Characterizing Nexus of Similarity between Entities

Giuseppe Agresta[1], Giovanni Amendola[1,*], Pietro Cofone[1], Marco Manna[1] and Aldo Ricioppo[1]

[1] *Department of Mathematics and Computer Science, University of Calabria, Italy*

Abstract

Similarities play a significant role in diverse real-world scenarios. Researchers across various fields have proposed different methodologies for measuring entity similarity and expanding sets of entities with similar ones. As a result, modern machines are adept at performing these tasks by taking in some regard relevant interconnected properties shared by entities, which we refer to as nexus of similarity. To complement existing approaches, we present a very general logic-based framework, equipped with a suitable formal semantics, for characterizing nexus of similarity between (tuples of) entities of any knowledge base, namely express such nexus, formally and comprehensively, in a manner that they are both understandable to machines and humans.

Keywords

Logic-based framework, Formal semantics, Nexus of similarity, Knowledge bases

1. Introduction

In real-life and everyday scenarios, the recognition and reasoning about similarities between entities often play a significant role. Even at a young age, kids begin to informally describe, classify, and compare entities. For example, they can easily recognize that both ⟨Paris⟩ and ⟨Rome⟩ are "cities", and that ⟨Paris⟩ is more similar to ⟨Rome⟩ than to ⟨Gardaland⟩. Growing up, people become also able to identify and explain *relevant interconnected properties shared by entities*, hereinafter called *nexus of similarity*. Adults can easily agree that ⟨Paris⟩ and ⟨Rome⟩ are both "Europe's capitals situated on rivers". Likewise, one can identify nexus of similarity between *n*-ary tuples of entities. For instance, ⟨Tokyo, Tokyo Tower⟩ and ⟨Paris, Eiffel Tower⟩ are fairly similar, as each is a "capital paired with one of its monument being a tower made of metal".

As we move towards more complex entities such as goods, services, or health conditions, the challenges become greater, and the stakes become more interesting and valuable. For example, in e-commerce, Amazon's recommendation system may suggest specific smartphones to a user interested in high-end devices equipped with features like accelerometer, compass, fingerprint, and Android 14. In the tourism industry, travel agencies may recommend other theme parks to

IPS-RCRA-SPIRIT 2023: Italian Workshop on Planning and Scheduling, RCRA Workshop on Experimental evaluation of algorithms for solving problems with combinatorial explosion, and SPIRIT Workshop on Strategies, Prediction, Interaction, and Reasoning in Italy. November 7-9th, 2023, Rome, Italy [1]

*Corresponding author.

gius3ppe.agresta@gmail.com (G. Agresta); giovanni.amendola@unical.it (G. Amendola); pietrocf02@outlook.it (P. Cofone); marco.manna@unical.it (M. Manna); aldo.ricioppo@unical.it (A. Ricioppo)

0000-0002-2111-9671 (G. Amendola); 0000-0003-3323-9328 (M. Manna)

a family based on those they have previously visited. Streaming services such as Netflix may suggest trailers to their customers based on their previous viewing history and preferences. In medicine, researchers may need to understand why certain individuals are more susceptible to certain diseases than others. Clearly, in all the considered real-world scenarios, nexus of similarity play a crucial role.

For over a century, researchers from various fields have proposed a range of approaches to measure the *semantic similarity* between entities, usually expressed in the form of a descriptive rating or a numerical score [2]. For example, computing machines nowadays reached a level of advancement where they are capable of computing a plausibly high similarity score between ⟨Paris⟩ and ⟨Rome⟩, by taking into account somehow that both of them are "European cities", "places situated on rivers", "capitals located in states that founded the European Economic Community", and so on. Moreover, some approaches are also capable of detecting that ⟨Paris⟩ and ⟨Eiffel Tower⟩ are not very similar, despite their high level of relatedness [3]. Finally, by following the same rationale, machines are also able to classify ⟨Rome⟩ as a "capital" rather then a "state" or a "park", by comparing similarity scores between the considered entity and some class names.

In the past two decades, inspired by "Google Sets" [4], considerable academic and commercial efforts have been devoted to providing solutions for expanding a given set of entities with similar ones. The most studied tasks here are *entity set expansion* [5], *entity recommendation* [6], *tuples expansion* [7], or *entity suggestion* [8]. For example, via existing approaches one can expand the set $U = \{\langle\text{Paris}\rangle, \langle\text{Rome}\rangle\}$ and obtain, for example, $U' = U \cup \{\langle\text{Amsterdam}\rangle\}$; then, one can reapply the process starting from U' to obtain, for example, the expanded set $U'' = U' \cup \{\langle\text{Brussels}\rangle, \langle\text{Rio de Janeiro}\rangle, \langle\text{Vienna}\rangle\}$. Indeed, all the elements of these sets share one or more of the aforementioned properties, namely "European cities", "places situated on rivers", etc.

Traditional approaches primarily measure similarities within (hyper)text corpora or tabular data [9, 10]. In recent years, there has been an increasing trend in exploiting structured knowledge bases (KBs), often represented as knowledge graphs (KGs) [11, 12]. Indeed, 'the heterogeneity, semantic richness and large-scale nature of knowledge base make traditional approaches less effective' [13].

The work conducted thus far is remarkable, as well as the achieved results. However, there remain some foundational aspects that, in our perspective, warrant further exploration. We propose a very general logic-based framework, equipped with a suitable formal semantics, for characterizing nexus of similarity between (tuples of) entities of any knowledge base, namely express such nexus, formally and comprehensively, so that the resulting explanations are readable by both humans and machines.

In Section 2, we introduce the notion of selective knowledge base, and our nexus explanation language together with an appropriate semantics taking into account summaries. Moreover, we illustrate how to characterize the nexus of similarity between tuples of entities. In Section 3, we show how to construct logic formulas, called canonical characterizations, that characterize the nexus of similarity between tuples of entities. Finally, in Section 5 we draw our conclusions.

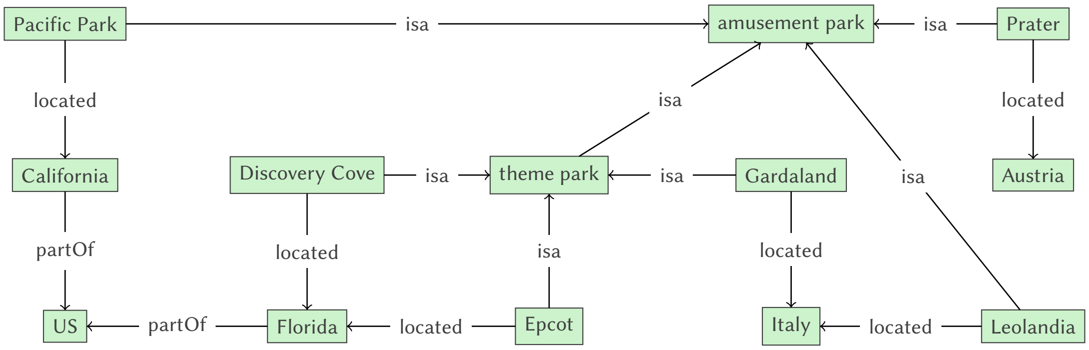

Figure 1: Knowledge Graph $\mathcal{G}$ underlying the Selective Knowledge Base $\mathcal{S}$ discussed in Section 2.

2. Framework

Let us assume that we have the Knowledge Graph (KG) $\mathcal{G}$ illustrated in Figure 1. This graph can be naturally encoded as the dataset:

$$\bar{D} = \{\mathsf{isa}(\mathsf{Epcot}, \mathsf{tp}), \mathsf{located}(\mathsf{Epcot}, \mathsf{Florida}), \mathsf{partOf}(\mathsf{Florida}, \mathsf{US}), \ldots\}.$$

Here, tp is a shorthand for theme_park and ap for amusement_park. Additionally, let us assume that we have an ontology $\bar{O}$ that specifies some intentional knowledge enriching the extensional knowledge already given by $\bar{D}$. As an example, we can consider the following ontology containing a single Datalog rule:

$$\bar{O} = \{\mathsf{isa}(x, z) \leftarrow \mathsf{isa}(x, y), \mathsf{isa}(y, z)\}.$$

By combining the dataset $\bar{D}$ and the ontology $\bar{O}$, we have the Knowledge Base (KB) $\bar{K} = (\bar{D}, \bar{O})$. As is customary, an atom α is *entailed* by some knowledge base K if it occurs in every model of K; accordingly, the set of atoms entailed by K is denoted by $ent(K)$. In our example, the set of entailed atoms is:

$$ent(\bar{K}) = \bar{D} \cup \{\mathsf{isa}(c_1, c_3) \,:\, \mathsf{isa}(c_1, c_2) \in \bar{D} \wedge \mathsf{isa}(c_2, c_3) \in \bar{D}\}.$$

Consider now a set U of n-ary tuples of entities, which can be referred to as an *anonymous relation* or simply as a *unit*. For example, we can choose the unit $\bar{U} = \{\langle \mathsf{Discovery\ Cove} \rangle, \langle \mathsf{Epcot} \rangle\}$ consisting of two unary tuples of entities. To express the nexus of similarity between the elements of U, it is indeed necessary to first establish a consensus on the relevant features or predicates describing any entity in D. Since such features or predicates might vary depending on the specific application scenario, we introduce the notion of summary selector.

Definition 1. *A* summary selector *is a computable function ς that takes as input a KB $K = (D, O)$ together with an n-ary tuple τ of entities from D, selects a subset S of $ent(K)$ containing at least all the entities in τ, and return S enriched with a* top atom *$\top(e)$ for each entity e in S.* ■

For the purposes of our example, let us adopt the simple yet effective selector $\bar{\varsigma}$ that builds, for each entity e in $\bar{D}$, the dataset $\varsigma(\bar{K}, \langle e \rangle)$ as the union of the following sets:

$$\begin{array}{rcl} A(e) & = & \{p(e',e'') \in ent(\bar{K}) \,:\, e' = e\}, \\ B(e) & = & \{p'(e',e'') \in ent(\bar{K}) \,:\, p(e,e') \in A(e) \wedge p \neq \mathsf{isa} \wedge p' \neq \mathsf{isa}\}, \\ C(e) & = & \{\top(e') \,:\, e' \text{ is an entity in } A(e) \cup B(e)\} \cup \{\top(e)\}. \end{array}$$

It is not difficult to see that ς satisfies Definition 1 and thus it is a summary selector. For instance, when $e = \mathsf{Discovery\_Cove}$, we have that

$$\begin{array}{l} A(e) = \{\mathsf{isa}(\mathsf{Discovery\_Cove}, \mathsf{tp}), \mathsf{located}(\mathsf{Discovery\_Cove}, \mathsf{Florida})\} \\ B(e) = \{\mathsf{partOf}(\mathsf{Florida}, \mathsf{US})\} \\ C(e) = \{\top(\mathsf{Discovery\_Cove}), \top(\mathsf{tp}), \top(\mathsf{Florida}), \top(\mathsf{US})\} \end{array}$$

Intuitively, in $A(e)$ we select all knowledge directly connected (at distance 1) to our entity; in $B(e)$ we select all knowledge connected at distance 2 with our entity, but without involving the relation isa; and in $C(e)$ we select all the necessary top atoms.

Definition 2. *A* selective knowledge base*, SKB for short, is a pair $\mathcal{S} = (K, \varsigma)$, where K is a knowledge base and ς is a summary selector.*

Our goal, now, is to express the nexus of similarity between the tuples of the given U with respect to the considered selective knowledge base $\mathcal{S}$. According to our running example that considers the unit $\bar{U}$ and the SKB $\bar{\mathcal{S}}$, by examining the next formula

$$\bar{\varphi}_1 = x \leftarrow \mathsf{isa}(x, \mathsf{ap}), \mathsf{located}(x, y), \mathsf{partOf}(y, \mathsf{US}),$$

where x is its free variable (also known as output variable), it is evident that $\bar{\varphi}_1$ explains some nexus of similarity between ⟨Discovery Cove⟩ and ⟨Epcot⟩. Indeed, formula $\bar{\varphi}_1$ says that x is an "amusement park located in some place y which, in turn, is part of US". However, $\bar{\varphi}_1$ neglects the additional information that both entities are also located in Florida according to their summaries. Indeed, for example, the following formula

$$\begin{array}{rcl} \bar{\varphi}_a = x & \leftarrow & \mathsf{isa}(x, \mathsf{tp}), \mathsf{isa}(x, \mathsf{ap}), \mathsf{located}(x, \mathsf{Florida}), \mathsf{partOf}(\mathsf{Florida}, \mathsf{US}), \\ & & \top(x), \top(\mathsf{ap}), \top(\mathsf{tp}), \top(\mathsf{Florida}), \top(\mathsf{US}) \end{array}$$

better explains the nexus of similarity between the two entities. For characterizing (all) the nexus of similarity between the tuples of $\bar{U}$, we have to both fix a suitable explanation language and formalize the notion of characterization.

An *(open conjunctive) formula* is an expression φ of the form

$$x_1, \ldots, x_n \leftarrow p_1(\mathbf{t}_1), \ldots, p_m(\mathbf{t}_m), \tag{1}$$

where $n > 0$ is its *arity*, $m > 0$ is its *size* often denoted by $|\varphi|$, each $\mathbf{t}_i$ is a sequence of terms, each $p_i(\mathbf{t}_i)$ is an atom, and each x_j is a variable —called *free*— occurring in some of the atoms of φ. A formula φ is *nearly connected* if each of its atoms is connected to some free variable of φ. For example, formula $\bar{\varphi}_1$ above is nearly connected. Indeed, both $\mathsf{isa}(x, \mathsf{ap})$ and $\mathsf{located}(x, y)$ are connected to the free variable x since they explicitly contain x; moreover, also $\mathsf{partOf}(y, \mathsf{US})$ is connected to x since it contains the variable y which occurs in an atom already marked as connected to x. Differently, formula $\bar{\varphi}_2 = x \leftarrow \mathsf{isa}(x, \mathsf{ap}), \mathsf{partOf}(y, \mathsf{US})$ is not nearly connected.

We utilize *nearly connected conjunctive formulas*, N$\mathbb{C}$F for short, as the formalism to explain the nexus of similarity between tuples within a unit, while considering their summaries.

As common in relational databases, a tuple $\langle t_1, \dots, t_n \rangle$ is an answer to φ over a dataset D if there exists a variable substitution that maps each x_i to t_i and each atom of φ to D. The *output* to φ over D is the set $\varphi(D)$ of all answers to φ over D. For example, $\bar{\varphi}_1(\bar{D})$ is the set $\{\langle \text{Pacific\_Park} \rangle\}$ and $\bar{\varphi}_1(ent(\bar{K}))$ is the set $\{\langle \text{Pacific\_Park} \rangle, \langle \text{Discovery\_Cove} \rangle, \langle \text{Epcot} \rangle\}$. Since we deal with summaries, the notion of output of a formula has to be refined. To this end, an *instance* of a formula φ according to some selective knowledge base $\mathcal{S} = (K, \varsigma)$ is any tuple τ which is an answer to φ over $\varsigma(K, \tau)$. Intuitively, if τ is not an answer to φ over its summary $\varsigma(K, \tau)$, then φ does not express properties of τ in terms of the considered scenario; if so, we consider τ not an instance of φ according to $\mathcal{S}$. The set of all φ-instances is denoted by $inst(\varphi, \mathcal{S})$.

Definition 3. *A nearly connected formula φ* characterizes *the nexus of similarity between the tuples of the unit U if both the next conditions hold:*

(*i*) *$inst(\varphi, \mathcal{S}) \supseteq U$;*

(*ii*) *for each nearly connected formula φ' such that $inst(\varphi', \mathcal{S}) \supseteq U$, it holds that $\varphi(D') \subseteq \varphi'(D')$ for every dataset D'.*

Accordingly, we may also say, for short, that φ characterizes U, that φ is a characterization for U, or that U is characterized by φ (with respect to $\mathcal{S}$). ■

It is now clear that formula $\bar{\varphi}_1$ above expresses some nexus of $\bar{U}$, but does not characterize it with respect to $\bar{\mathcal{S}}$. Conversely, $\bar{\varphi}_a$ characterizes $\bar{U}$. In the next section, we show how to construct a (canonical) characterization.

Essentially, having a canonical characterization is important because it shows directly two results, namely: a) a characterization always exists, which obviously is not an immediate result; b) such a characterization has a bound in its size with respect to the initial input and this bound is exponential with respect to the cardinality of the input unit, hence it becomes polynomial whenever you fix this parameter. The latter is particularly relevant, as often the sets of entities that we are interested in characterizing in the most diverse scenarios are formed from a few examples.

3. Canonical characterizations

We first recall and adapt to our purposes some well-known notions from database theory. Then, we show how to explicitly build a *canonical characterization* of the n-ary unit $U{=}\{\tau_1, \dots, \tau_m\}$ according to $\mathcal{S}{=}(K, \varsigma)$, called $can(U, \mathcal{S})$.

Definition 4. *Consider the n-ary tuples $\bar{\tau}_1, \dots, \bar{\tau}_\ell$. Their* direct product*, hereinafter denoted by $\bar{\tau}_1 \otimes \dots \otimes \bar{\tau}_\ell$, is the sequence $d_{\bar{\mathbf{s}}_1}, \dots, d_{\bar{\mathbf{s}}_n}$ of constants, where $\bar{\mathbf{s}}_i$ is the sequence $\bar{\tau}_1[i], \dots, \bar{\tau}_\ell[i]$, for each $i = 1, \dots, n$.*[1] *Accordingly, given k datasets $D_1, \dots, D_k$, their* direct product *is the dataset*

[1] The direct product of tuples is usually defined as a binary operation. Given two n-ary tuples, $\tau = \langle \tau[1], \dots, \tau[n] \rangle$ and

$$\{p(\langle c_1^1, \dots, c_1^n\rangle \otimes \dots \otimes \langle c_k^1, \dots, c_k^n\rangle) \;:\; p(c_1^1, \dots, c_1^n) \in D_1, \dots, p(c_k^1, \dots, c_k^n) \in D_k\}$$

hereinafter denoted by $D_1 \otimes \dots \otimes D_k$. ■

Let us illustrate these notions with a simple example.

Example 1. *Consider the following three binary tuples $\langle 1, 2\rangle$, $\langle 3, 4\rangle$, and $\langle 5, 6\rangle$. Hence, their direct product is the sequence*

$$\langle 1, 2\rangle \otimes \langle 3, 4\rangle \otimes \langle 5, 6\rangle = d_{1,3,5}, d_{2,4,6}.$$

Now, consider the following two datasets

$$D_1 = \{p(\mathrm{a}, \mathrm{c}), p(\mathrm{c}, \mathrm{e}), p(\mathrm{e}, \mathrm{b})\} \textit{ and } D_2 = \{p(\mathrm{a}, \mathrm{b}), p(\mathrm{b}, \mathrm{a}), r(\mathrm{b}, \mathrm{d})\}.$$

Then, their direct product is the dataset

$$D_1 \otimes D_2 = \{p(d_{\mathrm{a,a}}, d_{\mathrm{c,b}}),\ p(d_{\mathrm{a,b}}, d_{\mathrm{c,a}}),\ p(d_{\mathrm{c,a}}, d_{\mathrm{e,b}}),\ p(d_{\mathrm{c,b}}, d_{\mathrm{e,a}}),\ p(d_{\mathrm{e,a}}, d_{\mathrm{b,b}}),\ p(d_{\mathrm{e,b}}, d_{\mathrm{b,a}})\}.$$

Note that, since there is no element in D_1 capable of being in direct product with the element $r(\mathrm{b}, \mathrm{d})$ of D_2, the final set of atoms is devoid of the predicate r. ■

For a dataset D and an n-ary unit $U = \{\tau_1, \dots, \tau_m\}$, the direct product $P = D \otimes \dots \otimes D$ —multiplying D with itself m times— has been already used in database theory to check whether U admits a conjunctive query (i.e., a constant-free conjunctive formula) φ such that $\varphi(D) = U$ [14]. In particular, let $d_{\mathbf{s}_1}, \dots, d_{\mathbf{s}_n} = \tau_1 \otimes \dots \otimes \tau_m$, the query φ is the following formula

$$x_{\mathbf{s}_1}, \dots, x_{\mathbf{s}_n} \leftarrow \bigwedge_{p(t_1,\dots,t_k) \in P} p(\mu(t_1), \dots, \mu(t_k)),$$

where, for each constant t of the form $d_{\mathbf{s}}$ occurring in P, $\mu(t) = x_{\mathbf{s}}$. In case $K = (D, \emptyset)$ and $\varsigma(K, \tau) = D \cup \{\top(e) \;:\; e \text{ is an entity in } D\}$ for every τ, then there are cases in which φ would characterize U, but in general this is not guaranteed. The following examples precisely show where the classical direct product breaks and provide some useful insights on how the direct product could be enriched to correctly deal with nearly connected formulas and selective knowledge bases.

Example 2. *Let us start by considering the dataset*

$$D = \{r(1, 3), r(2, 4), r(5, 6), s(3, 5), p(4, 5)\},$$

which is graphically represented in Figure 2 as a directed graph. Let $U = \{\langle 1\rangle, \langle 2\rangle\}$. Accordingly, $D \otimes D$ is depicted in Figure 3. Since $\langle 1\rangle \otimes \langle 2\rangle = d_{1,2}$, the expected query φ is

$$\begin{aligned} x_{1,2} \;\leftarrow\;\; & r(x_{1,1}, x_{3,3}),\ r(x_{1,2}, x_{3,4}),\ r(x_{2,2}, x_{4,4}),\ r(x_{5,1}, x_{6,3}),\ r(x_{5,5}, x_{6,6}),\ r(x_{5,2}, x_{6,4}), \\ & r(x_{2,1}, x_{4,3}),\ r(x_{2,5}, x_{4,6}),\ r(x_{1,5}, x_{3,6}),\ s(x_{3,3}, x_{5,5}),\ p(x_{4,4}, x_{5,5}), \end{aligned}$$

$\tau' = \langle \tau'[1], \dots, \tau'[n]\rangle$, then $\tau \otimes \tau'$ is the n-ary tuple $\langle\langle \tau[1], \tau'[1]\rangle, \dots, \langle \tau[n], \tau'[n]\rangle\rangle$. Hence, this operation is associative up to isomorphisms, and it is possible to consider the direct product of more than two tuples [14, 15]. As we are not interested in a reiterate application of the direct product operator, for notational convenience, our direct product of two n-ary tuples is not an n-ary tuple, but just a sequence of fresh constants.

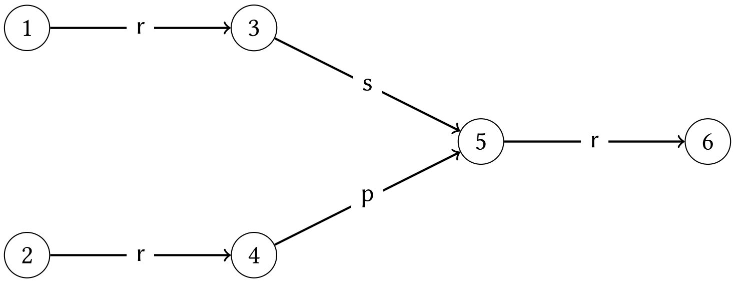

Figure 2: Graphical representation of dataset D presented in the Example 2.

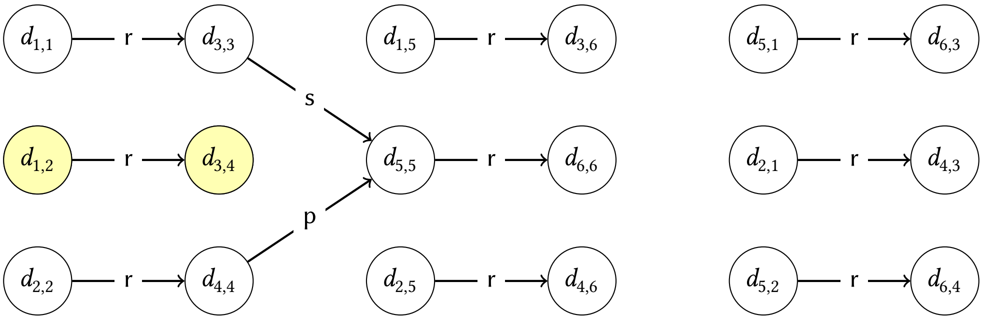

Figure 3: Graphical representation of dataset $D \otimes D$ presented in the Example 2.

whose atoms are isomorphic to the dataset depicted in Figure 3. Assume now that the given SKB $\mathcal{S} = (K, \varsigma)$ *is such that* $K = (D, \emptyset)$ *and* $\varsigma(K, \tau) = D \cup \{\top(e) \ : \ e \text{ is an entity in } D\}$ *for every* τ. *Clearly,* φ *is not a characterization for* U *with respect to* $\mathcal{S}$. *In this case, to obtain the following characterization for* U *with respect to* $\mathcal{S}$

$$x_{1,2} \quad \leftarrow \quad r(x_{1,2}, x_{3,4}),\ \top(x_{1,2}),\ \top(x_{3,4})$$

one can discard from φ *all the atoms that are not connected to* $x_{1,2}$ *and add a few* $\top$*-atoms. Note that, the only residual atom is* $r(x_{1,2}, x_{3,4})$, *which is isomorphic to* $r(d_{1,2}, d_{3,4})$ *—the one in yellow in Figure 3— which, in turn, is the only one connected to* $d_{1,2}$, *despite the fact that D is connected.*■

Unfortunately, discarding atoms not connected to free variables and adding extra $\top$-atoms is not enough to always yield to a characterization. This is illustrated via the following example.

Example 3. *Consider the SKB* $\mathcal{S} = (K, \varsigma)$, *where* $K = (D, \emptyset)$, $D = \{r(1,3), r(2,3), r(3,4)\}$, *and* $\varsigma(\tau) = D \cup \{\top(e) \ : \ e \text{ is an entity in } D\}$ *for every tuple* τ. *Let* $U = \{\langle 1 \rangle, \langle 2 \rangle\}$. *Accordingly, we have:*

$$\begin{aligned} D \otimes D \quad &= \quad \{r(d_{1,1}, d_{3,3}),\ r(d_{2,2}, d_{3,3}),\ r(d_{3,3}, d_{4,4}),\ r(d_{1,2}, d_{3,3}),\ r(d_{2,1}, d_{3,3}) \\ &\qquad r(d_{1,3}, d_{3,4}),\ r(d_{3,1}, d_{4,3}),\ r(d_{2,3}, d_{3,4}),\ r(d_{3,2}, d_{4,3})\}, \\ \varphi \ = \ x_{1,2} \quad &\leftarrow \quad r(x_{1,1}, x_{3,3}),\ r(x_{2,2}, x_{3,3}),\ r(x_{3,3}, x_{4,4}),\ r(x_{1,2}, x_{3,3}),\ r(x_{2,1}, x_{3,3}) \\ &\qquad r(x_{1,3}, x_{3,4}),\ r(x_{3,1}, x_{4,3}),\ r(x_{2,3}, x_{3,4}),\ r(x_{3,2}, x_{4,3})\}. \end{aligned}$$

By discarding from φ *all the atoms that are not connected to* $x_{1,2}$ *and adding the needed* $\top$*-atoms, we obtain*

$$\begin{array}{rcl}\varphi' = x_{1,2} & \leftarrow & r(x_{1,2}, x_{3,3}),\ r(x_{2,1}, x_{3,3}),\ r(x_{1,1}, x_{3,3}),\ r(x_{2,2}, x_{3,3}),\ r(x_{3,3}, x_{4,4}), \\ & & \top(x_{1,2}), \top(x_{3,3}), \top(x_{2,1}), \top(x_{1,1}), \top(x_{2,2}), \top(x_{4,4}).\end{array}$$

Even if φ' is nearly connected and inst$(\varphi', \mathcal{S}) \supseteq U$, formula φ' is still not a characterization in general. Indeed, it suffices to consider the following formula

$$\varphi'' = x \leftarrow r(x, 3), \top(x), \top(3)$$

and the dataset $D' = \{r(1, 1), \top(1)\}$. It is not difficult to see that φ' does not satisfy condition (ii) of Definition 3 since inst$(\varphi'', \mathcal{S}) \supseteq U$ but $\varphi'(D') \subseteq \varphi''(D')$ does not hold: 1 is an answer of φ' over D' but not an answer of φ'' over D'.

A possible turn around could be the following: replace in φ' any variable of the form $x_{c,\ldots,c}$ with the constant c. This would produce the following formula

$$\varphi''' = x_{1,2} \quad \leftarrow \quad r(x_{1,2}, 3),\ r(x_{2,1}, 3),\ r(1, 3),\ r(2, 3),\ r(3, 4), \top(x_{1,2}), \top(3), \top(x_{2,1}), \top(1), \top(2), \top(4),$$

which, this time, is a characterization for U. ■

Note, however, that in some cases a constant of the form $d_{c,\ldots,c}$ may give rise to a free variable $x_{c,\ldots,c}$. Clearly, if so, it cannot be replaced by c, otherwise the arity of the resulting formula would be smaller than the arity of the unit. Unfortunately, as shown by the following example, keeping the the variable $x_{c,\ldots,c}$ is not enough.[2]

Example 4. *Consider the SKB $\mathcal{S} = (K, \varsigma)$ together with the unit $U = \{\langle 1, 1\rangle, \langle 1, 2\rangle\}$, where $K = (D, \emptyset)$, $D = \{r(1, 2), r(2, 1), s(2, 1), s(1, 2)\}$, and ς is such that $\varsigma(\langle 1, 1\rangle) = \{r(1, 2), s(1, 2), \top(1), \top(2)\}$ and $\varsigma(\langle 1, 2\rangle) = \{r(1, 2), s(2, 1), \top(1), \top(2)\}$. In this case, by taking into account the summaries of the tuples of U, instead of computing $D \otimes D$ we compute $\varsigma(\langle 1, 1\rangle) \otimes \varsigma(\langle 1, 2\rangle)$. Hence, we obtain*

$$\begin{array}{rcl}P & = & \varsigma(\langle 1, 1\rangle) \otimes \varsigma(\langle 1, 2\rangle) = \{r(1, 2), s(1, 2), \top(1), \top(2)\} \otimes \{r(1, 2), s(2, 1), \top(1), \top(2)\} = \\ & = & \{r(d_{1,1}, d_{2,2}), s(d_{1,2}, d_{2,1}), \top(d_{1,1}), \top(d_{1,2}), \top(d_{2,1}), \top(d_{2,2})\}.\end{array}$$

Since the direct product between the tuples of U is $\langle 1, 1\rangle \otimes \langle 1, 2\rangle = d_{1,1}, d_{1,2}$, then the candidate characterization (constructed by using the approach discussed before) should be:

$$\varphi = x_{1,1}, x_{1,2} \quad \leftarrow \quad r(x_{1,1}, 2), s(x_{1,2}, x_{2,1}), \top(x_{1,1}), \top(x_{1,2}), \top(x_{2,1}), \top(2),$$

where only the constant $d_{2,2}$ is replaced by the constant 2. However, this is not a characterization. Indeed, it violates condition (ii) of Definition 3. To see that, consider the formula

$$\varphi' = x_{1,1}, x_{1,2} \quad \leftarrow \quad \top(x_{1,1}), \top(x_{1,2}), r(1, 2), \top(1), \top(2)$$

together with the dataset $D' = \{r(2, 2), s(2, 2), \top(2)\}$. ■

In light of the previous example, it seems that atoms containing constants of the form $d_{c,\ldots,c}$ occurring in the sequence $\tau_1 \otimes \ldots \otimes \tau_m$ should be "cloned" so that in some of them, $d_{c,\ldots,c}$ can be replaced by $x_{c,\ldots,c}$, while in some other, $d_{c,\ldots,c}$ can be replaced by c. We are now ready to show how to construct the canonical characterization $can(U, \mathcal{S})$.

[2] To lighten the notation, in what follows instead of writing $\varsigma(K, \tau)$, we will just write $\varsigma(\tau)$.

Step 1. Let $d_{\mathbf{s}_1}, \ldots, d_{\mathbf{s}_n} = \tau_1 \otimes \ldots \otimes \tau_m$ denote the sequence of constants used to determine the free variables of $can(U, \mathcal{S})$, and let Fr denote the set collecting these constants (note that, in general, $|Fr| \leq n$). According to Example 4, $\tau_1 = \langle 1, 1 \rangle$, $\tau_2 = \langle 1, 2 \rangle$, $d_{\mathbf{s}_1}, d_{\mathbf{s}_2} = d_{1,1}, d_{1,2}$, and Fr is the set $\{d_{1,1}, d_{1,2}\}$.

Step 2. Build the dataset $P = \varsigma(\tau_1) \otimes \ldots \otimes \varsigma(\tau_m)$ used to determine some atoms of $can(U, \mathcal{S})$. See, for instance, the set P of atoms constructed in Example 4. In particular, the domain of P is the set $\mathbb{D}_P = \{d_{1,1}, d_{1,2}, d_{2,2}, d_{2,1}\}$.

Step 3. Consider any $d_{\mathbf{s}}$ occurring in P, and let $\mathbb{D}_{\mathbf{s}}$ be the set of constants of $\mathbf{s}$. If $|\mathbb{D}_{\mathbf{s}}| = 1$, then the atoms of P containing $d_{\mathbf{s}}$ might have to be "cloned" to determine some extra atoms of $can(U, \mathcal{S})$. As we already discussed, this is needed whenever $d_{\mathbf{s}}$ satisfies both $|\mathbb{D}_{\mathbf{s}}| = 1$ and $d_{\mathbf{s}} \in Fr$. Accordingly, let $Ge = \{d_{\mathbf{s}} \in \mathbb{D}_P : |\mathbb{D}_{\mathbf{s}}| = 1\}$ be the set of constants used as possible "genes" for such clones. According to Example 4, $Ge = \{d_{1,1}, d_{2,2}\}$.

Step 4. For each $d_{\mathbf{s}} \in \mathbb{D}_P$, let

$$f(d_{\mathbf{s}}) = \begin{cases} \{d_{\mathbf{s}}, \mathbf{s}[1]\} & \text{if } d_{\mathbf{s}} \in Fr \cap Ge \\ \{d_{\mathbf{s}}\} & \text{otherwise.} \end{cases}$$

Now, for any atom $\alpha = p(t_1, \ldots, t_k)$ of P, we define

$$clones(\alpha) = \{p(c_1, \ldots, c_k) : each\ c_i \in f(t_i)\} \setminus \{\alpha\}$$

and, finally, let

$$C = \{\alpha' \in clones(\alpha) : \alpha \in P\}$$

be the set of all the clones that complement the atoms of P. According to Example 4, $Fr \cap Ge = \{d_{1,1}\}$. Moreover, $f(d_{1,1}) = \{d_{1,1}, 1\}$, $f(d_{1,2}) = \{d_{1,2}\}$, $f(d_{2,1}) = \{d_{2,1}\}$, and $f(d_{2,2}) = \{d_{2,2}\}$. Hence, $clones(\alpha) = \emptyset$, whenever $\alpha \in \{s(d_{1,2}, d_{2,1}), \top(d_{1,2}), \top(d_{2,1}), \top(d_{2,2})\}$; $clones(r(d_{1,1}, d_{2,2})) = \{r(1, d_{2,2})\}$; and $clones(\top(d_{1,1})) = \{\top(1)\}$. Finally, we have that $C = \{r(1, d_{2,2}), \top(1)\}$ and the domain of C is $\mathbb{D}_C = \{1, d_{2,2}\}$.

Step 5. Let μ be the mapping $\{c \mapsto c : c \in \mathbb{D}_C \setminus \mathbb{D}_P\} \cup \{d_{\mathbf{s}} \mapsto g(d_{\mathbf{s}}) : d_{\mathbf{s}} \in \mathbb{D}_P\}$ used to transform atoms of $P \cup C$ into atoms of $can(U, \mathcal{S})$, where

$$g(d_{\mathbf{s}}) = \begin{cases} x_{\mathbf{s}} & \text{if } d_{\mathbf{s}} \in Fr \\ y_{\mathbf{s}} & \text{if } d_{\mathbf{s}} \notin Fr \cup Ge \\ \mathbf{s}[1] & \text{if } d_{\mathbf{s}} \in Ge \setminus Fr. \end{cases}$$

Consider the next formula ("$\wedge$" is used instead of ","):

$$\Phi(U, \mathcal{S}) = x_{\mathbf{s}_1}, \ldots, x_{\mathbf{s}_n} \leftarrow \bigwedge_{p(t_1,\ldots,t_k) \in P \cup C} p(\mu(t_1), \ldots, \mu(t_k)).$$

According to Example 4, $Fr \cup Ge = \{d_{1,1}, d_{1,2}, d_{2,2}\}$ and $Ge \setminus Fr = \{d_{2,2}\}$. Hence, $g(d_{1,1}) = x_{1,1}$, $g(d_{1,2}) = x_{1,2}$, $g(d_{2,1}) = y_{2,1}$, and $g(d_{2,2}) = 2$. Moreover, $\mathbb{D}_C \setminus \mathbb{D}_P = \{1\}$ and, therefore, $\mu = \{1 \mapsto 1\} \cup \{d_{1,1} \mapsto x_{1,1}, d_{1,2} \mapsto x_{1,2}, d_{2,1} \mapsto y_{2,1}, d_{2,2} \mapsto 2\}$. Then, we have

$$P \cup C = \{r(d_{1,1}, d_{2,2}), s(d_{1,2}, d_{2,1}), \top(d_{1,1}), \top(d_{1,2}), \top(d_{2,1}), \top(d_{2,2}), r(1, d_{2,2}), \top(1)\}.$$

Finally, we obtain

$$\Phi(U, \mathcal{S}) = x_{1,1}, x_{1,2} \quad \leftarrow \quad r(x_{1,1}, 2), s(x_{1,2}, y_{2,1}), \top(x_{1,1}), \top(x_{1,2}), \top(y_{2,1}), \top(2), r(1, 2), \top(1).$$

Step 6. We are now ready to define the desired canonical characterization $can(U, \mathcal{S})$.

Definition 5. *We define $can(U, \mathcal{S})$ as the nearly connected formula obtained from $\Phi(U, \mathcal{S})$ by discarding all and only the atoms that are not connected to any free variable of $\Phi(U, \mathcal{S})$.We refer to $can(U, \mathcal{S})$ as the* canonical characterization *of U according to $\mathcal{S}$.*

According to Example 4, since $\Phi(U, \mathcal{S})$ is already a nearly connected formula, then we immediately get that $can(U, \mathcal{S}) = \Phi(U, \mathcal{S})$.

Theorem 1. *It holds that $can(U, \mathcal{S})$ characterizes U.*

For completeness of exposition, we close the section by showing how to systematically construct $can(\bar{U}, \bar{\mathcal{S}})$ according to our running example started in Section 2. The direct product of the elements of $\bar{U}$ is the (unary) sequence $\langle \mathsf{Epcot} \rangle \otimes \langle \mathsf{Discovery\_Cove} \rangle = d_{\mathsf{Epcot,Discovery\_Cove}}$. Hereinafter, to lighten the notation, we denote Discovery_Cove by D, Epcot by E, and Florida by F. Thus, $Fr = \{d_{\mathsf{E,D}}\}$. Now, to compute $P = \bar{\varsigma}(\langle \mathsf{Epcot} \rangle) \otimes \bar{\varsigma}(\langle \mathsf{Discovery\ Cove} \rangle)$, we need to exploit the summaries of $\langle \mathsf{Epcot} \rangle$ and $\langle \mathsf{Discovery\_Cove} \rangle$ already introduced in the previous section:

$$\begin{array}{rcl} \bar{\varsigma}(\langle \mathsf{E} \rangle) & = & \{\mathsf{located(E, F), partOf(F, US), isa(E, tp), isa(E, ap),} \\ & & \top(\mathsf{E}), \top(\mathsf{F}), \top(\mathsf{US}), \top(\mathsf{tp}), \top(\mathsf{ap})\} \\ \bar{\varsigma}(\langle \mathsf{D} \rangle) & = & \{\mathsf{located(D, F), partOf(F, US), isa(D, tp), isa(D, ap),} \\ & & \top(\mathsf{D}), \top(\mathsf{F}), \top(\mathsf{US}), \top(\mathsf{tp}), \top(\mathsf{ap})\} \end{array}$$

Therefore, the set P of atoms is:

$$\begin{array}{l} \{\mathsf{located}(d_{\mathsf{E,D}}, d_{\mathsf{F,F}}), \mathsf{partOf}(d_{\mathsf{F,F}}, d_{\mathsf{US,US}}), \mathsf{isa}(d_{\mathsf{E,D}}, d_{\mathsf{tp,tp}}), \mathsf{isa}(d_{\mathsf{E,D}}, d_{\mathsf{tp,ap}}), \mathsf{isa}(d_{\mathsf{E,D}}, d_{\mathsf{ap,tp}}), \\ \mathsf{isa}(d_{\mathsf{E,D}}, d_{\mathsf{ap,ap}}), \top(d_{\mathsf{E,D}}), \top(d_{\mathsf{E,F}}), \top(d_{\mathsf{E,US}}), \top(d_{\mathsf{E,tp}}), \top(d_{\mathsf{E,ap}}), \top(d_{\mathsf{F,D}}), \top(d_{\mathsf{F,F}}), \top(d_{\mathsf{F,US}}), \\ \top(d_{\mathsf{F,tp}}), \top(d_{\mathsf{F,ap}}), \top(d_{\mathsf{US,D}}), \top(d_{\mathsf{US,F}}), \top(d_{\mathsf{US,US}}), \top(d_{\mathsf{US,tp}}), \top(d_{\mathsf{US,ap}})\top(d_{\mathsf{tp,D}}), \top(d_{\mathsf{tp,F}}), \\ \top(d_{\mathsf{tp,US}}), \top(d_{\mathsf{tp,tp}}), \top(d_{\mathsf{tp,ap}}), \top(d_{\mathsf{ap,D}}), \top(d_{\mathsf{ap,F}}), \top(d_{\mathsf{ap,US}}), \top(d_{\mathsf{ap,tp}}), \top(d_{\mathsf{ap,ap}}), \}. \end{array}$$

Accordingly, $Ge = \{d_{\mathsf{F,F}}, d_{\mathsf{US,US}}, d_{\mathsf{tp,tp}}, d_{\mathsf{ap,ap}}\}$. This time, since $Fr \cap Ge = \emptyset$, we have that also the set C is empty. Therefore,

$$\begin{array}{rcl} \mu & = & \{d_{\mathsf{E,D}} \mapsto x_{\mathsf{E,D}}, d_{\mathsf{E,F}} \mapsto y_{\mathsf{E,F}}, d_{\mathsf{E,US}} \mapsto y_{\mathsf{E,US}}, d_{\mathsf{E,tp}} \mapsto y_{\mathsf{E,tp}}, d_{\mathsf{E,ap}} \mapsto y_{\mathsf{E,ap}}, \\ & & d_{\mathsf{F,D}} \mapsto y_{\mathsf{F,D}}, d_{\mathsf{F,F}} \mapsto \mathsf{F}, d_{\mathsf{F,US}} \mapsto y_{\mathsf{F,US}}, d_{\mathsf{F,tp}} \mapsto y_{\mathsf{F,tp}}, d_{\mathsf{F,ap}} \mapsto y_{\mathsf{F,ap}}, \\ & & d_{\mathsf{US,D}} \mapsto y_{\mathsf{US,D}}, d_{\mathsf{US,F}} \mapsto y_{\mathsf{US,F}}, d_{\mathsf{US,US}} \mapsto \mathsf{US}, d_{\mathsf{US,tp}} \mapsto y_{\mathsf{US,tp}}, d_{\mathsf{US,ap}} \mapsto y_{\mathsf{US,ap}}, \\ & & d_{\mathsf{tp,D}} \mapsto y_{\mathsf{tp,D}}, d_{\mathsf{tp,F}} \mapsto y_{\mathsf{tp,F}}, d_{\mathsf{tp,US}} \mapsto y_{\mathsf{tp,US}}, d_{\mathsf{tp,tp}} \mapsto \mathsf{tp}, d_{\mathsf{tp,ap}} \mapsto y_{\mathsf{tp,ap}}, \\ & & d_{\mathsf{ap,D}} \mapsto y_{\mathsf{ap,D}}, d_{\mathsf{ap,F}} \mapsto y_{\mathsf{ap,F}}, d_{\mathsf{ap,US}} \mapsto y_{\mathsf{ap,US}}, d_{\mathsf{ap,tp}} \mapsto y_{\mathsf{ap,tp}}, d_{\mathsf{ap,ap}} \mapsto \mathsf{ap}\}. \end{array}$$

Thus,

$$\begin{aligned}\Phi(\bar{U},\bar{\mathcal{S}}) = x_{\mathsf{E,D}} \quad \leftarrow \quad & \mathsf{located}(x_{\mathsf{E,D}},\mathsf{F}), \mathsf{partOf}(\mathsf{F},\mathsf{US}), \mathsf{isa}(x_{\mathsf{E,D}},\mathsf{tp}), \mathsf{isa}(x_{\mathsf{E,D}}, y_{\mathsf{tp,ap}}),\\ & \mathsf{isa}(x_{\mathsf{E,D}}, y_{\mathsf{ap,tp}}), \mathsf{isa}(x_{\mathsf{E,D}},\mathsf{ap}), \top(x_{\mathsf{E,D}}), \top(y_{\mathsf{E,F}}), \top(y_{\mathsf{E,US}}), \top(y_{\mathsf{E,tp}}),\\ & \top(y_{\mathsf{E,ap}}), \top(y_{\mathsf{F,D}}), \top(\mathsf{F}), \top(y_{\mathsf{F,US}}), \top(y_{\mathsf{F,tp}}), \top(y_{\mathsf{F,ap}}), \top(y_{\mathsf{US,D}}),\\ & \top(y_{\mathsf{US,F}}), \top(\mathsf{US}), \top(y_{\mathsf{US,tp}}), \top(y_{\mathsf{US,ap}}), \top(y_{\mathsf{tp,D}}), \top(y_{\mathsf{tp,F}}), \top(y_{\mathsf{tp,US}}),\\ & \top(\mathsf{tp}), \top(y_{\mathsf{tp,ap}}), \top(y_{\mathsf{ap,D}}), \top(y_{\mathsf{ap,F}}), \top(y_{\mathsf{ap,US}}), \top(y_{\mathsf{ap,tp}}), \top(\mathsf{ap}).\end{aligned}$$

Now, we can obtain the canonical characterization

$$\begin{aligned}\mathit{can}(\bar{U},\bar{\mathcal{S}}) = x_{\mathsf{E,D}} \quad \leftarrow \quad & \mathsf{located}(x_{\mathsf{E,D}},\mathsf{F}), \mathsf{partOf}(\mathsf{F},\mathsf{US}), \mathsf{isa}(x_{\mathsf{E,D}},\mathsf{tp}),\\ & \mathsf{isa}(x_{\mathsf{E,D}}, y_{\mathsf{tp,ap}}), \mathsf{isa}(x_{\mathsf{E,D}}, y_{\mathsf{ap,tp}}), \mathsf{isa}(x_{\mathsf{E,D}},\mathsf{ap}),\\ & \top(x_{\mathsf{E,D}}), \top(\mathsf{US}), \top(\mathsf{tp}), \top(y_{\mathsf{tp,ap}}), \top(\mathsf{F}), \top(y_{\mathsf{ap,tp}}), \top(\mathsf{ap}),\end{aligned}$$

by discarding all the atoms that are not connected to the free variable $x_{\mathsf{E,D}}$.

It is worth noting that, by the construction, we have not only proved that a characterization always exists, which is a non-trivial result, but also that, since its construction is clearly exponential in the cardinality of the input unit, whenever you fix this parameter you get something of polynomial size.

4. Related Work

There are some works that show how to construct *least general generalizations* in specific settings. We are going to discuss some key approaches using our terminology.

In [16], given a dataset D over a single ternary relation *triple* encoding an RDF graph, a unary unit U of resources, and a characteristic function ς returning, for any resource $\langle r\rangle \in U$, a set T_r of RDF triples connected to r (like our summaries which, however, are not necessarily connected), the authors show how to construct a generalized RDF graph (namely, an RDF graph with blank nodes) being connected and acting as the least common subsumer (LCS) of the resources in U (like our canonical characterizations). Hence, N$\mathbb{C}$Fs can be considered an extension of rooted RDF-graphs. Indeed, in [16], the formalism roughly coincides with unary (nearly) connected conjunctive formulas; here the authors adopt this notion to discard irrelevant (i.e., disconnected) properties. To guarantee the existence of LCSs, the authors consider only characteristic functions that return sets T_r containing at least one atom of the form $\mathit{triple}(r, p, o)$ for some p and o. Note that, differently from [16], we enforce summaries to be closed under $\top$. In case of unary units as considered by [16], one could avoid top atoms: whenever a characterization does not exist, then one might assume $x \leftarrow \top(x)$ as a default characterization. Conversely, in case of units of arbitrary arities, there are meaningful characterizations that would not exist if summaries are not closed under $\top$: for example, $x, y \leftarrow r(x), \top(x), \top(y)$ would not exist as a characterization but it is more informative than the default one, namely $x, y \leftarrow \top(x), \top(y)$.

Speaking of arbitrary arity, one of the key elements of our framework is precisely its non-limitation to being able to ask questions only in relation to unary units. The justification for this also lies, just to give an example, in wanting to study similarities between tuples of objects by finding so-called ”analogies” as in the case of the common ”a is to b as c is to d” [17]. In fact, one way to look at this in our setting is that an analogy between two tuples exists in our context when they have the same characterization when viewed as units. A concrete example of this is the one we gave in the introduction, where we referred to ⟨Tokyo, Tokyo Tower⟩ and ⟨Paris, Eiffel Tower⟩, which are fairly similar, as each is a "capital paired with one of its monuments being a tower made of metal", or as already stated in [17] the two tuples ⟨Leopard, Cat⟩ and ⟨Wolf, Dog⟩ in which essentially the first element is nothing other than a more ferocious species equipped with fangs who lives in a wild environment with respect to the second element. It is, however, important to note that while in the aforementioned work they give rise to real scores to determine whether an analogy exists or not we should turn to query answering, it will however be the aim of future work to also evaluate adequate metrics for the discovery of nexus of similarity.

In [18], given a KB K expressed in the $EL(I)$ description logic, a unary unit U with $|U| = 1$, and a summary selector always returning $ent(K)$, the authors study the problem of checking whether an $EL(I)$-characterization (called most specific concept or MSC) exists and verifying whether a given $EL(I)$ concept is an MSC. Moreover, the authors also study the variant of these problems where the input unit contains a set of $EL(I)$ concepts rather then entities, in order to study existence and verification of LCSs.

In [19], under arbitrary (union of) conjunctive queries (U)CQs, KBs with an empty (onto)logical part, and summary selectors always returning the entire dataset, the authors study existence, verification, and construction of characterizations (called fitting CQs).

Finally, it is worth noting that our N$\mathbb{C}$Fs can be considered an extension also of routed CQs [20]. This formalism essentially coincides with nearly connected conjunctive formulas without constants; here, the authors show that, without summaries, CQs and routed CQs are invariant with respect to their instances. However, in case summaries do not coincide with the whole $ent(K)$, CQs (resp., conjunctive formulas) and routed CQs (resp., N$\mathbb{C}$Fs) behave in different ways: in general, instances of characterizations are different.

5. Discussion and Conclusion

We start by examining some key design choices that have shaped our framework. After that, we will outline future directions for our research.

As already recognized by [16], when dealing with common properties, the use of summaries is rather crucial. Indeed, in our framework, avoiding the use of reasonably small summaries has the following negative effects: (a) characterizations would lose readability for humans; (b) nexus of similarity would not nicely fit the considered scenario; and (c) the direct product of two large datasets would be computationally unfeasible. In contrast, the ability to selectively exclude irrelevant features and predicates plays a crucial role in achieving meaningful characterizations. By focusing on the relevant information, we ensure that resulting characterizations are effective and tailored to the specific needs of the given application scenario.

N$\mathbb{C}$Fs naturally captures shared interconnected properties since they: (1) allow the inclusion of constants, which provide informative details; (2) allow for existential quantification (i.e., non-free variables), capturing connections beyond constants; (3) support conjunction and joins, as they inherently express connections between entities; (4) accommodate multiple free variables to go beyond unary concepts; (5) prevent disconnected components and disjunction, which go beyond semantic connections; (6) avoid forcing one connected component or acyclicity, as characterizations may not exist; (7) waive negation or universal quantification, as they inherently consider information beyond summaries; and (8) disallow built-in equality, which badly interacts with constants. Regarding the latter, consider the following SKB $\mathcal{S} = (K, \varsigma)$, where $K = (D, \emptyset)$, $D = \{r(a,a), r(a,b), r(b,a), r(b,b)\}$, $\varsigma(\langle a \rangle) = \varsigma(\langle b \rangle) = D \cup \{\top(e) \,:\, e \text{ occurs in } D\}$, and $U = \{\langle a \rangle\}$. Intuitively, a and b are indistinguishable as they "encounter" exactly the same constants; indeed, they are *transposable* (i.e., more than *automorphic*). If we allowed equality, then $\mathit{can}(U, \mathcal{S})$ would be of the form $x \leftarrow \ldots\ x = a$.

Future directions include: (*i*) a computational analysis of key reasoning tasks; (*ii*) further tuning NℂF; (*iii*) enriching summaries with intentional knowledge or anonymous individuals; for example, by incorporating rules such as $\mathsf{isa}(x, \mathsf{film}) \rightarrow \exists y \mathsf{directed}(x, y)$, films without known directors would share more nexus of similarities with films with known directors; (*iv*) dealing with entity set expansion; (*v*) designing and developing a prototype as a web service that implements the proposed framework on top of Linked Open Data; (*vi*) the definition of an appropriate metric to evaluate the nexus of similarity; and (*vii*) conducting experiments and evaluations to test and compare different summary selectors.

Acknowledgments

This work contributes to the basic research activities of the WP9.1: "KRR Frameworks for Green-aware AI" supported by the PNRR project FAIR - Future AI Research (PE00000013), Spoke 9 - Green-aware AI, under the NRRP MUR program funded by the NextGenerationEU.

References

[1] R. De Benedictis, N. Gatti, M. Maratea, A. Murano, E. Scala, L. Serafini, I. Serina, E. Tosello, A. Umbrico, M. Vallati, Preface to the Italian Workshop on Planning and Scheduling, RCRA Workshop on Experimental evaluation of algorithms for solving problems with combinatorial explosion, and SPIRIT Workshop on Strategies, Prediction, Interaction, and Reasoning in Italy (IPS-RCRA-SPIRIT 2023), in: Proceedings of the Italian Workshop on Planning and Scheduling, RCRA Workshop on Experimental evaluation of algorithms for solving problems with combinatorial explosion, and SPIRIT Workshop on Strategies, Prediction, Interaction, and Reasoning in Italy (IPS-RCRA-SPIRIT 2023) co-located with 22th International Conference of the Italian Association for Artificial Intelligence (AI* IA 2023), 2023.

[2] W. Gomaa, A. Fahmy, A survey of text similarity approaches, International Journal of Computer Applications 68 (2013) 13–18.

[3] D. Chandrasekaran, V. Mago, Evolution of semantic similarity - A survey, ACM Comput. Surv. 54 (2022) 41:1–41:37. URL: https://doi.org/10.1145/3440755. doi:10.1145/3440755 .

[4] J. Cirasella, Google sets, google suggest, and google search history: Three more tools for the reference librarians bag of tricks, The Reference Librarian 48 (2007). URL: http: //ref.haworthpress.com.

[5] P. Pantel, E. Crestan, A. Borkovsky, A. Popescu, V. Vyas, Web-scale distributional similarity and entity set expansion, in: Proceedings of the 2009 Conference on Empirical Methods

in Natural Language Processing, EMNLP 2009, 6-7 August 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL, ACL, 2009, pp. 938–947. URL: https://aclanthology.org/D09-1098/.

[6] R. Blanco, B. B. Cambazoglu, P. Mika, N. Torzec, Entity recommendations in web search, in: H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira, L. Aroyo, N. F. Noy, C. Welty, K. Janowicz (Eds.), The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II, volume 8219 of *Lecture Notes in Computer Science*, Springer, 2013, pp. 33–48. URL: https://doi.org/10.1007/978-3-642-41338-4_3. doi:10.1007/978-3-642-41338-4_3.

[7] N. A. S. Er, T. Abdessalem, S. Bressan, Set of t-uples expansion by example, in: G. Anderst-Kotsis (Ed.), Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services, iiWAS 2016, Singapore, November 28-30, 2016, ACM, 2016, pp. 221–230. URL: https://doi.org/10.1145/3011141.3011144. doi:10.1145/3011141.3011144.

[8] Y. Zhang, Y. Xiao, S. Hwang, H. Wang, X. S. Wang, W. Wang, Entity suggestion with conceptual expanation, in: C. Sierra (Ed.), Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, ijcai.org, 2017, pp. 4244–4250. URL: https://doi.org/10.24963/ijcai.2017/593. doi:10.24963/ijcai.2017/593.

[9] G. Xun, Y. Li, W. X. Zhao, J. Gao, A. Zhang, A correlated topic model using word embeddings, in: C. Sierra (Ed.), Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, ijcai.org, 2017, pp. 4207–4213. URL: https://doi.org/10.24963/ijcai.2017/588. doi:10.24963/ijcai.2017/588.

[10] J. Huang, W. Zhang, Y. Sun, H. Wang, T. Liu, Improving entity recommendation with search log and multi-task learning, in: J. Lang (Ed.), Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden, ijcai.org, 2018, pp. 4107–4114. URL: https://doi.org/10.24963/ijcai.2018/571. doi:10.24963/ijcai.2018/571.

[11] J. Chen, Y. Chen, X. Zhang, X. Du, K. Wang, J. Wen, Entity set expansion with semantic features of knowledge graphs, J. Web Semant. 52-53 (2018) 33–44. URL: https://doi.org/10.1016/j.websem.2018.09.001. doi:10.1016/j.websem.2018.09.001.

[12] M. Lissandrini, D. Mottin, T. Palpanas, Y. Velegrakis, Graph-query suggestions for knowledge graph exploration, in: Y. Huang, I. King, T. Liu, M. van Steen (Eds.), WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020, ACM / IW3C2, 2020, pp. 2549–2555. URL: https://doi.org/10.1145/3366423.3380005. doi:10.1145/3366423.3380005.

[13] H. Ma, Y. Ke, An introduction to entity recommendation and understanding, in: A. Gangemi, S. Leonardi, A. Panconesi (Eds.), Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015, Florence, Italy, May 18-22, 2015 - Companion Volume, ACM, 2015, pp. 1521–1522. URL: https://doi.org/10.1145/2740908.2741991. doi:10.1145/2740908.2741991.

[14] P. Barceló, M. Romero, The complexity of reverse engineering problems for conjunctive queries, in: M. Benedikt, G. Orsi (Eds.), 20th International Conference on Database Theory, ICDT 2017, March 21-24, 2017, Venice, Italy, volume 68 of *LIPIcs*, Schloss Dagstuhl - Leibniz-

Zentrum für Informatik, 2017, pp. 7:1–7:17. URL: https://doi.org/10.4230/LIPIcs.ICDT.2017.7. doi:10.4230/LIPIcs.ICDT.2017.7.

[15] B. ten Cate, V. Dalmau, The product homomorphism problem and applications, in: M. Arenas, M. Ugarte (Eds.), 18th International Conference on Database Theory, ICDT 2015, March 23-27, 2015, Brussels, Belgium, volume 31 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015, pp. 161–176. URL: https://doi.org/10.4230/LIPIcs.ICDT.2015.161. doi:10.4230/LIPIcs.ICDT.2015.161.

[16] S. Colucci, F. M. Donini, S. Giannini, E. D. Sciascio, Defining and computing least common subsumers in RDF, J. Web Semant. 39 (2016) 62–80. URL: https://doi.org/10.1016/j.websem.2016.02.001. doi:10.1016/j.websem.2016.02.001.

[17] H. Prade, G. Richard, Analogical proportions: From equality to inequality, Int. J. Approx. Reason. 101 (2018) 234–254. URL: https://doi.org/10.1016/j.ijar.2018.07.005. doi:10.1016/j.ijar.2018.07.005.

[18] J. C. Jung, C. Lutz, F. Wolter, Least general generalizations in description logic: Verification and existence, in: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, AAAI Press, 2020, pp. 2854–2861. URL: https://ojs.aaai.org/index.php/AAAI/article/view/5675.

[19] B. ten Cate, V. Dalmau, M. Funk, C. Lutz, Extremal fitting problems for conjunctive queries, in: F. Geerts, H. Q. Ngo, S. Sintos (Eds.), Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2023, Seattle, WA, USA, June 18-23, 2023, ACM, 2023, pp. 89–98. URL: https://doi.org/10.1145/3584372.3588655. doi:10.1145/3584372.3588655.

[20] J. C. Jung, C. Lutz, H. Pulcini, F. Wolter, Logical separability of labeled data examples under ontologies, Artif. Intell. 313 (2022) 103785. URL: https://doi.org/10.1016/j.artint.2022.103785. doi:10.1016/j.artint.2022.103785.