies* to compute and visualize control groups, i.e., groups of intermediaries sharing the same ultimate controller. With this working environment set, we will see how a supervision analyst can perform the *what-if* analysis we have exemplified in Section 2 to evaluate the impact of an acquisition. We shall see how simple acquisitions can trigger multiple changes and even move an intermediary from one control group to another. The demo will show our novel design

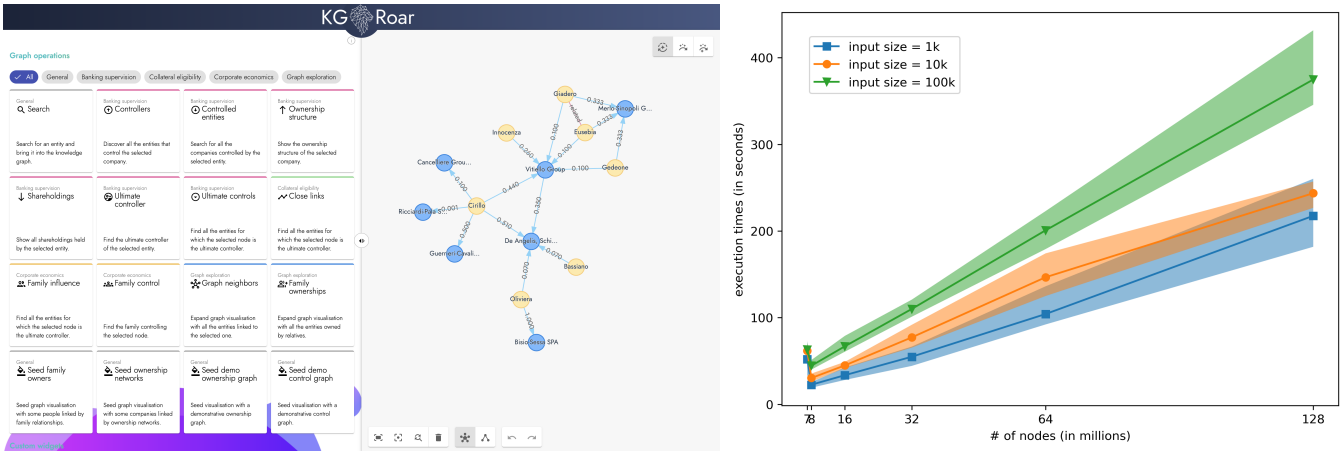

Figure 2: (a) The main working panel of KG-Roar, showing the intensional component (left) as a grid of cards, each containing a reasoning task, and the extensional component (right) as an interactive graph; (b) an experimental evaluation of the elapsed time for *company control*, with growing graph size. Real-world and synthetic datasets are used (available at https://bit.ly/3ZahQC5).

ideas adopted in KG-Roar to apply live modifications to the data, a key ingredient in the what-if analysis: We will resort to *dynamic fact injection* and *artificially negated eureka atoms*, two practical techniques to embed positive and negative deltas in the reasoning process and at runtime, in order to leverage the reasoning power of Vadalog and efficiently execute complex simulations in the back-end. We will touch on the intricacies of the related problems concerning the (re)identification of facts and the generation of labelled nulls, without being too technical, but still giving evidence of the effectiveness of the visual solutions we propose in KG-Roar. The audience will appreciate the presence of our micro-IDEs to understand and debug the widgets and enjoy the navigable working panel to inspect a graph-based representation of the KG facts. Business-oriented participants will perceive the benefit to use logic to augment the data with valuable information.

Creditworthiness assessment. Deciding whether some person or company x is worth credit is an essential process for commercial banks and financial intermediaries. On a different level, understanding and evaluating the criteria adopted to establish creditworthiness is important for authorities. We will see a family of widgets capturing these business cases and experience them in action, taking this chance to activate KG-Roar on a *very large graph* with millions of links. In particular, we will focus on the cases where we are interested in understanding whether a guarantor y is financially too close to x, and so the default risk is not properly mitigated. We shall see how complex creditworthiness criteria can be compactly encoded in Vadalog and applied on *ultra-large-scale* settings. The following program exemplifies the kind of widget we will analyze.

$$\text{own}(x, y, w), x \neq y, w \geq 0.2 \rightarrow \text{coi}(x, y) \quad (6)$$

$$\text{coi}(x, y) \rightarrow \text{coi}(y, x) \quad (7)$$

$$\text{coi}(z, x), \text{coi}(z, y), z \neq x, z \neq y, x \neq y \rightarrow \text{coi}(x, y) \quad (8)$$

Rule 6 is the base case, which associates a potential conflict of interest (coi) with x owning more than 20% of y. Such a relationship is symmetric (Rule 7). Finally, by Rule 8 we activate *non-linar recursion* (i.e., the head is mutually dependent on two body predicates), to spot the cases in which a third party z—in conflict with both x and y—determines an indirect conflict between them.

Exploratory analysis. In the last part of the demonstration, the participants will be allowed to freely browse the KG, using contextual navigation controls (e.g., to expand nodes based on available widgets) or defining their own widget at runtime, trying to be creative in designing patterns of interest.

Performance highlights. A special eye will be given to showing the good scalability and performance properties of KG-Roar. We will discuss the *access plans* created by Vadalog for the reasoning tasks and the *OS processes* that implement the data pipeline.

REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. 1995. *Foundations of Databases.*
[2] Mario Alviano and Andreas Pieris (Eds.). 2022. *Datalog 2.0.* CEUR Workshop Proceedings, Vol. 3203. CEUR-WS.org.
[3] Luigi Bellomarini, Davide Benedetto, Matteo Brandetti, and Emanuel Sallinger. 2022. Exploiting the Power of Equality-Generating Dependencies in Ontological Reasoning. *VLDB* 15, 13 (2022), 3976–3988.
[4] Luigi Bellomarini, Davide Benedetto, Georg Gottlob, and Emanuel Sallinger. 2022. Vadalog: A modern architecture for automated reasoning with large knowledge graphs. *Inf. Syst.* 105 (2022), 101528.
[5] Luigi Bellomarini, Daniele Fakhoury, Georg Gottlob, and Emanuel Sallinger. 2019. Knowledge Graphs and Enterprise AI: The Promise of an Enabling Technology. In *ICDE*. IEEE, 26–37.
[6] Luigi Bellomarini, Emanuel Sallinger, and Georg Gottlob. 2018. The Vadalog System: Datalog-based Reasoning for Knowledge Graphs. *VLDB* 11, 9 (2018), 975–987.
[7] Gerald Berger, Georg Gottlob, Andreas Pieris, and Emanuel Sallinger. 2019. The Space-Efficient Core of Vadalog. In *PODS*. ACM, 270–284.
[8] Andrea Calì, Georg Gottlob, and Andreas Pieris. 2011. New expressive languages for ontological query answering. In *AAAI*.
[9] Lisa Ehrlinger and Wolfram Wöß. 2016. Towards a Definition of Knowledge Graphs. In *SEMANTiCS (CEUR Workshop Proceedings)*, Vol. 1695.
[10] Georg Gottlob and Andreas Pieris. 2015. Beyond SPARQL under OWL 2 QL Entailment Regime: Rules to the Rescue. In *IJCAI*. AAAI Press, 2999–3007.
[11] Georg Gottlob, Andreas Pieris, and Emanuel Sallinger. 2019. Vadalog: Recent Advances and Applications. In *JELIA (LNCS)*, Vol. 11468. Springer, 21–37.
[12] Andrea Gulino, Stefano Ceri, Georg Gottlob, Emanuel Sallinger, and Luigi Bellomarini. 2021. Distributed Company Control in Company Shareholding Graphs. In *ICDE*.