Journal of Applied Logic 12 (2014) 67–74

Contents lists available at SciVerse ScienceDirect

Journal of Applied Logic

www.elsevier.com/locate/jal

Formal Ontologies and Coherent Spaces

V. Michele Abrusci [a,*], Christophe Fouqueré [b], Marco Romano [a]

[a] Dipartimento di Filosofia, Università Roma Tre, via Ostiense 234, 00144 Roma, Italy
[b] LIPN, Université Paris 13, 99 avenue Jean-Baptiste Clément, 93430 Villetaneuse, France

ARTICLE INFO

Article history:
Available online 25 July 2013

Keywords:
Ontologies
Linear Logic
Coherent Spaces

ABSTRACT

The paper contains a short summary – oriented by a logical point of view – of a joint work on Formal Ontologies. We shall show how Formal Ontologies correspond to Coherent Spaces, and operations on Formal Ontologies correspond to operations on corresponding Coherent Spaces. So, we are offering a new way to establish the semantics of Formal Ontologies. Surely, we are giving a contribution towards a *geometrical* treatment of Formal Ontologies (as decidable organizations of digital data).

1. Introduction

Firstly, in the section called *Formal Ontologies*, we describe what are Formal Ontologies, and what are operations and relations on Formal Ontologies, by using a description which is influenced by philosophical and logical considerations and allows further developments of our paper.

Secondly, in the section called *Formal Ontologies as Coherent Spaces* we show how each Formal Ontology induces a Coherent Space (a nice geometrical concept introduced in the proof-theoretical investigations of Linear Logic to modelize the logical proofs, as explained by Jean-Yves Girard in the papers [3], [4] and [5]) and may be identified with this Coherent Space, in such a way that to operations on Formal Ontologies correspond natural operations on Coherent Spaces (the operations which in the proof-theoretical investigations of Linear Logic modelize the logical connectives).

Finally, we put some questions to be discussed through the future development of this new geometrical approach to Formal Ontologies.

2. Formal Ontologies

2.1. Basic

In the traditional philosophy, *Ontology* is the science of the entities, the science describing also the organization of the entities. In some aspect, Set Theory and Category Theory may be considered as belonging to the Ontology of the mathematical entities.

Formal Ontologies and *General Formal Ontologies* (called also *Terminologies*) are used in Computer Science (see [6]).

A *Formal Ontology* aims to organize a domain of digital data, with two peculiar features:

- the organization must be performed by means of concepts (i.e. properties) and relations (called also roles);
- the organization must be performed in a decidable way.

* Corresponding author.
E-mail address: abrusci@uniroma3.it (V.M. Abrusci).

1570-8683/$ – see front matter
http://dx.doi.org/10.1016/j.jal.2013.07.003

A *General Formal Ontology* is a general framework for potential Formal Ontologies, so that each General Formal Ontology, when it is applied to a specific domain of digital data, produces a Formal Ontology for that domain.

Two Formal Ontologies are *disjoint* when they concern disjoint domains of digital data (and so they do not share concepts or relations).

Usually, Formal Ontologies are presented as *first-order theories*; so that each Formal Ontology is given by a first-order language and by an inferential engine producing a derivability mechanism which may be represented as a derivability predicate.

Thus, the language of a Formal Ontology must contain names for the digital data to be organized, and names for concepts and relations to be used in the organization, by considering concepts as unary predicates and relations as binary predicates; and the inferential engine, the derivability predicate of a Formal Ontology, must be a recursive predicate.

Indeed, a Formal Ontology O is a first-order theory such that

- the set $\underline{Al}(O)$ of the specific symbols of the language of O contains:
  - individual constants (an individual constant for each element of the domain of the digital data to be organized); the set of these individual constants will be denoted by $\underline{Ind}(O)$;
  - unary predicate symbols (an unary predicate symbol for each concept to be used in the organization of digital data);
  - binary predicate symbols (a binary predicate symbol for each relation to be used in the organization of digital data);
- the language of O, $\underline{L}(O)$, is the first-order language built from the set $\underline{Al}(O)$;
- the set of the axioms of O, $\underline{Th}(O)$, is the union of two disjoint sets of axioms:
  - the set $\underline{T-box}(O)$ whose elements are axioms without individual constants;
  - the set $\underline{A-box}(O)$ whose elements are atomic axioms with individual constants;
- the predicate “to be a formula derivable in O”, $\vdash_O$, is a recursive predicate, and thus also its negation $\nvdash_O$ is a recursive predicate.

If O is a Formal Ontology, then we may consider the set of all the objects on which O attributes some concept or some relation, i.e.

$$\underline{Ob}(O) = \{a\colon a \in \underline{Ind}(O) \wedge \exists P \vdash_O P(a)\} \cup \{\langle a, b\rangle\colon a \in \underline{Ind}(O) \wedge b \in \underline{Ind}(O) \wedge \exists R \vdash_O R(a, b)\}.$$

Usually, $\underline{Ind}(O) \subseteq \underline{Ob}(O)$.

When Formal Ontologies are considered as first-order theories as defined above, then we get the corresponding definitions of the notions “disjoint Formal Ontologies”, “Terminology (or General Formal Terminology)”, “Formal Ontology obtained from a Terminology”:

- two Formal Ontologies O_1 and O_2 are disjoint iff $\underline{Al}(O_1) \cap \underline{Al}(O_2) = \emptyset$;
- a General Formal Ontology, or a Terminology, is a Formal Ontology where $\underline{Ind}(O) = \emptyset$ and so $\underline{A-box}(O) = \emptyset$ and $\underline{Ob}(O) = \emptyset$;
- given a General Formal Ontology O, and a set X of digital data, each application of O to X produces a Formal Ontology O' such that:
  - $\underline{Al}(O') = \underline{Al}(O) \cup \underline{Ind}(O')$, where $\underline{Ind}(O')$ contains an individual constant for each element of X,
  - $\underline{L}(O')$ is the first-order language built from $\underline{Al}(O')$, so that $\underline{L}(O) \subseteq \underline{L}(O')$,
  - $\underline{A-box}(O')$ is a set of atomic formulas of $\underline{L}(O')$ containing individual constants;
  - $\underline{T-box}(O') = \underline{T-box}(O)$.

Classical logic is used in both the syntax and semantics of Formal Ontologies. But this use of classical logic leads to several problems among those the following ones:

- a “model” of a Formal Ontology must be a model where the objects are exactly the digital data represented by the individual constants, but a set of digital data cannot be reduced to a perennial set of objects without relations between them, as required in the classical logic view of “set”,
- classical logic forces us to see concepts (represented by unary predicate symbols) and relations (represented by binary predicate symbols) in a way very far from the natural way to see concepts concerning digital data and relations between digital data.

The common ways to overcome these problems (e.g. those based on modal logic) are rather artificial and in contrast with a good philosophical view of the relation between logic and the “external“ world (real world, or digital data).

2.2. Operations and relations on Formal Ontologies

In the traditional philosophy, there is no coexistence of several Ontologies (each proposed Ontology wants to replace the other ones) and no interaction between Ontologies; only one Ontology may exist.