Logical and Computational Properties of the Description Logic MIRTL[*]

P. Buongarzoni, C. Meghini, R. Salis, F. Sebastiani and U. Straccia
Istituto di Elaborazione dell'Informazione
Consiglio Nazionale delle Ricerche
Via S. Maria, 46 - 56126 Pisa (Italy)
E-mail: $\{lastname\}$@iei.pi.cnr.it

1 Introduction

In recent years a number of positive (i.e. tractability and decidability) results have been found concerning the computational complexity of Description Logics (DLs) [Buchheit *et al.*, 1993; Donini *et al.*, 1991; Donini *et al.*, 1992a; Donini *et al.*, 1992b; Schmidt-Schauß and Smolka, 1991; Sebastiani and Straccia, 1991]. Unfortunately, also negative results have appeared, e.g. showing that some DLs (e.g. NIKL [Patel-Schneider, 1989] and KL-ONE [Schmidt-Schauß, 1989]) are undecidable. This work contributes to this latter literature by showing a negative result for another DL (called MIRTL) which had not been previously studied from the standpoint of computational complexity, and which is not directly related to the ones already shown undecidable; in more standard DL terminology, MIRTL is the $\mathcal{ALEN}$ logic plus the $\mathcal{I}$ operator for inverse roles and the $\mathcal{O}_1$ operator for singleton concepts[1].

We have recently been investigating the use of MIRTL for modelling multimedia information retrieval (see [Meghini *et al.*, 1993; Sebastiani, 1994]). Given the requirements imposed by this application domain, and given the well-known negative results on the computational complexity of more powerful DLs, we had deemed MIRTL the best compromise between expressivity and tractability for our purposes. Somehow encouraged by the fact that the standard reasoning problems in the related logics $\mathcal{ALCNR}$ [Buchheit *et al.*, 1993], $\mathcal{ALCO}$ [Schaerf, 1994] and $\mathcal{ALNI}$ (also known as $\mathcal{PL}_1$) [Donini *et al.*, 1991] are all decidable, and considering that the well-known algorithms based on constraint propagation for reasoning on them tend to be easily customizable to a chosen set of operators, we had thought that developing a sound and complete algorithm for MIRTL could be reasonably straightforward.

Unfortunately, while trying to develop such an algorithm, we discovered that MIRTL does not have the *finite model property*: i.e. there are satisfiable MIRTL concepts (and assertions, and KBs) which are satisfiable only in interpretations of infinite cardinality. Although this result is not one of full-blown undecidability[2], it however casts a shadow on the possibility of making practical use of such a logic. In fact, since the constraint propagation algorithms by now standard in the field of DLs prove the satisfiability of a concept by building a finite model of the concept, these algorithms are not applicable to MIRTL unless one renounces to guaranteed termination, or unless one builds some non-trivial loop-detecting control structure into them.

Once we abandon the idea of checking the satisfiability of a concept by building a finite model for it, the problem arises whether an alternative method exists or not. We have thus tried to find an upper bound to the complement of the satisfiability problem, i.e. to find a threshold under which possible inconsistencies should necessarily show up and above which no new inconsistency could emerge any more. This work reports on some negative results we have obtained in this direction, and which might constitute a prelude to a true undecidability result[3].

2 MIRTL admits infinitary concepts

The language of MIRTL includes primitive concepts (A), negation of primitive concepts ($\neg A$), concept conjunction ($C \sqcap D$), universal quantification ($\forall R.C$), quali-

*This work has been carried out in the context of the project FERMI 8134 - "Formalization and Experimentation in the Retrieval of Multimedia Information", funded by the European Community under the ESPRIT Basic Research scheme.

[1]This latter operator is a restricted form of the $\mathcal{O}$ operator (also known as **one-of**); the extension of a singleton concept $\{a\}$ is just the singleton containing the individual denoted by a.

[2]There are in fact decidable logics that do not have the finite model property; see e.g. [Hughes and Cresswell, 1984; page 154] or [Vardi and Wolper, 1986].

[3]Until recently we actually thought we had an undecidability proof for MIRTL, based on a reduction of the halting problem for a model of computation equivalent to Turing Machines; the proof was then shown to contain a mistake by Diego Calvanese, a mistake that we have not been able to fix yet.

fied existential quantification ($\exists R.C$), number restrictions (($\geq n\ R$) and ($\leq n\ R$)), inversion of roles (R^{-1}) and singleton concepts ($\{a\}$). We will also use the notation ($= n\ R$) in place of ($\geq n\ R$) $\sqcap$ ($\leq n\ R$), and the notation $f(R).C$ in place of ($= 1\ R$) $\sqcap$ ($\forall R.C$). Finally, Mirtl allows assertions $C[a]$ and $R[a, b]$, where C is a concept, R is a role and a, b are individual constants; $C[a]$ states that a is an instance of C, whereas $R[a, b]$ states that $\langle a, b\rangle$ is an instance of R. The semantics of these expressions is standard (see e.g. [Donini *et al.*, 1992a]), so it will be omitted here.

Consider now the following Mirtl concept:

$$\begin{array}{l} \{z\} \sqcap (\leq 0\ S^{-1})\ \sqcap \\ f(S).(f(G^{-1}).\{z\})\ \sqcap \\ \forall G.(f(S).((=\ 1\ S^{-1})\ \sqcap\ f(G^{-1}).\{z\}) \end{array} \tag{1}$$

The effect of concept (1) is to state some properties of the natural number zero (z); for instance, it states that z has no predecessor (S^{-1}) and a unique successor (S), and that all the numbers greater than z have a unique successor, which in turn is greater than z and has a unique predecessor.

Theorem 2.1 *Concept (1) admits only infinite models.*

Concept (1) is satisfiable, as it admits the following infinite model:

$\mathcal{D} = \{0, 1, 2, 3, \ldots\}$
$z^{\mathcal{I}} = \{0\}$
$S^{\mathcal{I}} = \{\langle 0, 1\rangle, \langle 1, 2\rangle, \langle 2, 3\rangle, \ldots\}$
$G^{\mathcal{I}} = \{\langle 0, 1\rangle, \langle 0, 2\rangle, \langle 0, 3\rangle, \ldots\}$

It can also be shown that if an interpretation $\mathcal{I}$ is a model of concept (1), then $\mathcal{I}$ must be defined on an infinite domain $\{z^{\mathcal{I}}, d_1, \ldots, d_n, \ldots\}$ such that the extension of G contains $\{\langle z^{\mathcal{I}}, d_1\rangle, \ldots, \langle z^{\mathcal{I}}, d_n\rangle, \ldots\}$ and the extension of S contains $\{\langle z^{\mathcal{I}}, d_1\rangle, \langle d_1, d_2\rangle, \ldots, \langle d_{n-1}, d_n\rangle, \ldots\}$.

This shows the existence of Mirtl concepts which are satisfied only in interpretations with infinite domain; we will call them *infinitary concepts*. Similar concepts had already been shown to exist for other DLs, albeit substantially more expressive [Schild, 1991; Calvanese *et al.*, 1994].

The presence of infinitary concepts has the practical consequence that the standard methods based on constraint propagation (e.g. the methods discussed in [Schmidt-Schauß and Smolka, 1991; Buchheit *et al.*, 1993]) do not work properly; since these methods attempt to build a model of the concept they want to prove consistent, once applied to infinitary concepts they loop forever, unless they are endowed with a control structure able to detect the construction of an infinitely self-replicating structure (which every Mirtl infinitary concept, from what our experiments have shown, seems to contain).

The existence of Mirtl infinitary concepts has a computational counterpart in what might be called *self-reactivating constraints*: as a consequence of the interaction among Mirtl propagation rules (in particular, the rules for universal quantification, role inversion and singleton), some constraints may be such that the rules that process them generate the same activating conditions that obtained before the application of the rule (thus inducing an infinite loop).

After having noticed this, in our search for a decidability result for Mirtl, we had hoped to find, nonetheless, an upper bound on the number of steps necessary to find a clash deriving from an inconsistent Mirtl concept. If we had found such an upper bound, say k, decidability would have been established, as a Mirtl concept could be declared satisfiable after the k-th step had failed to produce a clash. Unfortunately, this strategy proved ineffective: for every $f(n)$ (where n is the size of the concept) that we had, in repeated attempts, conjectured to be such an upper bound, we were able to discovered an unsatisfiable Mirtl concept which generated a clash well after $f(n)$ steps. As we will see in the next section, this has happened for functions f of increasing order of magnitude.

3 Looking for an upper bound

Once we abandon the idea of checking the satisfiability of a concept by building a model for it, the problem arises whether an alternative method exists or not. To this end, we have studied the computational behaviour of various categories of infinitary concepts. For instance, if we graphically represent a constraint system as a directed graph, in which nodes represent objects (individual constants or variables) and the set of concepts constraining them, and edges represent binary assertions between them (the edge is oriented from the first element of the assertion to the second), concept (1) generates a graph that contains an infinite subgraph like the one shown in Figure 1).

By analyzing the structure of the constraint sets that infinitary concepts generate, we have noticed that the generated graph has the following structural property: it is composed by a "kernel" subgraph, in which all the constants appearing in the concept and some variables occur, to which other subgraphs are connected that repeat *ad infinitum* portions of the kernel subgraph. This shows that the constraint propagation process, after a finite number of de-activations of the same $\forall$-constraint, completes the construction of the kernel subgraph and starts building an infinite number of replicas of portions of it.

After observing this, we have tried to individuate the step of the constraint propagation process at which the building of the kernel subgraph is completed and the

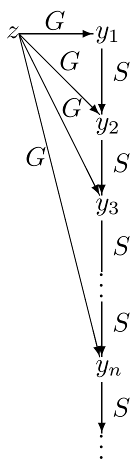

Figure 1: (Partial) graphical representation of the constraints generated by Concept (1).

building of the replicas start. In fact, once this step were individuated, the problem would be solved: in fact, it would suffice to check if the kernel subgraph (plus a small number of replicas) contained an inconsistency, as, if this were not the case, also the replicas would not contain any inconsistency, and the concept could be deemed satisfiable without any further rule application.

In other words, what we have been looking for is an upper bound to the complement of the satisfiability problem (co-**CS**), i.e. a threshold under which possible inconsistencies should necessarily emerge and above which no new inconsistency could possibly emerge. We have tried to find such an upper bound as an upper bound on the number of de-activations of the same constraint, expressed in terms of the size d of the initial concept. In fact, as we have already argued, the cause of non-termination of the computation is the re-activation of a constraint. Hence, we have made the hypothesis that, by limiting this number, one could limit the generation *ad infinitum* of replicas of portions of the kernel subgraph. We then started an empirical study in order to identify the maximum number of activations of the same constraint beyond which the procedure diverges and builds an infinite model.

3.1 Concepts denoting natural numbers (or: the d limit)

In Section 2 we have seen that the procedure of constraint propagation diverges when applied to concept 1. By examining the constraint system generated after $h < d$ de-activations of the self-reactivating constraint

$$(\forall G.(f(S).((=\ 1\ S^{-1})\ \sqcap\ f(G^{-1}).\{z\})))[z]$$

one can verify that no clash occurs in it. Within d activations, inconsistencies due to arbitrarily nested concepts like the following, are discovered:

$$\begin{aligned}&((\forall\ R.((\exists\ P.(\exists\ R^{-1}.\{i\}))\ \sqcap\\&(\forall\ P^{-1}.(\forall\ P^{-1}.(\forall\ P^{-1}.(\forall\ P^{-1}.C'))))))\ \sqcap\\&(\exists\ R)\ \sqcap\ \{i\})\end{aligned}\tag{2}$$

The peculiarity of this case comes from the depth of the syntactic tree of concept 2; the subconcept C' is to be found at depth level 7. In order for C' to constrain an individual, it is necessary that the longest branch of the syntactic tree is completely explored. At this point, C' constrains the individual constant i, and from the development of $C'[i]$ a clash may arise (e.g. if $C' = (\leq\ 0\ R)$), or the infinite generation of variables may be blocked ($C' = (\leq\ n\ R)$), or still other things may happen.

3.2 Concepts denoting the multiples of a natural number (or: the d^2 limit)

The d limit has resisted numerous empirical tests, until we have been able to find a Mirtl concept denoting the set of the multiples of a natural number. For instance, the set of multiples of 2 may be represented by the following Mirtl concept:

$$\begin{array}{lll}
(0)\ C_3 & = & (\{z\}\ \sqcap\\
(1) & & (\leq\ 0\ E^{-1})\ \sqcap\\
(2) & & (\leq\ 0\ O^{-1})\ \sqcap\\
(3) & & (\leq\ 0\ E)\ \sqcap\\
(4) & & (=\ 1\ O)\ \sqcap\\
(5) & & (\exists\ M.\{t\})\ \sqcap\\
(6) & & (\exists\ G.(\exists\ O^{-1}.\{z\}))\ \sqcap\\
(7)(7') & & (\forall\ G.((\forall\ O^{-1}.(\forall\ O.((=\ 1\ O^{-1})\ \sqcap\\
 & & \qquad (\leq\ 0\ O)\ \sqcap\\
 & & \qquad (\leq\ 0\ E^{-1})\ \sqcap\\
 & & \qquad (=\ 1\ E)\ \sqcap\\
 & & \qquad (\forall\ E.((\exists\ G^{-1}.\{z\})\ \sqcap\\
 & & \qquad\quad (\exists\ M.\{t\}))))))\ \sqcap\\
(7'') & & \quad (\forall\ E^{-1}.(\forall\ E.((=\ 1\ E^{-1})\ \sqcap\\
 & & \qquad (\leq\ 0\ E)\ \sqcap\\
 & & \qquad (\leq\ 0\ O^{-1})\ \sqcap\\
 & & \qquad (=\ 1\ O)\ \sqcap\\
 & & \qquad (\forall\ O.(\exists\ G^{-1}.\{z\}))))))))
\end{array}$$

If we interpret the roles O, E, M e G as the "odd successor", "even successor", "multiple-of" and "bound above by", one may see that C_3 denotes the set of even numbers; in fact, starting from the individual constant z and generating variables y_1, y_2, ..., only variables with an even subscript are put in relation with the the individual constant t. In fact, the first sub-concepts state that the natural number 0 (subconcept 0) is neither the even successor of a number (1), nor its odd successor (2); also, 0

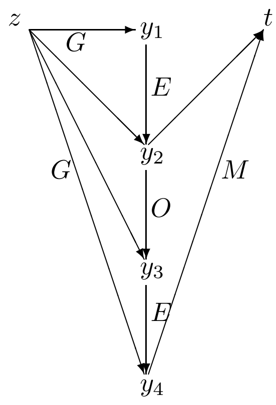

Figure 2: Graphical representation of the set of constraints denoting the multiples of 2.

does not have any even successor (3), has exactly an odd successor (4), is a multiple of 2 (6). Subconcept (7) (i.e. the one which generates the self-reactivating constraint) states that the odd successors of their predecessor have an even successor which is both a multiple of two and a positive number, while the even successors of their predecessors have an odd successor which is positive. In this way, a set of constraints is generated which is partially represented in Figure 2.

Each relation of type M is generated after two activations of the constraint ($\forall\ G \ldots$). In a similar way, it would be possible to describe the multiples of 3, 4, etc..

Note that in the self-reactivating constraint (7) the disjunction of two concepts is simulated by means of two occurrences of the $\forall$ operator; in fact, of the two subconcepts:

(7') $(\forall\ O^{-1}.(\forall\ O.(\ldots)))$
(7") $(\forall\ E^{-1}.(\forall\ E.(\ldots)))))$

only one is activated when they are applied to the same variable. The activation depends on the subscript of the variable: variables with even subscript k represent the even successor of the $k-1$ variable; variables with odd subscript k represent the odd successor of the same variable.

In order to describe the multiples of the numbers greater than 2 it is sufficient to modify concept C_3 by changing roles E and O into S_0 and S_1, and adding as many roles S_j and as many subconcepts $(\forall\ S_j^{-1}.(\forall\ S_j.\ldots)))$ as the number whose successors we want to represent (S_j stands for *"successor modulo j"*).

This concept, which is in itself satisfiable, can be used in order to write unsatisfiable concepts whose first clash occurs after d activations of a constraint. In fact, if the individual constant t were constrained by concept $(\leq\ k\ M^{-1})$, for some k, the clash would be generated only after $n(k+1)-1$ de-activations of the self-reactivating constraint, where n is the number of subconcepts of type $(\forall\ S_j^{-1}.(\forall\ S_j.\ldots)))$ of the self-reactivating constraint. As the numbers n and k have the same order of magnitude of the dimension d of the original concept, the first clash will appear in the constraint system before $O(d^2)$ steps but after $O(d)$ steps.

3.3 Concepts denoting the first n powers of a number (or: the d^d limit)

If we substitute, within concept 1 the self-reactivating subconcept:

$(\forall G.(f(S).((=\ 1\ S^{-1})\ \sqcap\ f(G^{-1}).\{z\})$

by

$$
\begin{aligned}
&(\forall\ G.f(S).((=\ 1\ S^{-1}) \sqcap \\
&\quad f(G^{-1}).\{z\} \sqcap \\
&\qquad (\exists\ R_1.((\leq\ n\ R_1^{-1}) \sqcap \\
&\qquad\quad (\exists\ R_2.((\leq\ n\ R_2^{-1}) \sqcap \\
&\qquad\qquad (\exists\ R_3.((\leq\ n\ R_3^{-1}) \sqcap \\
&\qquad\qquad\quad \{t\}))))))))
\end{aligned}
$$

we obtain an unsatisfiable concept, whose first clash appears within the constraint system after d^2 activations of the self-reactivating constraint. In fact, during the application of the propagation rules, the subconcept $(\exists\ R_1 \ldots)$ generates within the graph of Figure 3 a path of length 3 ending with the individual constant t. When a path for every variable y_j is generated, a tree is created with root t and depth 3. As the individual constant t is constrained by concept $(\leq\ n\ R_3^{-1})$, when variable y_{n^3+1} is generated (at the n^3+1-th disactivation of the $(\forall\ G\ldots)$ constraint, the first clash appears within the constraint system. Adding a role to path R_1, R_2, ... corresponds to incrementing by one the depth of the tree, and consequentially incrementing by one the value h of the exponent; instead, modifying the value n of the number restrictions corresponds to modify the width of the tree, and consequentially modifying the base n. Obviously, both h and n are strictly smaller than d, but still remain of the same order of magnitude; hence, the number of de-activations of a constraint remains exponential in the dimension d of the original concept C.

It is also possible to combine these two latter sources of complexity (multiples of a number (d^2) and powers of a number (d^d)), by generating the role R_1 only for the multiples of a number; this brings the maximum number of activations to $d \cdot d^d$.

4 Conclusions

In this work we have shown that Mirtl does not enjoy the finite model property, and have discussed some consequences of this fact. In particular, we have argued that, even if Mirtl should be proven decidable (which is still, to the best of our knowledge, an open problem), reasoning on it by means of constraint propagation algorithms is probably going to be computationally onerous,

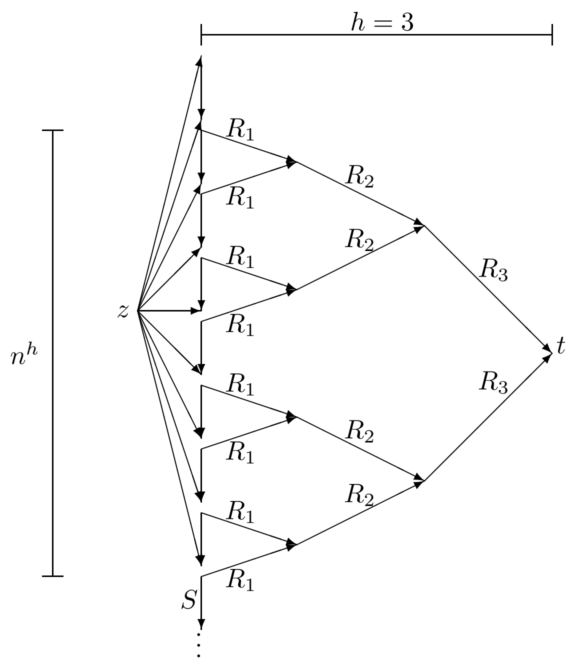

Figure 3: Graphical representation of the constraints denoting the first power of 2.

as Mirtl concepts may be written which generate clashes only after a computationally unfeasible number of steps.

Acknowledgements

We are grateful to Diego Calvanese for his detailed comments to an earlier draft (see Footnote 3), and for giving us interesting pointers to the literature.

References

[Buchheit *et al.*, 1993] Martin Buchheit, Francesco M. Donini, and Andrea Schaerf. Decidable reasoning in terminological knowledge representation systems. *Journal of Artificial Intelligence Research*, 1:109–138, 1993.

[Calvanese *et al.*, 1994] Diego Calvanese, Maurizio Lenzerini, and Daniele Nardi. A unified framework for class-based representation formalisms. In *Proceedings of KR-94, 5th International Conference on Principles of Knowledge Representation and Reasoning*, pages 109–120, Bonn, FRG, 1994.

[Donini *et al.*, 1991] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Werner Nutt. Tractable concept languages. In *Proceedings of IJCAI-91, 12th International Joint Conference on Artificial Intelligence*, pages 458–463, Sidney, Australia, 1991.

[Donini *et al.*, 1992a] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, Werner Nutt, and Andrea Schaerf. Adding epistemic operators to concept languages. In *Proceedings of KR-92, 3nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 342–353, Cambridge, MA, 1992.

[Donini *et al.*, 1992b] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. From subsumption to instance checking. Technical Report 15.92, Dipartimento di Informatica e Sistemistica, Università di Roma “La Sapienza”, Roma, Italy, 1992.

[Hughes and Cresswell, 1984] George E. Hughes and Maxwell J. Cresswell. *A companion to modal logic.* Methuen, London, UK, 1984.

[Meghini *et al.*, 1993] Carlo Meghini, Fabrizio Sebastiani, Umberto Straccia, and Costantino Thanos. A model of information retrieval based on a terminological logic. In *Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval*, pages 298–307, Pittsburgh, PA, 1993.

[Patel-Schneider, 1989] Peter F. Patel-Schneider. Undecidability of subsumption in NIKL. *Artificial Intelligence*, 39:263–272, 1989.

[Schaerf, 1994] Andrea Schaerf. Reasoning with individuals in concept languages. *Data and Knowledge Engineering*, 13:141–176, 1994.

[Schild, 1991] Klaus Schild. A correspondence theory for terminological logics: preliminary report. In *Proceedings of IJCAI-91, 12th International Joint Conference on Artificial Intelligence*, pages 466–471, Sidney, Australia, 1991.

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

[Schmidt-Schauß, 1989] Manfred Schmidt-Schauß. Subsumption in KL-ONE is undecidable. In *Proceedings of KR-89, 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–431, Toronto, Ontario, 1989.

[Sebastiani and Straccia, 1991] 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.

[Sebastiani, 1994] Fabrizio Sebastiani. A probabilistic terminological logic for modelling information retrieval. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 122–130, Dublin, IRL, 1994.

[Vardi and Wolper, 1986] Moshe Vardi and P. Wolper. Automata-theoretic techniques for modal logics of programs. *Journal of Computer and Systems Science*, 32:183–221, 1986.