DIS: A Data-centred Knowledge Representation Formalism

Alicia Marinache
Ridha Khedri
Andrew LeClair
McMaster University
1280 Main St. West, Hamilton,
ON L8S 4K1, Canada

Wendy MacCaull
St Francis Xavier University
Antigonish
NS B2G 2W5, Canada

***Abstract*—The Domain Information System (DIS) is a novel data-centered knowledge representation formalism that offers a modular structure, which separates the domain knowledge (i.e., the ontology) from the domain data view. The formalism is designed to take advantage of a Cartesian perspective on information, using `partOf` as its fundamental ontological relation. A DIS consists of a domain ontology, to model the domain knowledge, a cylindric algebra, to model the domain data view, and an operator which couples the two. The core component of the ontology is a Boolean lattice built from atomic concepts taken from a data schema. It is enriched with relevant concepts from the domain of application. A detailed case study is presented to highlight the features of this formalism. The integration of several datasets and their respective ontologies for reasoning tasks requiring data-grounded and domain-related answers to user queries is illustrated. The ability to handle information evolution is demonstrated. Two open issues in ontologies, namely, the lack of clear and explicit guidelines for ontology construction, and the prohibitive cost of adapting and reusing existing ontologies, are addressed.**

I. Introduction

Analysing and processing stored data to extract useful information is an essential and frequently time consuming activity for most organisations. Among the issues that contribute to the difficulty are the lack of uniformity in terminology, the need to update data models due to frequent information evolution, and the need to integrate heterogeneous data schema from associated domains [1]. Ontologies, which are defined as shared conceptual representations of domains [2], have proved useful for harmonizing terminology. However, a typical ontology, which is built on a logical foundation starting with a taxonomy of concepts, is less suited to dealing with the problems of information evolution and integration. In an enterprise setting, data is being analysed and understood from different perspectives. Each perspective offers its own view of the application domain, and its own set of goals, expressed as user queries [3], [4]. Such systems allow the use of various sets of data to develop ontologies that represent each application domain independently of each other. This is referred to as semantic heterogeneity, and results in the need to unify the ontologies through techniques that reconcile the semantic differences among various domains [5].

978-1-7281-6937-8/20/$31.00 ©2020 IEEE

Domain representation formalisms, such as Description Logic (DL) [6], typically incorporate the data as part of the ontology. In order to access and process existing datasets in an efficient manner, there is a need for formalisms that separate the description of the domain (i.e., the ontology) from its content (i.e., the data). Current formalisms that achieve this separation include Ontology-Based Data Access (OBDA) systems [7], DIS [8], and DOGMA [9]. Typically, OBDA and DOGMA start from the premise that the data, usually called the *local view*, and the ontology, usually called the *global view*, have been built independently of each other [7]. One of the objectives of these techniques is to provide a mapping between the two views. However, matching the data schema to the concepts in the domain introduces non-trivial challenge to the field [7], [10]. In addition, as information evolves, a change at either the data or ontology level may render the mapping obsolete; consequently, the mapping may need to be rebuilt. Existing formalisms use semi-automated mapping tools. The field would benefit from a formalism where the mapping process can be fully automated.

The DIS formalism is a bottom-up, data-centered framework for knowledge representation, consisting of three parts: a domain ontology $\mathcal{O}$, a domain data view $\mathcal{A}$, and an operator τ that loosely couples $\mathcal{A}$ to $\mathcal{O}$. To model the domain knowledge, we use a monoid of concepts, a Boolean lattice, and a family of rooted graphs. We use Tarski's cylindric algebra [11] to model the data view. The construction of a DIS starts from an existing dataset, which is used to guide the design of the core component of the domain ontology (i.e., its Boolean lattice, as we explain below). In normalized databases, it has been demonstrated that the number of attributes in a dataset schema rarely exceeds ten [12]. Therefore, the Boolean lattice is often within the range of at most 1,024 elements. Other concepts and relations from the domain of application are added to the ontology to enrich it. By adopting a data-guided approach for the construction of the ontology, a DIS addresses the mapping challenge. On construction of the DIS, the mapping is performed in an automated manner, and changes of the domain or data need only localised modifications of the DIS and its mapping.

In [13], the authors describe the state of the formal ontology

field after two decades of research. Listed in [13] are some open issues, such as (1) the lack of clear and explicit guidelines for ontology construction, and (2) the prohibitive cost of adapting and reusing existing ontologies (e.g., removing irrelevant parts or adding new stakeholders' views). In this paper, we illustrate the use of the DIS formalism, showing how the two open issues are addressed.

The paper is organised as follows. In Section II, we give the formal definitions of the DIS, and the rationale for the proposed construction. In Section III, we present a case study illustrating how to design the ontology underlying a DIS, how to link the ontology to the dataset under consideration, and how to integrate it with DISs from other (associated) domains. In Section IV, we reason on the DIS developed in Section III to answer a number of user queries (or competency questions). In Section V, we discuss relevant existing formalisms, and show how the DIS formalism addresses some of the challenges discussed. In Section VI, we conclude the paper and point to future work.

II. Overview of the Domain Information System

We are interested in generating knowledge from organised datasets and domain knowledge, thus we start the construction of a DIS from an organised dataset. We focus on structured data, represented either as a set of tuples of the same size, such as records in a dataset, or of different sizes, such as in log files. Therefore, we consider the elements of data to be within a Cartesian space.

In a DIS, the concepts in the domain ontology are derived from two sources: (i) the data, and (ii) the domain of application. The concepts in the domain ontology can be composed to form new concepts. We understand the composition as the Cartesian construction of concepts. The composition operator is commutative (up to an isomorphism) and associative, to ensure that the order in which concepts are composed is not relevant. In addition, the composition operator is idempotent: composing a concept with itself does not create a new concept. In current solutions that connect data to an ontology, mapping is a time consuming task that needs to bridge the typically large conceptual gap between the two layers [7]. To avoid this issue, the core component of the domain ontology O of DIS is built using the Cartesian `partOf` relation. A concept with no sub-parts is called an *atomic* concept or an *atom*. We take a subset of atoms, where each atom corresponds to an attribute of the data schema, and use the `partOf` relation to generate a free Boolean lattice over this subset of atoms. By the construction of the lattice, the task of mapping the data to the ontology becomes trivial, as the attributes of the data view schema have a one-to-one correspondence to the atoms of the Boolean lattice.

Other concepts that are deemed significant by the domain experts, and are not part of the Boolean lattice, are captured from the domain of knowledge or from an existing ontology as rooted graphs. To ensure that all the concepts relevant to the domain have a link to the data view under consideration, each rooted graph has a unique root in the Boolean lattice. In addition, domain experts can define new concepts through axioms, using existing concepts, relations, and data values.

Since its introduction in [14], Codd's relational model of data has been accepted as a clear and succinct model for relational databases. This model offers great scalability; however it assumes a closed-world, in which only explicit data is recognized [15]. Typically, in a knowledge system no one domain expert has complete knowledge of a domain, therefore the system needs to operate under the open-world assumption (OWA). In [16], the authors use a finitary version of cylindric set algebras to deal with incomplete information in relational databases. In [11], the authors show there exists a natural embedding of a relational algebra into a diagonal-free cylindric set algebra (i.e., all operations available in a relational algebra can be performed on its corresponding cylindric algebra). Through the cylindrification operators, a cylindric algebra offers support for non-uniform structured data, taking the OWA approach. In our work, the OWA approach is restricted to the domain data view under consideration. If a record in the dataset is missing a value for one of its attributes, it may be replaced with other values from the dataset for that attribute.

We now introduce the formal definitions of the DIS components. We denote by C the set of all concepts under consideration and by $\oplus$ the composition operator on concepts. On the set of concepts, we define a pseudo-concept, e_C, that is neutral to the composition. Hence, the structure $\mathcal{C} = \big(C, \oplus, e_C\big)$ is an idempotent commutative monoid. The `partOf` relation is the ordering relation on concepts, denoted by $\sqsubseteq_C$, and defined as $k \sqsubseteq_C k' \overset{\text{def}}{\iff} k \oplus k' = k'$. An atom is defined as follows: k is atomic $\overset{\text{def}}{\iff} \forall (k' \mid k' \in C : k' \sqsubseteq_C k \implies k' = k \vee k' = e_C)$. The set of all atoms in C is denoted by $At(C)$.

The subset of atoms corresponding to the attributes of the data schema is denoted by $T \subseteq At(C)$. The free Boolean lattice generated over T is denoted by $\mathcal{L} = \big(L, \sqsubseteq_C\big)$, where $L = \mathcal{P}(T)$, the power set of T. An algebraic representation of $\mathcal{L}$ would involve a join operator, which is the composition operator $\oplus$, and its dual $\otimes$, the meet operator. The two operators distribute over each other. For a given concept κ, its complement is a concept κ' such that $\kappa \oplus \kappa' = \top_{\mathcal{L}}$ and $\kappa \otimes \kappa' = e_C$, where $\top_{\mathcal{L}}$ is the top of the lattice $\mathcal{L}$.

Before giving the formal definitions of the components of a DIS, we recall the notion of a rooted graph. Let $C_i \subseteq C$, $R_i \subseteq C_i \times C_i$, and $t_i \in C_i$. A rooted graph at t_i, $G_{t_i} = \big(C_i, R_i, t_i\big)$, is a connected directed graph of concepts in C_i with a unique *root* $t_i \in C_i$. We say that[1] $t_i \in C_i$ is the *root* of G_{t_i} *iff* $\forall (k \mid k \in C_i : k = t_i \vee (k, t_i) \in {R_i}^+)$, where ${R_i}^+$ is the transitive closure of R_i.

Definition 2.1 (Domain Ontology): Let $\mathcal{C} = \big(C, \oplus, e_C\big)$ be a commutative idempotent monoid. Let $\mathcal{L} = \big(L, \sqsubseteq_C\big)$ be a free Boolean lattice, generated from a set of atoms in C. Let I be a finite set of indices, and $\mathcal{G} = \{G_{t_i}\}_{t_i \in L, i \in I}$ a set of graphs rooted

[1] We adopt the uniform linear notation provided by Gries and Schneider in [17]. The general form of the notation is $\star(x \mid R : P)$ where $\star$ is the quantifier, x is the dummy or quantified variable, R is predicate representing the range, and P is an expression representing the body of the quantification.

in L. A *domain ontology* is the structure $O \stackrel{\text{def}}{=} (\mathcal{C}, \mathcal{L}, \mathcal{G})$.

Datasets with records of different lengths can be modeled by cylindric algebras, thus, the DIS domain data view component is formalised using a diagonal-free cylindric algebra. The cylindrification operators $\mathbf{c}_\kappa$ are indexed by the elements of L, the carrier set of the Boolean lattice $\mathcal{L}$ of the domain ontology, as described in Definition 2.1.

Definition 2.2 (Domain Data View associated with an ontology): Let $O = (\mathcal{C}, \mathcal{L}, \mathcal{G})$ be a domain ontology. A *domain data view* associated with O is a diagonal-free cylindric algebra $\mathcal{A} = (A, +, \star, -, 0_A, 1_A, \{\mathbf{c}_\kappa\}_{\kappa \in L})$, with L the carrier set of $\mathcal{L}$.

For the properties of a cylindric algebra, we refer the reader to [18]. The elements of A are understood as n-dimensional objects (i.e., a set of n-dimensional points), and the Boolean operators of $\mathcal{A}$ (i.e., $+, \star, -$) create new solids on A [18]. The cylindrification operator on the i-th dimension can be understood as a projection of the solid on the remaining $k-1$-dimensional space, which is then extended to the whole "cylinder" along the removed dimension. In a 3-dimensional space, if we let $a = \{(v_x, v_y, v_z)\} \subseteq X \times Y \times Z$ be an element of A, then $\mathbf{c}_Z(a) = \{(x, y, z) \mid x = v_x \wedge y = v_y \wedge z \in Z\}$.

Definition 2.3 (Domain Information System): Let O be a domain ontology, $\mathcal{A}$ its associated domain data view, and $\tau : A \to L$ a mapping that relates the elements of A to the elements of the Boolean lattice in O. A *Domain Information System (DIS)* is the structure $\mathcal{D} = (O, \mathcal{A}, \tau)$.

The components of DIS are analogous to those proposed in [19]; the domain data view corresponds to the Data Box (D-Box), the Boolean lattice $\mathcal{L}$ to the Assertional Box (A-Box), and the family of rooted graphs $\mathcal{G}$ (plus the domain expert-defined axioms) corresponds to the Terminological Box (T-Box). Using the τ operator, the domain data view is mapped to the Boolean lattice. Through the use of the rooted graphs, the domain expert captures concepts related to the data view. In addition, multiple DISs, each built from a specific data view, may be required to integrate knowledge from other application domains. Fig. 1 illustrates DIS integration; DIS $\mathcal{D} = (O, \mathcal{A}, \tau)$ is depicted in red, and DIS $\mathcal{D}' = (O', \mathcal{A}', \tau')$ is depicted in blue. The overlap shows concepts common to both $\mathcal{D}$ and $\mathcal{D}'$ (i.e., *Fresh*), or defined by the domain expert to be equivalent (i.e., $tconst \equiv imdbID$ and $\bot_{\mathcal{L}} \equiv \bot_{\mathcal{L}'}$). In Section III, we elaborate further on the use of multiple DISs.

The domain ontology can be extended using the lattice structure. Given a rooted graph $G_{t_i} \in \mathcal{G}$, we define the lattice extension of its relation R_i as a relation between elements of C_i and elements of L, as follows:

Definition 2.4 (Lattice extension of a rooted graph relation): Let $\mathcal{D} = (O, \mathcal{A}, \tau)$ be a DIS and let $G_{t_i} = (C_i, R_i, t_i)$, $G_{t_i} \in \mathcal{G}$ be a rooted graph, with $t_i \in L$ its root. We call the relation $R_i^{\uparrow} = R_i \cup R_i'$ the *lattice extension* of R_i, where $R_i' = \{(c, t) \mid c \in C_i \wedge t \in L \wedge (c, t_i) \in R_i \wedge t_i \sqsubseteq_C t\}$.

Lattice extensions can be used during the reasoning process to answer more complex queries.

We consider the ordered set of the attributes that form a datum $a \in A$ to be the "structure of a". When we cylindrify a on an index $\kappa \in L$, the structure of a may be extended by the concept κ. E.g., if $a \in A, \kappa_1, \kappa_2, \kappa_3 \in L$, such that $\tau(a) = \kappa_1 \times \kappa_2, \kappa_3 \not\sqsubseteq_C \kappa_1$ and $\kappa_3 \not\sqsubseteq_C \kappa_2$, then $\tau(\mathbf{c}_{\kappa_3}(a)) = \kappa_1 \times \kappa_2 \times \kappa_3$. Cylindrification of a on dimension κ_3 has extended the structure of a by κ_3. Since $\tau(\mathbf{c}_\kappa(a)) = \kappa \oplus \tau(a)$, where $a \in A$ and $\kappa \in L$, then for the case where $\kappa \sqsubseteq_C \tau(a)$, we get $\tau(\mathbf{c}_\kappa(a)) = \tau(a)$. For every $c, c' \in L$, we denote by $c \backslash c'$ the composition of all atoms that are `partOf` c, and not `partOf` c' as follows: $c \backslash c' = \oplus(k \mid k \in At(L) \wedge k \sqsubseteq_C c \wedge \neg(k \sqsubseteq_C c') : k)$.

On the elements $a \in A$, we define a *focusing*[2] operator that corresponds to the projection operator in Codd's relation algebra. We adopt the notation $a.\kappa$ for indicating the focusing of the datum $a \in A$ on the index κ. If κ is a part of $\tau(a)$, to focus a on κ means to find the element $b \in A$ such that $\tau(b) = \kappa$ and b is the projection of a on the concepts given by κ. This can be expressed using the cylindrification operator, by ensuring that cylindrification of both a and b is invariant on κ. Thus, the data view provides us with a language to describe properties on data elements. The focusing operator is used to obtains parts of a datum. If $\kappa \sqsubseteq_C \tau(a)$, the operator is defined as $a.\kappa \stackrel{\text{def}}{=} \{b \mid b \in A \wedge \tau(b) = \kappa \wedge \mathbf{c}_{(\tau(a) \backslash \kappa)} a = \mathbf{c}_{(\tau(a) \backslash \kappa)} b\}$. Otherwise, $a.\kappa \stackrel{\text{def}}{=} 0_A$.

For a more detailed description of DIS, we refer the reader to [8].

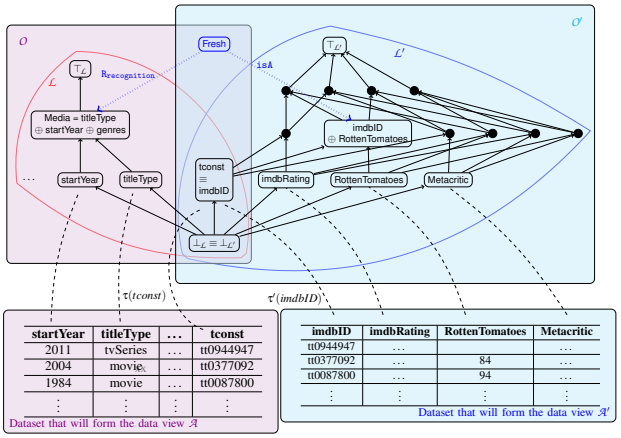

Fig. 1: Integrating Multiple DISs

III. Case Study: A Film and TV Domain Information System

Starting from public data available on the IMDb website [21] and the Rotten Tomatoes website (accessed through the Open Movie Database (OMDb) [22]), we illustrate the construction of the DISs for the Film and TV domain of application. We consider three views (or perspectives) on the application domain: the Media view V1, described by the IMDb 'Titles.basic' dataset; the People view V2, described by the IMDb 'Names.basic' and IMDb 'Titles.Principals' datasets; and the Reviews view V3, described by the OMDb (Rotten Tomatoes) dataset. The IMDb Titles dataset contains general data about movies and tv shows, such as title, genre,

[2]The term *focusing* is borrowed from information algebra [20].

type, and year. The IMDb Names dataset contains data about persons that work in the movie industry, such as name, and birth date, while the IMDb Principals dataset relates titles to the main credited persons. The OMDb dataset contains data about how well a title is received by fans and by critics.

A. Delimiting the Scope: The Competency Questions

In the V1 view, one goal is to find more information about movies. In the V2 view, we are interested in finding more information about credits. In the V3 view, we seek information about ratings. In order to limit the scope of the DIS, we guide the construction of the information system with the following Competency Questions (CQs) (their associated views are shown within parentheses):

CQ1 (V1) What movies are slashers?
CQ2 (V1, V2) What slashers are a critically acclaimed?
CQ3 (V2) Are there any infamous fresh media?
CQ4 (V1, V3) Who are the actors credited in the "Ghostbusters 2016" movie?

The notions of *slasher*, *infamous*, etc. cannot be found in the given data schema; they are defined by the domain expert. Therefore, CQ1, CQ2, and CQ3 cannot be answered through a simple database query.

B. Building the Boolean Lattice of the Domain Ontology

We remind the reader that a DIS is defined as the triple $\mathcal{D} = \big(O, \mathcal{A}, \tau\big)$. The component $\mathcal{A}$ models the dataset to be associated with the domain ontology O. A partial view of the IMDb Titles dataset is presented in Table I. It contains attributes such as *tconst*, the unique identifier for a record, *primaryTitle*, the title of the movie or series. An entry '\N' indicates an empty value.

The first step is to build the Boolean lattice, using the set of attributes of the IMDb Titles schema to generate the set of atoms. Thus, the set of atoms is given by $At(L) = \{$*tconst, titleType, primaryTitle, originalTitle, isAdult, startYear, endYear, runtimeMinutes, genres*$\}$. The carrier set of the Boolean lattice includes the pseudo-concept e_C, called the empty concept, that corresponds to $\oplus(k \mid false : k)$. The Boolean lattice, $\mathcal{L}$, is the free structure generated over $At(L)$ by the composition operator $\oplus$. In this case, the lattice is formed over 9 atoms. In Fig. 2, due to its size, we present only a partial view of the Boolean lattice. The top of the lattice is the composition of all of the atoms, and the bottom is the empty concept.

The composite lattice concepts that were recognised as significant by the domain expert are named and defined next. For instance, as shown in Fig. 2, concepts *Media* and *Listing* are defined as $Listing = titleType \oplus genres$ and $Media = titleType \oplus primaryTitle \oplus startYear \oplus genres$. The remaining composite concepts in the Boolean lattice (such as $tconst \oplus titleType \oplus genres$) exist and may be used by the reasoning process, however they are not described here with a commonly used identifier.

The operator τ of a DIS triple maps the data view elements to the concepts of the Boolean lattice. The attributes of a datum $a \in A$ have each a corresponding concept through the operator τ. Thus, the corresponding concept of a datum $a \in A$ is given by the composition of the atoms that correspond to the attributes of a. For example, given an element $a \in A$, $a = \{(\text{'}movie\text{'}, \text{'}Mean\ Girls\text{'}, 2004, \text{'}Comedy\text{'})\}$, we say that $\tau(a) = Media$. The cylindrification of a on the *runtimeMinutes* dimension will extend the structure of a with the new dimension, $\tau(\mathbf{c}_{runtimeMinutes}(a)) = \tau(a) \oplus runtimeMinutes$:

$$
\begin{aligned}
&\mathbf{c}_{runtimeMinutes}(a) = \\
&\quad \{(\text{'}movie\text{'}, \text{'}Mean\ Girls\text{'}, 2004, \text{'}Comedy\text{'}, 57), \\
&\quad\ (\text{'}movie\text{'}, \text{'}Mean\ Girls\text{'}, 2004, \text{'}Comedy\text{'}, 91), \\
&\quad\ (\text{'}movie\text{'}, \text{'}Mean\ Girls\text{'}, 2004, \text{'}Comedy\text{'}, 121)\}.
\end{aligned} \tag{1}
$$

This corresponds to a form of the open-world assumption. If, on a given tuple, the dataset does not contain values for an attribute, we provide a method of replacing the missing data value with possible values corresponding to that attribute.

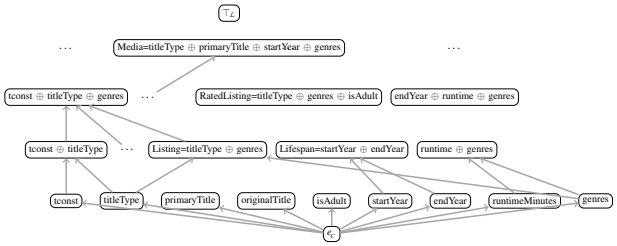

Fig. 2: IMDb Titles Boolean Lattice (a partial view)

C. Constructing the Rooted Graphs

The concepts within the Boolean lattice alone are not enough to answer all the competency questions, as they do not contain notions of *Slasher* or *Fresh*. For concept *Slasher*, it suffices to constrain (i.e., specialise) the concepts of the Boolean lattice with data values. For concept *Fresh*, no combination of the tuples will produce a concept that captures the information of a fresh rating. However, we have access to data that includes this information, in particular, the OMDb dataset. In this section, we show how the creation of rooted graphs assists with specialising concepts within the lattice, as well as with linking additional datasets to the Boolean lattice.

In taxonomic ontologies, the `isA` relations are described using graphs. In a DIS, there are two ways of capturing the `isA` relations. Specialisation of lattice concepts are described through axioms that use concepts of the domain ontology and values from the data view. Specialisations of non-lattice concepts are described using graphs. For example, the domain expert describes the *Slasher* concept to be a movie that is either a horror or thriller, and it was released between the years 1970 and 1990. Therefore, the *Slasher* concept is a specialisation of the *Media* concept. Other specialisations of the concept *Media* are *Comedy* or *TV Show*, and they are defined as follows:

$$
\begin{aligned}
Slasher = \{a \mid\ & \tau(a) = Media \wedge a.titleType = \text{"}movie\text{"} \\
& \wedge \big(\text{"}Horror\text{"} \in a.genres \ \vee\ \text{"}Thriller\text{"} \in a.genres\big) \\
& \wedge\ 1970 < a.startYear < 1990\};
\end{aligned} \tag{2}
$$

TABLE I: A Partial View of the IMDb Titles dataset

tconst	titleType	primaryTitle	originalTitle	isAdult	startYear	endYear	runtime Minutes	genres
tt0944947	tvSeries	Game of Thrones	Game of Thrones	0	2011	2019	57	Fantasy, Adventure, Drama
tt0377092	movie	Mean Girls	Mean Girls	0	2004	\N	\N	Comedy
tt0087800	movie	A Nightmare on Elm Street	A Nightmare on Elm Street	0	1984	\N	91	Horror
tt0076759	movie	Star Wars: Episode IV - A New Hope	Star Wars	0	1977	\N	121	Action, Adventure, Fantasy
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

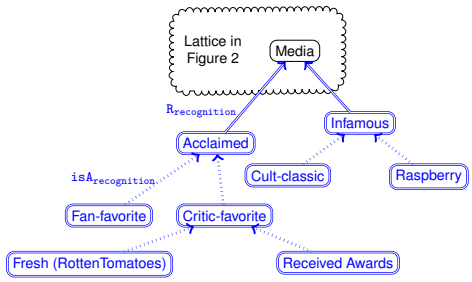

Fig. 3: Media Rooted Graph for $\mathtt{R}_{\mathtt{recognition}}$ relation

$$Comedy = \{a \mid \tau(a) = Media \wedge \text{“}Comedy\text{”} \in a.genres\};$$
$$TV\ Show = \{a \mid \tau(a) = Media \wedge a.titleType = \text{“}tvSeries\text{”}\}.$$

The *a.genres* is the value obtained by focusing the datum a on the *genres* concept. We note that in Equation (2), the variable a is free, as we are defining a concept that can be found in different data views. We will see later in the examples of Section IV that when we consider a specific dataset, a becomes bound to the set of values in that dataset.

The domain expert recognises that the concepts *Acclaimed* and *Fresh* are needed to answer the competency questions. However, within the IMDb Titles schema, there are no attributes to define the concept *Fresh* through specialisation or composition of existing concepts. Thus, the domain expert must define these concepts and relate them to the Boolean lattice. This is achieved by building a rooted graph that describes various types of acclamation, and connecting it to the Boolean lattice. In Fig. 3, we show such a graph, formed over the $\mathtt{R}_{\mathtt{recognition}}$ relation, depicted with a double line. The edges of the graph are concepts from the V3 domain that describe how a media can be recognised by both critics and fans, such as *Acclaimed* or *Infamous*. The graph connects these concepts to the Boolean composite concept *Media* (i.e., the root of the graph). In addition, depicted with a double dotted line, the graph describes the taxonomy of the recognition concepts. The concepts *Acclaimed* and *Infamous* are not part of the Boolean lattice, therefore the $\mathtt{isA}_{\mathtt{recognition}}$ relation is described in the classic way, as a taxonomic graph. The relation of the graph is understood as $\mathtt{R}'_{\mathtt{recognition}} = \mathtt{R}_{\mathtt{recognition}} \cup \mathtt{isA}_{\mathtt{recognition}}$. To define these concepts outside the Boolean lattice, we must bring in additional datasets, through other DISs.

In Fig. 3, the concepts depicted in a double line (i.e., *Acclaimed*, *Infamous*, etc.) are captured through a second DIS, $\mathcal{D}' = (\mathcal{O}', \mathcal{A}', \tau')$, built from the OMDb dataset. Its set of attributes, $\{imdbID, imdbRating, RottenTomatoes, Metacritic\}$, is used to build the Boolean lattice $\mathcal{L}'$.

The two DIS constructions, $\mathcal{D}$ and $\mathcal{D}'$, share some concepts, as illustrated in Fig. 1. In our example, the concepts $tconst \in C$ and $imdbID \in C'$ are equivalent. We chose to depict them as the same concept in Fig. 1. In addition, the concept *Fresh* is part of both DISs. In $\mathcal{D}$, *Fresh* is an unspecified concept in a rooted graph with the $\mathtt{R}_{\mathtt{recognition}}$ relation, whereas in $\mathcal{D}'$, it is a specialisation, given by the following axiom:

$$Fresh = \{a \mid \tau'(a) = imdbID \oplus RottenTomatoes \wedge a.RottenTomatoes \geq 70\}.$$

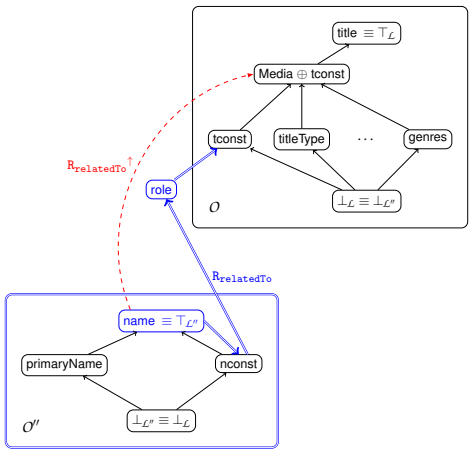

Fig. 4: Linking multiple DISs

To answer the CQ4, the domain expert uses the IMDb Names schema to create the core (i.e., the Boolean lattice) of a third DIS and describe details about persons in the domain. The attribute *nconst* is the unique identifier of the IMDb Names schema. The data in the IMDb Principals dataset shows persons credited for titles, by providing links between the atoms *nconst*, *tconst* of the two Boolean lattices. The end-users of the ontology would be more interested in the relations between the top concepts or other named concepts. This link is provided through the notion of lattice extensions, illustrated in Fig. 4. The relation $\mathtt{R}_{\mathtt{relatedTo}}$ (depicted with a double line) is given by enumeration $\{(name, nconst), (nconst, role), (role, tconst)\}$. By assuming the relation $\mathtt{R}_{\mathtt{relatedTo}}$ is transitive, the *name* concept is linked to the concept $Media \oplus tconst$ through the lattice extension (see Definition 2.4) $R_{relatedTo}{}^{\uparrow}$, depicted with a dashed line. We show how lattice extensions are used to answer queries that involve multiple DISs in Section IV-D.

Finally, through axioms that are at a strictly conceptual level (i.e., they do not rely on the datasets), the domain expert can express more complex relationships between the concepts of the domain. For example, to assist with answering CQ3, the

domain expert states that a media that is acclaimed cannot be categorised as infamous with the following axiom:

$$Acclaimed \cap Infamous = \emptyset. \tag{3}$$

So far, the concepts in the DIS have been specified in a virtual manner, without being instantiated (or materialised) by data. A concept $c \in C$ that is part of the Boolean lattice or is defined as a specialisation of a Boolean lattice concept can be instantiated using the tuples $a \in A$ in the data view. For example, assume there is an axiom defining that "A Horror movie or TV show must have at least a PG13 rating". Then the abstract definition of concept *Slasher*, along with the DIS axioms suffices to infer that "A *Slasher* must have at least a PG13 rating". However, to answer the CQ1 query, the concept *Slasher* must be instantiated using the tuples in the data view. This approach allows us to consider an abstract schema of the data view; by plugging in different data views, we offer different instantiations of the domain ontology. The only constraint for replacing a data view with another one is that they have the same set of attributes (i.e., their schema corresponds to identical Boolean lattices).

IV. Reasoning examples

In this section, we sketch reasoning examples that use the DISs built in Section III to answer the CQs given in Subsection III-A. Some of the queries that we used in this section (such as CQ1 or CQ4) are similar to queries found in the literature [23], and have been included to show that the DIS formalism has the capabilities of existing formalisms in terms of reasoning. The rest of the queries have been added to showcase the integration of multiple DISs. Some CQs (such as CQ4) can be answered through a direct database query. For others (such as CQ1, CQ2, CQ3), the information is not captured directly within the dataset, and can only be answered by defining new concepts, such as *Slasher*.

We use all of the mentioned datasets and we build multiple DISs; an IMDb Titles dataset-based DIS ($\mathcal{D}$), an OMDb dataset-based DIS ($\mathcal{D}'$), an IMDb Principals dataset-based DIS ($\mathcal{D}''$), and an IMDb Names dataset-based DIS ($\mathcal{D}'''$). We denote by A, A', A'', and A''' the carrier sets of the domain data views constructed from the IMDb Titles, OMDb, IMDb Principals, and IMDb Names schema, respectively.

A. Answering CQ1: What movies are slashers?

In Equation (2), the concept *Slasher* was specified without any reference to specific datasets. In the context of $\mathcal{D}$, it is instantiated to the IMDb Titles dataset. The answer to CQ1 is given by the following set:

$$\begin{aligned} Slasher(A) = \{a \in A \mid \tau(a) = Media \\ \wedge\, a.titleType = \text{“}movie\text{”} \\ \wedge\, (\text{“}Horror\text{”} \in a.genres \ \vee\ \text{“}Thriller\text{”} \in a.genres) \\ \wedge\, 1970 < a.startYear < 1990\}. \end{aligned} \tag{4}$$

The query result set includes tuples such as:
(‘*movie*’, ‘*A Nightmare on Elm Street*’, 1984, ‘*Horror*’).

B. Answering CQ2: What slashers are critically acclaimed?

As described in Section IV-A, the concept *Slasher* is instantiated in the context of $\mathcal{D}$, to the carrier set A of its data view. In this context, there is not enough information to refine the *Slasher* instances to the critically acclaimed ones. We make use of the $\mathtt{R}_{\mathtt{recognition}}$ rooted graph, to bring in the concept *Fresh*, through $\mathcal{D}'$. The attributes corresponding to the *tconst* and *imdbID* concepts are considered the unique identifiers of their respective data schemas, and the two concepts are equivalent, as described in Section III-C. The *Fresh* concept is instantiated as follows:

$$\begin{aligned} Fresh(A') = \\ \{a \in A' \mid \tau'(a) = RT \wedge a.RottenTomatoes \geq 70\}. \end{aligned} \tag{5}$$

Using Equations (4) and (5), we give the answer for CQ2 as the following set:

$$\begin{aligned} \{a \in Slasher(A) \mid (\exists b, b' \mid b \in A \wedge b' \in Fresh(A') : \\ \tau(b) = Media \oplus tconst \wedge a \leq b \wedge b.tconst = b'.imdbID)\}. \end{aligned}$$

The query result set includes tuples such as:
(‘*movie*’, ‘*A Nightmare on Elm Street*’, 1984, ‘*Horror*’).

C. Answering CQ3: Are there any infamous fresh media?

This CQ can be answered by observing that, in $\mathcal{D}'$, the concept *Fresh* is on the concept *Acclaimed* branch of the `isA` rooted graph. Therefore, we have

$$Fresh \subseteq Acclaimed. \tag{6}$$

From Equations (3) and (6), it is immediate that $Fresh \cap Infamous = \emptyset$, therefore, the answer to CQ3 is "No".

D. Answering CQ4: Who are the actors credited in the "Ghostbusters 2016" movie?

For CQ4, we use the link between the $\mathcal{D}$, $\mathcal{D}''$, and $\mathcal{D}'''$. There are two approaches to answer this competency question. The first one uses a database-like query, and the second approach uses of the $\mathtt{R}_{\mathtt{relatedTo}}$ rooted graph (see Fig. 4). We illustrate the second approach below:

$$\begin{aligned} Name(A, A'', A''') = \\ \{a.primaryName \mid a \in Name(A''') \wedge \\ \exists (b, c \mid b \in Principal(A'') \wedge c \in Media(A) : \\ (a,b) \in \mathtt{R}_{\mathtt{relatedTo}}{}^{\uparrow} \wedge (b,c) \in \mathtt{R}_{\mathtt{relatedTo}}{}^{\uparrow} \\ \wedge\, b.role = \text{‘}actor\text{’} \wedge c.startYear = 2016 \\ \wedge\, c.primaryTitle = \text{‘}Ghostbusters\text{’})\}. \end{aligned}$$

Both approaches produce the same set of tuples, including the tuple (‘*Chris Hemsworth*’).

V. Related Work and Discussion

There already exist solutions to the challenge of decoupling the data view from the domain knowledge representation [7], [9]. The primary goals of these solutions are to allow the ontology to adapt easily to changes in both the domain and the data, and to provide a unified access to distinct data

sources. These solutions develop the global domain view (i.e., the ontology) independently from the local domain view (i.e., the data). Even when they are not, it is likely there exists a mismatch between the two views, due to the differences in relational representation (taxonomic vs. mereologic) [7]. In addition, as the language of the domain representation (usually DL-based) and the language of the data source (usually SQL-like) are different, the user queries need to be translated to and from these languages, which is an inefficient process [7]. The literature is abound with optimisation research in both areas [24], [25], [26], however, as yet, no available solutions are generally accepted.

One widely used formalism for domain knowledge specification is Web Ontology Language (OWL) [27], which is a set of ontology languages that have their semantics based on DL [6]. An ontology specified using DL starts as a taxonomy, based on the `isA` relation, then it is enriched with other binary relations and axioms. This approach works well for describing a domain, as humans look at the world through classes and subclasses. However, the taxonomic approach poses a problem when existing data is incorporated into the system, as the main relation between the attributes of a data schema is given by the mereological `partOf` relation.

Ontology evolution is sparked by one of three actions: the change of data, the change of the data schema, or the change of the domain [28]. However, it is difficult to both determine when evolution must occur and to validate that the changes made to the ontology are appropriate. Given an action that sparks ontology evolution, it is not clear the degree to which the changes might ripple out, requiring updates on various parts of the domain (or on other ontologies in an interconnected scenario). Ontology evolution is a topic commonly discussed in the context of OBDA [7]. In addition, providing different views of the same data and hiding information is important [13]. We discuss below how the DIS formalism approaches all these challenges.

The DIS framework explores similar directions as DL, OBDA, and DOGMA. While it can be argued that DL offers a modular structure by separating the A-Box from the T-Box, this is a logical separation. With the DIS formalism, we offer a clear separation of the domain ontology (i.e., $\mathcal{O}$) from the data view (i.e., $\mathcal{A}$). Unlike existing formalisms, we start the construction of a domain ontology from an existing dataset, therefore offsetting the mismatch between the data and ontology by construction. The use of the `partOf` relation in building the Boolean lattice allows for a direct correspondence between the attributes in the data view and the atomic concepts in the Boolean lattice. This correspondence can be captured in an automated manner by the τ operator. The DIS approach offers a minimal coupling of the data view component to the domain ontology component. There is no need to perform expensive mappings between the domain data view $\mathcal{A}$ and the domain ontology $\mathcal{O}$.

When using a DIS, the effort to maintain the system is lessened, as we can determine exactly which component of the DIS is impacted by change. If the source of information evolution is a change in the data content (i.e., records are added or deleted to or from the dataset), only the domain data view and the mapping operator need to be updated, and there is no impact on the domain ontology component. This task can be fully automated, which makes it efficient. If the data schema is changed (i.e., attributes in the dataset are modified), the domain ontology needs to be modified; however the impact on it is localised. The Boolean lattice can be rebuilt in an automated manner, and the graphs and axioms affected by change can be identified automatically. If the source of evolution is a change in the domain of application, it translates to a need to modify the rooted graphs and/or the axioms of the DIS. The Boolean lattice is not affected, as it is linked strictly to the domain data view, which remain unchanged. Thus, a DIS handles information evolution with localised effort.

VI. Conclusion and Future Work

Our research addresses two open issues in the field; (1) the lack of clear and explicit guidelines for ontology construction, and (2) the prohibitive cost of adapting and reusing existing ontologies. As a solution for (1), instead of an ad-hoc process for ontology building, we provide a clear, systematic, rigorous process for representing knowledge, guided by existing datasets and competency questions. The DIS construction process assists the domain experts with existing ontology construction issues, such as possible semantic confusion between concepts with similar names [29]. In a DIS, a distinct Boolean lattice is built for each domain of application, and a concept in one domain cannot be mistaken for a concept in the other domain, unless explicitly defined as such. For example, in Fig. 1, *tconst* and *imdbID* are explicitely defined as being equivalent. Without this equivalence, the two concepts would be semantically distinct. As the core component of each DIS is guided by existing datasets, a concept occurring in one of the domains of applications (e.g., a *spa* concept within the health industry) is clearly different from a concept occurring in another domain (e.g., a *spa* concept within the tourism industry). By the process of building a DIS, one *spa* concept is semantically distinct from the other.

Regarding (2), we show that the modular structure of a DIS facilitates its modification by the domain expert. As described above, adding or modifying the stakeholder's point of view is a localised process, in which only (parts of the) rooted graphs and/or axioms of the DIS are to be modified, while the rest of the DIS remains unchanged. In addition, hiding information or pruning irrelevant parts of the DIS is enabled through the process of modularization. In [30], the authors show that modularization of any DIS is both a systematic and straightforward process. The work presented in [30] demonstrates that the loss of knowledge due to the modularization using view traversal is quantified and expressed using the kernel.

The DIS formalism is well suited for structured organised datasets, and less ideal for mapping natural language text data. To model the data view, a DIS uses the schema of existing structured datasets, organised as sets of tuples. The domain of

application is then conceptualised through the components of the DIS domain ontology, as described in this work.

Our future work takes several directions. The first one is to build a user-friendly language to specify a DIS. This language would hide the mathematical background and allow a domain expert to employ commonly used terms in declaring concepts and relationships. The second direction is to adapt existing ontology bootstrapping techniques [31], to automate the construction of some parts of a DIS from existing datasets, such as the domain data view, the Boolean lattice of the domain ontology, as well as the mapping operator τ. The construction of the DIS will then require intervention from the domain expert only to elicit domain knowledge by asking for names of specific, domain-relevant composite concepts, and by defining the rooted graphs, and the domain-specific axioms. Another direction to explore is to develop reasoning tools to generate knowledge on a given DIS. We are currently using the proof assistant Isabelle/HOL [32] to formally define the DIS theory, as an extension of the HOL-Algebra theories with explicit carrier sets. While using implicit carrier sets in Isablle to reason on algebraic structures leads to complex algebraic constructions that are less related to the original theories and proofs [33], we adopt an efficient alternative way to reason about algebras. It consists of explicitly representing their carrier sets in the corresponding classes and locales. This allows us to use a less complex approach for building the free Boolean lattice component, and for reasoning on a DIS. Future steps will make use of the cylindrification operators and their OWA capabilities, to generate knowledge from existing data and domain knowledge.

REFERENCES

[1] A. Pomp, A. Paulus, A. Kirmse, V. Kraus, and T. Meisen, "Applying semantics to reduce the time to analytics within complex heterogeneous infrastructures," *Technologies*, vol. 6, no. 3, p. 86, 2018.

[2] N. Guarino, D. Oberle, and S. Staab, "What is an Ontology?" in *Handbook on ontologies*. Springer, 2009, pp. 1–17.

[3] N. Anand, R. van Duin, and L. Tavasszy, "Ontology-based multi-agent system for urban freight transportation," *International Journal of Urban Sciences*, vol. 18, no. 2, pp. 133–153, 2014.

[4] C. Legat, C. Seitz, S. Lamparter, and S. Feldmann, "Semantics to the shop floor: towards ontology modularization and reuse in the automation domain," *IFAC Proc. Vol.*, vol. 47, no. 3, pp. 3444–3449, 2014.

[5] M. Maree and M. Belkhatir, "Addressing semantic heterogeneity through multiple knowledge base assisted merging of domain-specific ontologies," *Knowledge-Based Systems*, vol. 73, pp. 199–211, 2015.

[6] F. Baader, D. Calvanese, D. McGuinness, P. Patel-Schneider, and D. Nardi, *The description logic handbook: Theory, implementation and applications*. Cambridge university press, 2003.

[7] G. Xiao, D. Calvanese, R. Kontchakov, D. Lembo, A. Poggi, R. Rosati, and M. Zakharyaschev, "Ontology-based data access: a survey," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. AAAI Press, 2018, pp. 5511–5519.

[8] A. Marinache, R. Khedri, and W. MacCaull, "A data-centered framework for domain knowledge representation," McMaster University, Tech. Rep. CAS-19-09-RK, 2019.

[9] M. Jarrar and R. Meersman, "Ontology engineering–the DOGMA approach," in *Advances in Web Semantics I*. Springer, 2008, pp. 7–34.

[10] J. F. Sequeda, "Ontology based data access: Where do the ontologies and mappings come from?" in *AMW*, 2017.

[11] T. Imielinski and W. Lipski Jr, "The relational model of data and cylindric algebras," *Journal of Computer and System Sciences*, vol. 28, no. 1, pp. 80–102, 1984.

[12] P. Vassiliadis, A. V. Zarras, and I. Skoulis, "How is life for a table in an evolving relational schema? Birth, death and everything in between," in *International Conference on Conceptual Modeling*. Springer, 2015, pp. 453–466.

[13] S. Borgo and P. Hitzler, "Some open issues after twenty years of formal ontology." in *FOIS*, 2018, pp. 1–9.

[14] E. F. Codd, "A relational model of data for large shared data banks," *Communications of the ACM*, vol. 13, no. 6, pp. 377–387, 1970.

[15] D. Moldovan, M. Antal, D. Valea, C. Pop, T. Cioara, I. Anghel, and I. Salomie, "Tools for mapping ontologies to relational databases: A comparative evaluation," in *2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 2015, pp. 77–83.

[16] G. Grahne and A. Moallemi, "Universal (and existential) nulls," *Fundamenta Informaticae*, vol. 167, no. 4, pp. 287–321, 2019.

[17] D. Gries and F. Schenider, *A Logical Approach to Discrete Math*, ser. Springer Texts And Monographs In Computer Science. New York: Springer-Verlag, 1993.

[18] A. Tarski, L. Henkin, and J. Monk, *Cylindric Algebras*. North-Holland, 1971.

[19] D. Calvanese and E. Franconi, "First-order ontology mediated database querying via query reformulation," in *A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years*. Springer, 2018, pp. 169–185.

[20] J. Kohlas and R. F. Stärk, "Information algebras and consequence operators," *Logica Universalis*, vol. 1, no. 1, pp. 139–165, 2007.

[21] "IMDb Datasets," http://www.imdb.com/interfaces/, 2020, accessed: Dec. 10, 2020.

[22] "The Open Movie Database," http://www.omdbapi.com/, 2019, accessed: Dec. 8, 2020.

[23] T. Bagosi, D. Calvanese, J. Hardi, S. Komla-Ebri, D. Lanti, M. Rezk, M. Rodríguez-Muro, M. Slusnys, and G. Xiao, "The Ontop framework for ontology based data access," in *Chinese Semantic Web and Web Science Conference*. Springer, 2014, pp. 67–77.

[24] M. Bienvenu, S. Kikot, R. Kontchakov, V. V. Podolskii, and M. Zakharyaschev, "Ontology-mediated queries: Combined complexity and succinctness of rewritings via circuit complexity," *Journal of the ACM (JACM)*, vol. 65, no. 5, p. 28, 2018.

[25] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro, and G. Xiao, "Ontop: Answering SPARQL queries over relational databases," *Semantic Web*, vol. 8, no. 3, pp. 471–487, 2017.

[26] D. Hovland, D. Lanti, M. Rezk, and G. Xiao, "OBDA constraints for effective query answering," in *International Symposium on Rules and Rule Markup Languages for the Semantic Web*, J. J. Alferes, L. Bertossi, G. Governatori, P. Fodor, and D. Roman, Eds., Springer. Springer, Cham, 2016, pp. 269–286.

[27] WC3, "OWL 2 recommendation," http://www.w3.org, 2012, accessed: Dec. 14, 2020.

[28] F. Zablith, G. Antoniou, M. d'Aquin, G. Flouris, H. Kondylakis, E. Motta, D. Plexousakis, and M. Sabou, "Ontology evolution: a process-centric survey," *The knowledge engineering review*, vol. 30, no. 1, pp. 45–75, 2015.

[29] C. Chantrapornchai and C. Choksuchat, "Ontology construction and application in practice case study of health tourism in Thailand," *SpringerPlus*, vol. 5, no. 1, p. 2106, 2016.

[30] A. LeClair, R. Khedri, and A. Marinache, "Toward measuring knowledge loss due to ontology modularization," in *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - Volume 2: KEOD*, INSTICC. SciTePress, 2019, pp. 172–184.

[31] G. Xiao, L. Ding, B. Cogrel, and D. Calvanese, "Virtual knowledge graphs: An overview of systems and use cases," *Data Intelligence*, vol. 1, no. 3, pp. 201–223, 2019.

[32] "Isabelle," http://isabelle.in.tum.de/index.html, accessed: Jan. 9, 2021.

[33] W. Guttmann, "Reasoning about algebraic structures with implicit carriers in Isabelle/HOL," in *International Joint Conference on Automated Reasoning*. Springer, 2020, pp. 236–253.