A Model of Information Retrieval based on a Terminological Logic*

Carlo Meghini, Fabrizio Sebastiani, Umberto Straccia and Costantino Thanos
Istituto di Elaborazione dell'Informazione
Consiglio Nazionale delle Ricerche
Via S. Maria, 46 - 56126 Pisa (Italy)
E-mail: ⟨*lastname*⟩@iei.pi.cnr.it

Abstract

According to the *logical model* of Information Retrieval (IR), the task of IR can be described as the extraction, from a given document base, of those documents d that, given a query q, make the formula $d \rightarrow q$ valid, where d and q are formulae of the chosen logic and "$\rightarrow$" denotes the brand of logical implication formalized by the logic in question. In this paper, although essentially subscribing to this view, we propose that the logic to be chosen for this endeavour be a *Terminological Logic* (TL): accordingly, the IR task becomes that of singling out those documents d such that $d \preceq q$, where d and q are *terms* of the chosen TL and "$\preceq$" denotes *subsumption* between terms. We call this the *terminological model* of IR.

TLs are particularly suitable for modelling IR; in fact, they can be employed: 1) in representing documents under a variety of aspects (e.g. structural, layout, semantic content); 2) in representing queries; 3) in representing lexical, "thesaural" knowledge. The fact that a single logical language can be used for all these representational endeavours ensures that all these sources of knowledge will participate in the retrieval process in a uniform and principled way.

In this paper we introduce MIRTL, a TL for modelling IR according to the above guidelines; its syntax, formal semantics and inferential algorithm are described.

*This paper appears in the Proceedings of ACM SIGIR-93, 16th International Conference on Research and Development in Information Retrieval, Pittsburgh, PA, 1993, pages 298–307.

1 Introduction

According to the *logical model* of Information Retrieval (IR) [21, 22], the task of IR can be described as the extraction, from a given document base, of those documents d that, given a query q, make the formula $d \rightarrow q$ valid, where d and q are formulae of the chosen logic and "$\rightarrow$" denotes the brand of logical implication formalized by the logic in question.

This analysis, however, leaves open the problem of how to represent documents and queries in a way that, while being amenable to a formalization in the style of mathematical logic, be at the same time adequate for accounting for the complexity of real documents and queries, especially when multimedia documents are considered.

In fact, it has been argued (see e.g. [11]) that a radical improvement in the performance of IR systems may only be accomplished by endowing these systems with high level representations of the documents contained in the document base.

In order to be effective, these representations should consider documents from multiple viewpoints. For instance, the representations should not be confined to describing the content of the document by means of a simple set of keywords, as the connection of these to the real nature of the document is often loose at best. Instead, the representation of a document should also address its internal architecture (usually known as *structure*), its physical appearance (*layout*), and give a better account of what the document actually deals with (*semantic content*). It is then evident that the language needed for expressing such multifaceted representations should be much more expressive and richly structured than the language of simple statements adopted by (either the classical or the non-classical variants of) propositional logic and considered in [16, 21, 22].

This latter language is also inadequate in another important respect. The basic linguistic unit of propositional logic is the statement, an expression whose denotation in a given interpretation is a truth value. Instead, we feel that the description of a document should not denote a truth value at all, but an object of the domain of discourse: the document itself. In the same way, we feel that a query should not denote a truth value, but a set of objects of the domain of discourse: the set of documents of the document base that the user would like

the system to retrieve. In summary, we feel that, in order for the representation to be adequate, the symbols d and q should *not* be statements (i.e. expressions denoting truth values), but *terms* (i.e. expressions denoting objects or sets of objects); accordingly, $d \rightarrow q$ should not stand for "statement d logically implies statement q", but for "term d is an instance of (or: is less general than) term q"[1]. This implies that a logic which is adequate for the IR modelling task should allow the formulation of this latter kind of expressions, something that is clearly outside the domain of (classical or non-classical) propositional logic.

Besides, the fact that documents are *complex and multifaceted* objects (especially when multimediality is at stake), adds the constraint that the terms of the logic we are seeking should have a *complex internal structure*. Chiaramella and Chevallet [5, page 233] are explicit on this:

> (...) we think that the more complex the stored information is in its structure and content, various in its nature (text, images, etc.), flexible in its interrelations (related texts, images), the more the underlying semantics of this complexity and variety has to be made accessible, or explicit, to the users in order to allow them to retrieve properly and exploit this information.

In this paper, although "essentially" subscribing to the "$d \rightarrow q$ view" put forth in [21, 22], we propose that the logic for the description of both documents and queries be a *Terminological Logic* (TL); as we will show, TLs comply with all the desiderata listed above. We call the resulting model the *terminological model* of IR.

Terminological Logics are formalisms developed within artificial intelligence research, and specifically oriented to the vast class of application domains that are describable by means of taxonomic organizations of classes of structured objects. We will argue that their "term-oriented" syntax, the existence of a wide corpus of theoretical results concerning them, and their mostly positive computational properties, make them interesting tools for IR-related modelling tasks: in fact, we will show that these logics not only can be employed for both document and query representation, but they can be profitably used for the representation of lexical, "thesaural" knowledge, i.e. of knowledge supporting the process of interpretation "in context" of both documents and queries.

The fact that all three sources of knowledge (*viz.* representation of the document, of the query and of the thesaural knowledge) can be expressed in the same formalism ensures that they will participate in the retrieval process in a uniform fashion and as an undistinguished whole. The fact that this formalism is a *logic* (instead of being e.g. a non-logically-specified representation language) is a "certificate of guarantee", ensuring that this participation will also be *principled*, i.e. that the inferences performed by the underlying implementation will fully comply with the semantics of the representation language.

[1] This point is argued in full detail in an extended version of this paper [13].

The rest of this paper is organized as follows. In order to make the paper self-contained, in Section 2 we give a brief, informal overview of the fundamentals of TLs and of how knowledge on taxonomically structured domains may be conveyed in terms of them. In Section 3 we describe (still at a fairly informal level) the terminological model of IR: starting from the observation that a taxonomical structure can be discerned in document bases, we describe how a TL might profitably be used in an IR context, and to this purpose we give sample representations of simple documents, queries and items of lexical knowledge. In Section 4 we describe the terminological model in full formal detail: in order to do this, we introduce MIRTL, a terminological logic for information retrieval[2]: we give the formal syntax and semantics of MIRTL, and sketch how a sound and complete algorithm for performing inferences on MIRTL may be built in a modular, straightforward way. Section 5 takes a look at possible further developments of this line of research.

2 An informal introduction to Terminological Logics

Terminological Logics have their roots in artificial intelligence research and, in particular, in the subfield of AI known as "knowledge representation"; TLs derive, in fact, from a large class of early knowledge representation languages (of which KL-ONE [3] may be considered the founding father) based on *semantic networks* and inspired by the notion of *frame*, originally introduced by Minsky [14] in the context of cognitive science applications.

Quite apart from AI domains, these early languages have also been applied in conceptual data modelling (see e.g. [10, 18]) and conceptual document modelling [11]. Unfortunately these languages did not possess a formal semantics; hence, the fact that the notion of inference they enforced were the same as the user expected, relied on an improbable like-mindedness among designer, implementor and user. TLs may then be seen as the result of a work of distillation and formalization that was carried out on KL-ONE-like languages, a work that resulted in the purge of linguistic primitives of dubious semantic content and the recasting of the others in terms of mathematical logic.

We have previously said that TLs have *terms* as their primary syntactic expressions. In TLs a term is usually an expression that denotes a monadic or dyadic relation on the domain of discourse; terms representing monadic relations are called *concepts*, while terms representing dyadic relations are called *roles*. For example, in a TL that includes the term-forming operators **and**, **atleast** and **atmost**[3], and whose alphabet contains the monadic predicate symbol `paper` and the dyadic predicate symbol

[2] MIRTL stands for Multimedia Information Retrieval Terminological Logic; the acronym reflects our strong belief that the "philosophy" underlying this model of IR will extend smoothly to the multimedia case.

[3] For an informal explanation of the meaning of the term-forming operators involved in the examples that follow, the reader may peek ahead in Section 4.1.

`author`, the expression

(**and** `paper` (**atmost** `3 author`)
(**atleast** `3 author`))

is a concept that, under the obvious interpretation, denotes the set of all those papers that have exactly three authors. In general, the languages of TLs allow for a number of term-forming operators by means of which one may build complex terms starting from a basic repertory of simple terms (*viz.* predicate symbols).

Many TLs also allow the "definition" of a predicate symbol by means of a term. For example, one such definition is the following, where one states that triangles are precisely those polygons that have exactly three sides.

`triangle` $\doteq$ (**and** `polygon` (**atleast** `3 side`)
(**atmost** `3 side`))

This definition lists a set of necessary and sufficient conditions that an individual should satisfy in order to be recognized as a triangle. It is however useful (especially when trying to give "incomplete" descriptions of classes) to be able to specify lists of conditions that are only necessary; in TLs this is expressed by what might be called a "connotation", as the following example shows:

`dog` $<$ (**and** `animal` (**exactly** `4 leg`))

This expression states that dogs are four-legged animals, but does *not* not state that all four-legged animals are dogs (we assume here that **exactly** is an operator defined in terms of **atmost** and **atleast**.).

Most TLs also allow for "instance assertions", by means of which one can assert that a given individual constant is an instance of a given concept (or that a pair of individual constants is an instance of a given role)[4]. Supposing that the alphabet of individual constants includes the symbols `Fido`, `John` and `paper2501`, the following are examples of instance assertions:

(**and** `animal` (**exactly** `4 leg`)) [`Fido`]

`dog` [`Fido`]

`author` [`John`,`paper2501`]

We have remarked before that terms, the primary syntactic expressions of TLs, are not truth-value carriers; hence, unlike logics of statements such as propositional logic, the metalinguistic relation of interest to TLs is not validity: only formulae can be valid or invalid, terms cannot. Usually, *subsumption* (i.e. "conceptual containment") between terms is considered the most important metalinguistic notion in TLs, as the decision problem for most other relations of interest to TLs can be reduced to subsumption checking. A theorem prover for a TL is thus a program that, given two terms, decides whether the former subsumes the latter, i.e. whether all individuals denoted by the latter are also denoted by the former. For example, such a theorem prover will be able to answer affirmatively to the question whether the term

(**and** `polygon` (**atmost** `4 sides`))

(which denotes the set of all triangles and quadrangles) subsumes

(**and** `polygon` (**exactly** `4 sides`))

This is an example of what is often called *implicit* subsumption: the former term subsumes the latter just because of the structure of the two terms and of the semantics of the term-forming operators involved. When definitions and/or connotations (also globally called *terminological axioms*) are involved, one usually speaks of *explicit* subsumption[5]. For example, given the above definition of the term `triangle`, the term

(**and** `triangle regular-polygon`)

(denoting the set of regular triangles) will (explicitly) be subsumed by the term

(**and** `polygon` (**atmost** `4 sides`))

A "terminology" (i.e. a set of terminological axioms) transforms the set of predicate symbols into a *partially ordered set* (or *poset*), with the partial order coinciding with the relation of (explicit) subsumption.

Of course, the applicability of TLs to problems of realistic size largely depends on the expressive power of the TLs themselves, and on the possibility of specifying theorem provers that, besides being correct and complete with respect to the TL in question, are able to decide subsumption *efficiently*. A thorough understanding of TLs requires then a rigorous and systematic study of the relationship between their expressive power and their computational complexity.

From the point of view of expressive power, one may see a TL as formed by three different modules:

1. a *terminological module* (TM), consisting of the operators for forming complex terms;
2. a *definitional module* (DM), consisting of the "$\doteq$" and "$<$" operators for defining and connoting simple terms (i.e. predicate symbols)
3. an *assertional module* (AM), consisting of the linguistic primitives for expressing instance assertions.

The DM and AM are actually identical for all TLs (although a given TL may lack one or both of them); the TM usually varies among different TLs, and is what most characterizes a given TL, to the extent that each TL is usually named after its TM.

In order to define what the expressive power of TMs is, let us consider a "reference" set Σ of all the term-forming operators that we deem of significant applicative value, and let us define a TM to be a subset of Σ[6]. The relationship "has less expressive power than" may now be

[4] Some authors (see e.g. [7, 19]) call "terminological" only those TLs that do not allow instance assertions; these authors call "hybrid logics" those TLs that do allow such assertions.

[5] Throughout this paper we will assume that terminological axioms are *acyclic*, i.e. that their left-hand side does not occur, either directly or indirectly, in their right-hand side. This is a standard simplifying assumption for TLs; see e.g. [15] for a framework that does away with this limitation.

[6] For an attempt at giving a more general and rigorous definition of "expressive power of a TM", see the work by Baader [2].

defined as the "$\subseteq$" partial order defined on the powerset of Σ[7].

It is well known that there is a tradeoff between the expressive power of a logic and its computational tractability; hence, in general, the relation "has less expressive power than" coincides with the relation "is decidable at least as efficiently as". One of the fundamental problems of the research on TMs is that of individuating the TMs which are "optimal" from the standpoint of this tradeoff, that is, individuating those logics which are most expressive among the ones for which a polynomial decision algorithm can be given; these logics can then be safely and profitably used in computationally demanding applications.

A number of results relative to the computational complexity of subsumption checking in TMs have recently appeared in the literature (see e.g. [20]). For our purposes, it will suffice to say that some of these results are encouraging, in the sense that for some (reasonable) choice of Σ, optimal logics have been found [6] that have also a fair amount of expressive power[8].

3 The Terminological Model of IR. An informal introduction.

Up to now, we have given an informal account of what a TL is and what are the main problems one encounters in choosing the right TL for the application at hand. We now go on to give an overview of how we plan to use a TL in an IR modelling context, yielding the terminological model of IR.

The terminological model is composed of:

1. *a model for documents*: a document will be represented as an individual constant of the chosen TL; this constant will be the subject of a number of instance assertions; the concepts of which the constant is asserted to be an instance will then altogether constitute the description of the document;
2. *a model for queries*: a query will be represented as a concept of the chosen TL; the intuitive meaning of this choice is that all documents represented by individual constants that are recognized to be instances of this concept should be retrieved;
3. *a model for lexical entries*: lexical knowledge will be represented by means of a set of terminological axioms of the chosen TL.

The fourth key component of the model, the *matching function* (describing the notion of system relevance) between document representations and query representations, will be described later.

Let us consider point 1 first. For instance, suppose we want to give a representation, in TL terms, of this paper. First of all, we might want to specify a number of "contextual attributes" of this document (e.g. who the authors are). We can do this by entering the following instance assertion:

```
(and paper
     (func appears-in (sing SIGIR93)))
     (all author
          (func affiliation (sing IEI-CNR)))
     (c-some
          author (sing Carlo-Meghini))
     (c-some
          author (sing Fabrizio-Sebastiani))
     (c-some
          author (sing Umberto-Straccia))
     (c-some
          author (sing Costantino-Thanos))
     (exactly 4 author)) [paper666]
```

This asserts that `paper666` (a unique code we give to the document) is a `paper` that `appears-in` the proceedings of `SIGIR93`, that has exactly four `authors`, that these authors are `Carlo-Meghini`, `Fabrizio-Sebastiani`, `Umberto-Straccia` and `Costantino-Thanos`, and that they are affiliated with `IEI-CNR`.

Quite likely, we might want to go beyond this simple specification. For instance, we might want to describe also some of the layout characteristics of the paper; we can do this by entering the following assertion:

```
(and (func typeset-with (sing LaTeX))
     (func format (sing double-column))
     (no figure)
     (no running-header)
     (no running-footer)) [paper666]
```

This asserts that `paper666` has been typeset with LaTeX, that it has been formatted in double column and that it has no figures, running headers and running footers.

Note that we have added this latter information to the former in a piecemeal fashion, i.e. without the need to search and modify the previously entered description of `paper666`; in fact, total *incrementality* is an "added bonus" of adopting a *logic* as a representation language, a possibility that is seldom offered by non-logically-specified representation languages.

Also, the fact that the contextual attributes and layout characteristics have been represented by means of the same representation language, and using the same set of operators, *grants equal status to these two kinds of knowledge*. This obviously extends to the other kinds of knowledge that we discuss below.

At his point, we might not be content with describing just contextual attributes and layout characteristics, and we might want to detail how the paper is structurally organized:

```
(and (exactly 1 abstract)
     (exactly 5 section)
     (exactly 1 bibliography)) [paper666]

bibliography [paper666,bib666]

(and (func typeset-with (sing BibTeX))
     (func style (sing plain))
     (exactly 22 reference)) [bib666]
```

[7] For the sake of simplicity, we will assume that none among the operators in Σ may be expressed by means of a combination of other operators of Σ; this will allow us to say that different subsets always have different expressive power.

[8] See [7] for a review of these complexity results.

The first assertion states that paper666 has an abstract, five sections and a bibliography; the second states that bib666 is the bibliography of paper666; the third states that bib666 has been typeset with BIBTEX using the plain bibliographic style and has twentytwo bibliographic references.

Note that we have represented the bibliography of paper666 as an individual in its own right, in order to be able to make assertions about it. It goes without saying that all assertions involving bib666 contribute to the meaning of the individual constant paper666, as the two co-occur in an assertion according to which the former is the bibliography of the latter.

After representing contextual attributes, layout characteristics and structural characteristics of paper666, we might want to represent information about its content:

```
(and (c-some dw (sing Mirtl))
     (c-some dw (sing syn666))
     (c-some dw (sing sem666))
     (c-some dw (sing alg666))
     (c-some dw (and terminological-logic
                     (c-some modelling-tool
                             (sing IR)))))
                                   [paper666]

terminological-logic [Mirtl]

syntax [Mirtl,syn666]

semantics [Mirtl,sem666]

inferential-algorithm [Mirtl,alg666]
```

These assertions state that paper666 deals with ("dw") MIRTL, its syntax, semantics and inferential algorithm, and in general with terminological logics as modelling tools for information retrieval.

Up to now we have dealt with simple examples of how to use a TL for representing a document. One advantage of the terminological model of IR is that the same TL may also be used for formulating queries: these queries will be evaluated against the document representations built according to the above guidelines. For instance, suppose that the user wants to retrieve all papers authored (or co-authored) by Costantino Thanos and dealing with the semantics of TLs. The set of all such papers may be represented as:

```
(and paper
     (c-some author
             (sing Costantino-Thanos))
     (c-some dw
             (c-some (inv semantics)
                     terminological-logic)))
```

Note that queries may express information pertaining to different aspects of the document: in our example, authorship by Costantino Thanos is a contextual attribute of the documents we want to be retrieved, while the fact that they deal with the semantics of TLs is a matter of semantic content.

The system will then just need to retrieve all the documents represented by individual constants that are instances of this concept. In Section 4 we will see that this task corresponds essentially to subsumption checking; subsumption is thus the matching function that gives an adequate account of (system) relevance. When evaluated, the above query will return paper666, among others, because in a previous assertion we have stated that paper666 is both a paper and a (**c-some** dw (**sing** sem666)), and in a subsequent assertion we have stated that sem666 is the semantics of MIRTL, which in a previous assertion had been stated to be a TL; the same argument trivially applies for authorship by Costantino Thanos.

However, suppose that the user now wants to retrieve all papers dealing with the semantics of extensional logics. The set of all such papers may be represented as:

```
(and paper
     (c-some dw
             (c-some (inv semantics)
                     extensional-logic)))
```

Terminological logics are indeed extensional logics, and because of this the user would probably be interested in paper666. Nonetheless, paper666 will not be retrieved; in fact, sem666 is *not* an instance of (**c-some** (**inv** semantics) extensional-logic), as no assertion has yet gone to the trouble of specifying that all TLs are indeed extensional logics. In order to repair this situation, we may add the following terminological axioms, from which it can be inferred that TLs are indeed extensional logics:

```
terminological-logic ≐
   (and logic
        (func syntax
              term-oriented-syntax)
        (func semantics
              extensional-semantics))

extensional-logic ≐
   (and logic
        (func semantics
              extensional-semantics))
```

These axioms may be regarded as dictionary entries, providing a definition of the concepts of terminological-logic and extensional-logic in terms of other concepts. After the addition of these further pieces of information, paper666 will indeed be retrieved as a result of the above query. Terminological axioms allow thus the specification of lexical, "thesaural" knowledge, i.e. they contribute to the specification of the meaning of the predicate symbols used in both document representation and query formulation. In the process of subsumption checking this kind of knowledge is brought to bear, and serves thus as "background knowledge" according to which queries are to be interpreted; the net effect is that axioms are in fact a recall-enhancing mechanism (a mechanism that has no negative side-effects in terms of precision), because it is by virtue of them that

documents relevant to the query that would have otherwise gone undetected are discovered to be such.

4 The Terminological Model of IR. A formal specification.

In the previous section we have informally described the terminological model of IR by showing how a model of documents, a model of queries and a model of thesaural knowledge can all be accommodated within a TL, and how subsumption is a matching function that gives an adequate account of (system) relevance. We now go on to formally specify the terminological model by specifying one particular TL, the MIRTL logic, that we deem particularly suitable for our task, as it embodies a set of linguistic primitives that prove of considerable interest in the IR context; MIRTL will thus serve as an example of the potentialities of the terminological model, although the model itself is independent of the choice of a particular TL. We will first deal with the syntactic and semantic aspects of MIRTL, and then switch to a brief description of how a deductive algorithm can be designed for it.

4.1 Syntax and semantics of MIRTL

In order to introduce the syntax of MIRTL we will need three disjoint alphabets: an alphabet I of individual constants (with metavariables i, i_1, i_2, ...), an alphabet P_m of monadic predicate symbols (with metavariables M, M_1, M_2, ...) and an alphabet P_d of dyadic predicate symbols (with metavariables D, D_1, D_2, ...).

Let us first discuss the terminological module. The syntax of this module is specified by the following set of BNF clauses.

$$
\begin{array}{lcl}
\langle concept \rangle & ::= & \langle monadic\ predicate\ symbol \rangle \\
& & \mid (\textbf{top}) \\
& & \mid (\textbf{bottom}) \\
& & \mid (\textbf{a-not}\ \langle monadic\ predicate\ symbol \rangle) \\
& & \mid (\textbf{sing}\ \langle individual\ constant \rangle) \\
& & \mid (\textbf{and}\ \langle concept \rangle^{+}) \\
& & \mid (\textbf{all}\ \langle role \rangle\ \langle concept \rangle) \\
& & \mid (\textbf{c-some}\ \langle role \rangle\ \langle concept \rangle) \\
& & \mid (\textbf{atleast}\ \langle natural\ number \rangle\ \langle role \rangle) \\
& & \mid (\textbf{atmost}\ \langle natural\ number \rangle\ \langle role \rangle) \\
\langle role \rangle & ::= & \langle dyadic\ predicate\ symbol \rangle \\
& & \mid (\textbf{inv}\ \langle role \rangle)
\end{array}
$$

We will use metavariables $C, C_1, C_2, \ldots$ ranging on concepts and metavariables $R, R_1, R_2, \ldots$ ranging on roles. The informal meaning of the above primitives is the following:

- (**top**) and (**bottom**) denote the set of all individuals of the domain of discourse and the empty set, respectively;
- (**a-not** M) denotes the set of all individuals of the domain that are not denoted by M;
- (**sing** i) denotes the set containing only the individual denoted by i; this construct is included in order to be able to have individual constants as subcomponents of concepts;
- (**and** $C_1\ C_2\ \ldots C_n$) denotes the set of those individuals that are denoted by C_1 and, at the same time, by C_2 and $\ldots C_n$;
- (**all** $R\ C$) denotes the set of those individuals whose R's are all C's; for instance, (**all** `author italian`) denotes the set of individuals whose `authors` are all `italians`;
- (**c-some** $R\ C$) denotes the set of those individuals having at least one R that is a C; for instance, (**c-some** `author italian`) denotes the set of individuals that have at least one `author` who is an `italian`;
- (**atleast** $n\ R$) (resp. (**atmost** $n\ R$)) denotes the set of those individuals having at least (resp. at most) n R's;
- (**inv** R) denotes the set containing the inverses of those pairs that are denoted by R; for instance, (**inv** `husband`) will be, under the obvious interpretation, equal to the role `wife`.

We will also use the following shorthands:

- (**exactly** $n\ R$) will be used in place of (**and** (**atleast** $n\ R$) (**atmost** $n\ R$));
- (**func** $R\ C$) will be used in place of (**and** (**all** $R\ C$) (**exactly** $1\ R$));
- (**no** R) will be used in place of (**atmost** $0\ R$).

The meaning of the term constructors and of other linguistic primitives of MIRTL may be more formally described by means of the following definitions, given in the style of denotational semantics.

For the logically uninitiated, we should say that *denotational semantics* (also known as *model-theoretic* or *Tarskian semantics*) is the standard way of formally specifying the meaning of logical languages. Such a specification is accomplished by postulating the existence of a number of "ways the world could be" (*interpretations*), and of systematically specifying in which of these interpretations the expressions of the language are true. Inference is then defined as the derivation of only those formulae that are true in all the interpretations in which the premises are also true. The specification that follows fully conforms to this systematic pattern.

Definition 1 *An* interpretation $\mathcal{I}$ *over a nonempty set of individuals* $\mathcal{D}$ *(the* domain of discourse*) is a function that maps individual constants into elements of* $\mathcal{D}$ *such that* $\mathcal{I}(i_1) \neq \mathcal{I}(i_2)$ *whenever* $i_1 \neq i_2$*, concepts into subsets of* $\mathcal{D}$ *and roles into subsets of* $\mathcal{D} \times \mathcal{D}$ *in such a way that:*

$$
\begin{array}{l}
\mathcal{I}(\textbf{top}) = \mathcal{D} \\
\mathcal{I}(\textbf{bottom}) = \emptyset \\
\mathcal{I}(\textbf{a-not}\ M) = \mathcal{D} \setminus \mathcal{I}(M) \\
\mathcal{I}(\textbf{sing}\ i) = \{x \in \mathcal{D} \mid x = \mathcal{I}(i)\} \\
\mathcal{I}(\textbf{and}\ C_1\ C_2\ \ldots C_n) = \\
\qquad \mathcal{I}(C_1) \cap \mathcal{I}(C_2) \cap \ldots \cap \mathcal{I}(C_n) \\
\mathcal{I}(\textbf{all}\ R\ C) = \\
\qquad \{x \in \mathcal{D} \mid\ \forall y : \langle x, y \rangle \in \mathcal{I}(R) \Rightarrow y \in \mathcal{I}(C)\} \\
\mathcal{I}(\textbf{c-some}\ R\ C) = \\
\qquad \{x \in \mathcal{D} \mid\ \exists y : \langle x, y \rangle \in \mathcal{I}(R) \wedge\ y \in \mathcal{I}(C)\}
\end{array}
$$

$\mathcal{I}(\textbf{at-least}\ n\ R) =$
$\{x \in \mathcal{D} \mid \| \{y \in \mathcal{D} | \ \langle x,y\rangle \in \mathcal{I}(R)\} \| \geq n\}$
$\mathcal{I}(\textbf{at-most}\ n\ R) =$
$\{x \in \mathcal{D} \mid \| \{y \in \mathcal{D} | \ \langle x,y\rangle \in \mathcal{I}(R)\} \| \leq n\}$
$\mathcal{I}(\textbf{inv}\ R) =$
$\{\langle x,y\rangle \in \mathcal{D} \times \mathcal{D} \mid \langle y,x\rangle \in \mathcal{I}(R)\}$ ■

In our formalism, interpretations are then all and only those mappings of linguistic items onto the domain of discourse, that comply with our intended meaning of the term-forming operators. Interpretations will be, in fact, the only mappings we will be interested in.

Definition 1 completes the specification of the terminological module of MIRTL; let us now switch to the definitional module. The next definition introduces terminological axioms and specifies their syntax and semantics.

Definition 2 *A* terminological axiom *(or* axiom*, for short) is an expression of the form $M \sqsubset C$ or $D \sqsubset R$ (in this case the axiom is called a* connotation*) or of the form $M \doteq C$ or $D \doteq R$ (in this case it is called a* definition*), where M is an element of P_m, D is an element of P_d, C is a concept and R is a role. An interpretation $\mathcal{I}$ over a nonempty domain $\mathcal{D}$* satisfies a connotation *$M \sqsubset C$ (resp. $D \sqsubset R$) iff $\mathcal{I}(M) \subseteq \mathcal{I}(C)$ (resp. iff $\mathcal{I}(D) \subseteq \mathcal{I}(R)$), and* satisfies a definition *$M \doteq C$ (resp. $D \doteq R$) iff $\mathcal{I}(M) = \mathcal{I}(C)$ (resp. iff $\mathcal{I}(D) = \mathcal{I}(R)$).* ■

A terminological axiom such as

`triangle` $\doteq$ (**and** `polygon` (**atleast** `3 side`) (**atmost** `3 side`))

has thus the effect of restricting the set of interpretations to those in which the term `triangle` and the term (**and** `polygon` (**atleast** `3 side`) (**atmost** `3 side`)) are mapped to the same subset of the domain of discourse. Our use of axioms to encode thesaural knowledge will then have the desired effect, i.e. that of enforcing the desired relations of synonymity or "conceptual inclusion".

We will use metavariables $\delta, \delta_1, \delta_2, \ldots$ ranging on axioms and metavariables $\Delta, \Delta_1, \Delta_2, \ldots$ ranging on sets of axioms.

Definition 2 completely handles the definitional module of MIRTL. The syntax and semantics of the assertional module are specified in Definition 3.

Definition 3 *An* assertion *is an expression of the form $C[i]$ or of the form $R[i_1, i_2]$, where C is a concept, R is a role and i, i_1, i_2 are individual constants. An interpretation $\mathcal{I}$ over a nonempty domain $\mathcal{D}$* satisfies an assertion *$C[i]$ iff $\mathcal{I}(i) \in \mathcal{I}(C)$, and* satisfies an assertion *$R[i_1, i_2]$ iff $\langle \mathcal{I}(i_1), \mathcal{I}(i_2)\rangle \in \mathcal{I}(R)$. An assertion is* valid *iff it is satisfied by all interpretations.* ■

An assertion such as

(**and** `paper`
(**func** `appears-in` (**sing** `SIGIR93`)))
[`paper666`]

has thus the effect of restricting the set of interpretations to those in which the set onto which the term (**and** `paper` (**func** `appears-in` (**sing** `SIGIR93`))) is mapped includes the individual onto which the individual constant `paper666` is mapped.

We will use metavariables $\gamma, \gamma_1, \gamma_2, \ldots$ ranging on assertions and metavariables $\Gamma, \Gamma_1, \Gamma_2, \ldots$ ranging on sets of assertions.

This completes the definition of the syntax and semantics of the three MIRTL modules. We may now take a look at what a MIRTL knowledge base is.

Definition 4 *A* MIRTL knowledge base *is a pair $\Omega = \langle \Delta, \Gamma\rangle$, where Δ is a set of terminological axioms and Γ is a set of assertions. An interpretation $\mathcal{I}$* satisfies a knowledge base *$\Omega = \langle \Delta, \Gamma\rangle$ iff it satisfies all axioms in Δ and all assertions in Γ; in this case, we say that $\mathcal{I}$ is a* model *of Ω. A knowledge base Ω is* satisfiable *iff it has a model.* ■

We will use metavariables $\Omega, \Omega_1, \Omega_2, \ldots$ ranging on knowledge bases.

For IR purposes, a MIRTL knowledge base will be a representation of the document base, where assertions will represent documents and their appurtenance to given document classes, and axioms will represent lexical, "thesaural" knowledge.

What we need to do now is to specify how we want to represent the service that is to be provided by the IR system. Let us remind that we declared our substantial adhesion to the picture espoused in [21, 22], according to which, in response to a query q, the service to be provided is the retrieval of all and only those documents d that make the conditional $d \rightarrow q$ valid. In the terminological model, this means retrieving those documents represented by individual constants i such that (**sing** i) is subsumed by the concept C representing the query. However, in checking subsumption between (**sing** i) and C, *the contents of the knowledge base must be brought to bear*: in fact, we want the assertions concerning i and the axioms involving (directly or indirectly) the predicate symbols occurring in C to participate in the inferential process. We need then a notion of "subsumption *modulo* the contents of Ω"[9].

Definition 5 *Let Ω be a satisfiable knowledge base, and let C_1, C_2 be two concepts. We say that C_1 is* subsumed *by C_2 in Ω (written $C_1 \preceq_\Omega C_2$) iff for every model $\mathcal{I}$ of Ω it is true that $\mathcal{I}(C_1) \subseteq \mathcal{I}(C_2)$.* ■

The terminological model then sees IR as the task of retrieving, as a response to a query C, all and only those documents i such that (**sing** i) $\preceq_\Omega C$, where Ω is a TL representation of the document base. In other words, IR is the task of retrieving all those documents whose membership in the class denoted by C is a direct consequence of the truth of all the assertions and axioms of Ω.

It may be interesting to note that both the vector model and the boolean model of IR are special cases of our terminological model: the vector model is obtained by letting the definitional module be empty and the terminological module consist of the **and** operator only, while the boolean model is obtained by also considering the **or** and **not** operators, that can be defined in the obvious way.

[9] Subsumption *modulo* the contents of Ω is what, in Section 2, we have called "explicit subsumption"; "implicit subsumption" is then explicit subsumption in an empty knowledge base.

Before closing this section, we note that our model does not suffer from the "false document problem" (see e.g. [5, page 240]). In our model, a "false document" i obviously is one for which no assertion of type $C[i]$ has been entered: it turns out that no query will cause the retrieval of i, as i is not subsumed by any non-trivial concept[10].

4.2 Reasoning on MIRTL

We now turn to briefly sketching how an algorithm that performs deductions on MIRTL knowledge bases can be specified, and how such algorithm should be used for best results in an IR context.

Schmidt-Schauß and Smolka [19] have recently proposed *constraint propagation* as a technique for specifying inferential algorithms for TLs. This proposal has gained widespread acceptance, and constraint propagation has virtually become the standard technique in the field; in fact, it allows to easily specify inferential algorithms for TLs in a way that is both modular and informative as to what the computational complexity of the resulting algorithm is.

In order to introduce our algorithm, we will need an alphabet V of individual variables (with metavariables x, x_1, x_2, ..., y, y_1, y_2, ...). Individual variables and individual constants will collectively be called *objects* (with metavariables w, w_1, w_2, ...). We define *constraints* (with metavariables θ, θ_1, θ_2, ...) to be expressions of the form $C[w]$ or of the form $R[w_1, w_2]$, where C is a concept, R is a role and w, w_1, w_2 are objects. Assertions are thus a particular brand of constraints. *Constraint sets* (CSs - with metavariables Θ, Θ_1, Θ_2, ...) will be finite sets of constraints. A CS is said to contain a *clash* iff it contains one of the following:

- a constraint of type $M[w]$ and another constraint of type (**a-not** M)$[w]$;
- a constraint of type (**bottom**)$[w]$;
- a constraint of type (**sing** i_1)$[i_2]$, with $i_1 \neq i_2$;
- a constraint of type (**atleast** n R)$[w]$ and another constraint of type (**atmost** m R)$[w]$, with $n > m$.

Constraint propagation works by transforming a CS Θ into a CS Θ' by adding to it one or more constraints; each transition between CSs is the result of the application of a *completion rule*. For instance, one such rule (the "$\rightarrow_{\mathbf{all}}$" rule) stipulates that, if the constraints (**all** R C)$[w_1]$ and $R[w_1, w_2]$ both belong to Θ, the constraint $C[w_2]$ should also be added to Θ. Constraint propagation algorithms for TLs basically consist of a set of completion rules, one for each (major) term-forming operator belonging to the terminological module of the logic in question.

Most such algorithms work by reducing the problem of checking subsumption to the problem of checking knowledge base unsatisfiability. To check whether a knowledge base $\Omega = \langle \Delta, \Gamma \rangle$ is unsatisfiable, Ω is first transformed into an equivalent set of constraints Θ by expanding all assertions in Γ by means of the axioms in Δ. The completion rules are then applied to Θ until the resulting constraint set Θ' is *complete*, i.e. until no more rule is applicable; the way the completion rules are specified guarantees the finiteness of the process. If Θ' is clash-free (this is easily checked), then the subsumption relationship in question does indeed obtain; otherwise, it does not. The set of completion rules for MIRTL is given in an extended version of this paper [13].

Given these premises, it might be argued that the process we have described looks intolerably slow for IR purposes: if for every query C and for every document i contained in the document base represented by Ω, the system can decide whether to retrieve i as a response to C only by checking that (**sing** i) $\preceq_\Omega C$ by means of the above process, then it looks like answering to a single query might take years!

In general, however, it will not be the case that, given a query C, this algorithm is applied anew for each document i in the document base; and it will neither be the case that this algorithm is applied anew for each new query C! In fact, unlike most other logics, TLs encourage inference to be performed *at KB construction time* rather than at query time. This means that the most relevant part of the inferential workload is performed while knowledge (i.e. axioms and assertions) is entered.

Accordingly, the standard way of operating in an IR setting will be to constantly keep the KB representing the document base in the form of a complete constraint set while it is incrementally being built. Given a KB in the form of a complete constraint set, the addition of an assertion (resp. an axiom) to the KB will involve the application of the completion rules to the assertion (resp. to the constraints affected by the axiom) only, hence causing a minor update to the constraint set itself. In this way, the constraint set is built incrementally, and it is already in complete (hence computationally convenient) form when queries are formulated to the system[11]. When a query is formulated, (a modified form of) it is added to the constraint set and the completion rules are applied; at this point, for every individual constant i occurring in the set, the (**sing** i) $\preceq_\Omega C$ condition is checked by simple table lookup techniques. In the end, this means that one single processing of the knowledge base serves multiple queries, and one single processing of a query serves multiple documents, thus reducing the query-time theorem proving operation to table lookup.

5 Further research

In this paper we have argued that a terminological logic may be profitably used for IR modelling tasks, yielding a *terminological model* of IR. Actually, we think that the material presented here just scratches the surface of the problem, and that a lot of issues still need to be tackled in order to arrive at a satisfactory account of the

[10] Here "non-trivial" means "not mentioning i itself": quite obviously, if the query is the concept (**sing** i) (corresponding to an *explicit* user request to retrieve document i), i is of course retrieved as a result of it.

[11] Incidentally, we should note that by keeping the knowledge base in the form of a complete constraint set, we get consistency checking of the knowledge base "for free".

IR endeavour.

Among the issues that, in our opinion, are the most urgent and/or promising are those related to the expressive power of the TL. The MIRTL logic we have described in the previous sections may be considered as a first attempt at IR modelling, and as a demonstrative of what the potential of TLs is for such a task. Of course, MIRTL is still susceptible of modifications under several respects.

First of all, the set of term-forming operators that make up its terminological module is far from being final. The choice of primitives has been made with an eye towards maintaining the logic small enough to be computationally viable, while at the same time having enough expressive power for accounting for the representation of documents of real-life complexity. However, other operators might be deemed particularly useful, or even necessary.

An inadequacy of the model as it stands is that it only addresses what has been called "correspondence" (or"system relevance") of a document to a query; it is clear that a further effort is needed to have it address "user relevance" too. Another inadequacy of the model as it stands is that, even confining the discussion to "system relevance", only "total" (or "perfect") relevance is modelled; again, a further effort is needed to have the model address "partial relevance" too. Luckily enough, current research in TLs seems to indicate that both these shortcomings can be addressed by remaining within the paradigm of TLs. To this respect, we are currently considering the idea of introducing probabilistic reasoning in our model along the approach described in [8].

The computational complexity of the reasoning algorithm is also a major problem that has to be tackled[12]. From this point of view, we are already treading on dangerous ground, as MIRTL is a proper superset of the $\mathcal{ALEN}$ logic[13], which is known to be NP-hard [7].

In order to solve this problem we are currently working [12] on a variant of MIRTL based, rather than on classical logic, on *relevance logic* [1]; early results in the field suggest that "relevance" TLs are in general computationally easier to handle than their classical equivalents [17].

An alternative solution to the complexity problem that we are considering is the investigation of probabilistic algorithms for subsumption checking. The answer that these algorithms would give to the question if concept C_1 subsumes concept C_2 would be accurate only with probability $P < 1$; while, given the seemingly inherent impossibility of achieving perfect performance in IR systems, this would not be a great loss, the advantage to be gained might be considerable, as in many cases probabilistic algorithms have much better complexity characteristics than their non-probabilistic counterparts.

[12] Actually, when speaking of the complexity of a problem, one always refers to "worst case complexity"; although in an IR context this might not be an interesting parameter, in the absence of statistical characterizations of the "average case" we will use it as a first approximation of the computational feasibility of the logics in question.

[13] The terminological module of MIRTL is equal to $\mathcal{ALEN}$ plus the **sing** and **inv** operators.

Acknowledgements

This work has been partially funded by the ESPRIT BRA Working Group MIRO. Thanks to the members of MIRO and to three anonymous referees for providing stimulating comments.

References

[1] Alan R. Anderson and Nuel D. Belnap. *Entailment - the logic of relevance and necessity.* Princeton University Press, Princeton, NJ, 1975.

[2] Franz Baader. A formal definition for the expressive power of knowledge representation languages. In *Proceedings of ECAI-90, 9th European Conference on Artificial Intelligence*, pages 53–58, Stockholm, Sweden, 1990.

[3] Ronald J. Brachman. A structural paradigm for representing knowledge. Technical Report 3605, Bolt Beranek and Newman, Cambridge, MA, 1978.

[4] Ronald J. Brachman and Hector J. Levesque, editors. *Readings in knowledge representation.* Morgan Kaufmann, Los Altos, CA, 1985.

[5] Yves Chiaramella and J.P. Chevallet. About retrieval models and logic. *The Computer Journal*, 35:233–242, 1992.

[6] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Werner Nutt. The complexity of concept languages. In *Proceedings of KR-91, 2nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 151–162, Cambridge, MA, 1991.

[7] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. From subsumption to instance checking. Technical Report 15.92, Università degli studi di Roma "La Sapienza". Dipartimento di informatica e sistemistica, Rome, Italy, 1992.

[8] Joseph Y. Halpern. An analysis of first-order logics of probability. In *Proceedings of IJCAI-89, 11th International Joint Conference on Artificial Intelligence*, pages 1375–1381, Detroit, MI, 1989.

[9] John Haugeland, editor. *Mind design.* The MIT Press, Cambridge, MA, 1981.

[10] Richard Hull and Roger King. Semantic database modelling: survey, applications and research issues. *ACM Computing Surveys*, 19:201–260, 1987.

[11] Carlo Meghini, Fausto Rabitti, and Costantino Thanos. Conceptual modeling of multimedia documents. *IEEE Computer*, 24(10):23–30, 1991.

[12] Carlo Meghini and Fabrizio Sebastiani. Towards a relevance logic of information retrieval. Unpublished manuscript, 1992.

[13] Carlo Meghini, Fabrizio Sebastiani, Umberto Straccia, and Costantino Thanos. A model of information retrieval based on a terminological logic (extended version). Technical report, Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy, 1993. Forthcoming.

[14] Marvin Minsky. A framework for representing knowledge. In Patrick J. Winston, editor, *The psychology of computer vision*, pages 211–277. McGraw-Hill, New York, NY, 1975. [a] An extended version appears also in [4], pp. 245–262, and in [9], pp. 95–128.

[15] Bernhard Nebel. Terminological cycles: Semantics and computational properties. In John F. Sowa, editor, *Principles of Semantic Networks*, pages 331–361. Morgan Kaufmann, Los Altos, 1991.

[16] Jianyun Nie. An information retrieval model based on modal logic. *Information processing and management*, 25:477–491, 1989.

[17] Peter F. Patel-Schneider. A four-valued semantics for frame-based description languages. In *Proceedings of AAAI-86, 5th Conference of the American Association for Artificial Intelligence*, pages 344–348, Philadelphia, PA, 1986.

[18] Joan Peckham and Fred Maryanski. Semantic data models. *ACM Computing Surveys*, 20:154–189, 1988.

[19] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48:1–26, 1991.

[20] Fabrizio Sebastiani and Umberto Straccia. A computationally tractable terminological logic. In *Proceedings of SCAI-91, 3rd Scandinavian Conference on Artificial Intelligence*, pages 307–315, Roskilde, Denmark, 1991.

[21] Keith van Rijsbergen. A new theoretical framework for information retrieval. In *Proceedings of the 1986 ACM Conference on Research and Development in Information Retrieval*, pages 194–200, Pisa, Italy, 1986.

[22] Keith van Rijsbergen. A non-classical logic for information retrieval. *The Computer Journal*, 29:481–485, 1986.