GOCCIOLA: Generating New Knowledge by Combining Concepts in Description Logics of Typicality

Antonio Lieto[1], Federico Perrone[1], and Gian Luca Pozzato[1]*

Dipartimento di Informatica, Università di Torino, Italy - {antonio.lieto@, federico.perrone@edu.,gianluca.pozzato@} unito.it

Abstract. In this work we describe GOCCIOLA (Generating knOwledge by Concept Combination In descriptiOn Logics of typicAlity), a tool for the dynamic generation of novel knowledge by exploiting a recently introduced extension of a Description Logic of typicality able to combine prototypical descriptions of concepts. Given a goal expressed as a set of properties, in case an intelligent agent cannot find a concept in its initial knowledge base able to fulfill all these properties, GOCCIOLA exploits the Description Logic $\mathbf{T}^{\mathsf{CL}}$ in order to find two concepts whose creative combination satisfies the goal. The knowledge base of the agent is then extended by the prototype resulting from the concept combination, and the combined concept represents the solution for the initial goal. We have tested GOCCIOLA on some paradigmatic examples in the literature of automatic generation of knowledge, and we have shown that it seems to be a promising instrument to tackle the problem of implementing a dynamic generation of novel knowledge obtained through a process of commonsense reasoning.

1 Introduction

A challenging problem in Artificial Intelligence concerns the capability of an intelligent agent to achieve its goals when its knowledge base does not contain enough information to do that. In this line of research, existing goal-directed systems usually implement a re-planning strategy in order to tackle the problem. This is systematically performed by either an external injection of novel knowledge or as the result of a communication with another intelligent agent [1].

In this work, we propose an alternative approach, consisting in a dynamic and automatic generation of novel knowledge obtained through a process of commonsense reasoning. The idea is as follows: given an intelligent agent and a set of *goals*, if it is not able to achieve them from an initial knowledge base, then it tries to dynamically generate new knowledge by *combining* available information. Novel information will be then used to extend the initial knowledge base in order to achieve the goals. As an example, suppose that an intelligent agent is aware of the facts that, normally, coffee contains caffeine and is a hot

* The author is partially supported by the project "METALLIC #2: METodi di prova per il ragionamento Automatico per Logiche non-cLassIChe #2", INdAM GNCS 2019.

beverage, that the chocolate with cream is normally sweet and has a taste of milk, whereas Limoncello is not a hot beverage (normally, it is served chilled). Both coffee and Limoncello are after meal drinks. Cold winters in Turin suggest to have a hot after-meal drink, also being sweet and having taste of milk. None of the concepts in the knowledge base of the agent are able to achieve the goal on their own, however, the combination between coffee and chocolate with cream provides a solution.

In this paper we introduce GOCCIOLA, a tool following this approach in the context of Description Logics (from now on, DLs for short). DLs are one of the most important formalisms of knowledge representation and are at the base of the languages for building ontologies such as OWL. In this respect, we exploit the Description Logic $\mathbf{T}^{\mathsf{CL}}$, recently introduced in order to account for the phenomenon of concept combination of prototypical concepts [2]. The logic $\mathbf{T}^{\mathsf{CL}}$ relies on the logic of typicality $\mathcal{ALC} + \mathbf{T}_{\mathbf{R}}^{RaCl}$ [3], whose semantics is based on the notion of rational closure, as well as on the DISPONTE semantics of probabilistic DLs [4], and is equipped with a cognitive heuristic used by humans for concept composition. In this logic, typicality inclusions of the form $p :: \mathbf{T}(C) \sqsubseteq D$ are used to formalize that "we believe with degree p about the fact that typical Cs are Ds". As in the distributed semantics, this allows us to consider different scenarios containing only some typicality inclusions, each one having a suitable probability. Such scenarios are then used to ascribe typical properties to a concept C obtained as the combination of two concepts, revising the initial knowledge base with the addiction of typical properties of C. In the example, the revised knowledge base provided by the logic $\mathbf{T}^{\mathsf{CL}}$ contains typical properties of the combination of coffee and chocolate with cream, which suggests to consider a beverage corresponding to the famous Turin drink known as *Bicerín* (little glass), made by coffee, chocolate and cream.

The plan of this short paper is as follows. In Section 2 we briefly recall the DL for concept combination $\mathbf{T}^{\mathsf{CL}}$. In Section 3 we provide a formal description of the problem of dynamic knowledge generation in the context of the logic $\mathbf{T}^{\mathsf{CL}}$. In Section 4 we introduce the system GOCCIOLA, and we show that it is a promising candidate to tackle such a problem by means of some paradigmatic examples. We conclude in Section 5 with some pointers to future issues.

2 Concept Combination in DLs of Typicality: the Logic $\mathbf{T}^{\mathsf{CL}}$

In [2] we have introduced a nonmonotonic Description Logic of typicality called $\mathbf{T}^{\mathsf{CL}}$ (typicality-based compositional logic). This logic combines two main ingredients. The first one relies on the DL of typicality $\mathcal{ALC} + \mathbf{T}_{\mathbf{R}}^{RaCl}$ introduced in [3], which allows to describe the *protoype* of a concept. In this logic, "typical" properties can be directly specified by means of a "typicality" operator $\mathbf{T}$ enriching the underlying DL, and a TBox can contain inclusions of the form $\mathbf{T}(C) \sqsubseteq D$ to represent that "typical Cs are also Ds". As a difference with standard DLs, in the logic $\mathcal{ALC} + \mathbf{T}_{\mathbf{R}}^{RaCl}$ one can consistently express exceptions and reason about defeasible inheritance as well. For instance, a knowledge base can

consistently express that "normally, athletes are fit", whereas "sumo wrestlers usually are not fit" by $\mathbf{T}(\mathit{Athlete}) \sqsubseteq \mathit{Fit}$ and $\mathbf{T}(\mathit{SumoWrestler}) \sqsubseteq \neg \mathit{Fit}$, given that $\mathit{SumoWreslter} \sqsubseteq \mathit{Athlete}$. The semantics of the $\mathbf{T}$ operator is characterized by the properties of *rational logic* [5], recognized as the core properties of nonmonotonic reasoning. $\mathcal{ALC} + \mathbf{T}^{RaCl}_{\mathbf{R}}$ is characterized by a minimal model semantics corresponding to an extension to DLs of a notion of *rational closure* as defined in [5] for propositional logic: the idea is to adopt a preference relation among $\mathcal{ALC} + \mathbf{T}^{RaCl}_{\mathbf{R}}$ models, where intuitively a model is preferred to another one if it contains less exceptional elements, as well as a notion of *minimal entailment* restricted to models that are minimal with respect to such preference relation. As a consequence, $\mathbf{T}$ inherits well-established properties like *specificity* and *irrelevance*: in the example, the logic $\mathcal{ALC} + \mathbf{T}^{RaCl}_{\mathbf{R}}$ allows us to infer $\mathbf{T}(\mathit{Athlete} \sqcap \mathit{Bald}) \sqsubseteq \mathit{Fit}$ (being bald is irrelevant with respect to being fit) and, if one knows that Hiroyuki is a typical sumo wrestler, to infer that he is not fit, giving preference to the most specific information.

As a second ingredient, we have considered a distributed semantics similar to the one of probabilistic DLs known as DISPONTE [4], allowing to label inclusions $\mathbf{T}(C) \sqsubseteq D$ with a real number between 0.5 and 1, representing its degree of belief/probability. In a slight extension of the above example, we can express a degree of belief in the typicality inclusions about athletes and sumo wrestlers: we believe with a probability of 80% that, normally, athletes are fit whereas sumo wrestlers are not; furthermore, we believe that athletes are usually young with a higher degree of 95%. This is formalized by the following knowledge base: (1) $\mathit{SumoWrestler} \sqsubseteq \mathit{Athlete}$; (2) $0.8 :: \mathbf{T}(\mathit{Athlete}) \sqsubseteq \mathit{Fit}$; (3) $0.8 :: \mathbf{T}(\mathit{SumoWrestler}) \sqsubseteq \neg \mathit{Fit}$; (4) $0.95 :: \mathbf{T}(\mathit{Athlete}) \sqsubseteq \mathit{YoungPerson}$. We consider eight different scenarios, representing all possible combinations of typicality inclusion: as an example, $\{((2), 1), ((3), 0), ((4), 1)\}$ represents the scenario in which (2) and (4) hold, whereas (3) does not. We equip each scenario with a probability depending on those of the involved inclusions: the scenario of the example, has probability 0.8×0.95 (since 2 and 4 are involved) $\times(1 - 0.8)$ (since 3 is not involved) $= 0.152 = 15.2\%$. Such probabilities are then taken into account in order to choose the most adequate scenario describing the prototype of the combined concept.

The logic $\mathbf{T}^{\mathsf{CL}}$ seems to be a promising candidate in order to tackle the problem of concept combination. Combining the typical knowledge of pre-existing concepts is among the most creative cognitive abilities exhibited by humans. This generative phenomenon highlights some crucial aspects of the knowledge processing capabilities in human cognition and concerns high-level capacities associated to creative thinking and problem solving. Dealing with this problem requires, from an AI perspective, the harmonization of two conflicting requirements that are hardly accommodated in symbolic systems (including formal ontologies [6]): the need of a syntactic and semantic compositionality (typical of logical systems) and that one concerning the exhibition of typicality effects. According to a well-known argument [7], in fact, prototypes are not compositional. The argument runs as follows: consider a concept like *pet fish*. It results from the

composition of the concept *pet* and of the concept *fish*. However, the prototype of *pet fish* cannot result from the composition of the prototypes of a pet and a fish: e.g. a typical pet is furry and warm, a typical fish is grayish, but a typical pet fish is neither furry and warm nor grayish (typically, it is red).

Given a KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{R}$ is the set of standard (rigid) inclusions of $\mathcal{ALC}$, $\mathcal{T}$ is the set of typicality inclusions, and $\mathcal{A}$ is the ABox, and given two concepts C_H and C_M occurring in $\mathcal{K}$, the logic $\mathbf{T}^{\mathsf{CL}}$ allows defining a prototype of the compound concept C as the combination of the HEAD C_H, the dominant element in the combination, and the MODIFIER C_M, where the typical properties of the form $\mathbf{T}(C) \sqsubseteq D$ (or, equivalently, $\mathbf{T}(C_H \sqcap C_M) \sqsubseteq D$) to ascribe to the concept C are obtained by considering blocks of scenarios with the same probability, in decreasing order starting from the highest one. We first discard all the inconsistent scenarios, then:

- we discard those scenarios considered as *trivial*, consistently inheriting all the properties from the HEAD from the starting concepts to be combined. This choice is motivated by the challenges provided by task of common-sense conceptual combination itself: in order to generate plausible and creative compounds it is necessary to maintain a level of surprise in the combination. Thus both scenarios inheriting all the properties of the two concepts and all the properties of the HEAD are discarded since prevent this surprise;
- among the remaining ones, we discard those scenarios inheriting properties from the MODIFIER in conflict with properties that could be consistently inherited from the HEAD;
- if the set of scenarios of the current block is empty, i.e. all the scenarios have been discarded either because trivial or because preferring the MODIFIER, we repeat the procedure by considering the block of scenarios, having the immediately lower probability.

Remaining scenarios are those selected by the logic $\mathbf{T}^{\mathsf{CL}}$. The ultimate output of our mechanism is a knowledge base in the logic $\mathbf{T}^{\mathsf{CL}}$ whose set of typicality properties is enriched by those of the compound concept C. Given a scenario w satisfying the above properties, we define the properties of C as the set of inclusions $p :: \mathbf{T}(C) \sqsubseteq D$, for all $\mathbf{T}(C) \sqsubseteq D$ that are entailed from w in the logic $\mathbf{T}^{\mathsf{CL}}$. The probability p is such that:

- if $\mathbf{T}(C_H) \sqsubseteq D$ is entailed from w, that is to say D is a property inherited either from the HEAD (or from both the HEAD and the MODIFIER), then p corresponds to the degree of belief of such inclusion of the HEAD in the initial knowledge base, i.e. $p : \mathbf{T}(C_H) \sqsubseteq D \in \mathcal{T}$;
- otherwise, i.e. $\mathbf{T}(C_M) \sqsubseteq D$ is entailed from w, then p corresponds to the degree of belief of such inclusion of a MODIFIER in the initial knowledge base, i.e. $p : \mathbf{T}(C_M) \sqsubseteq D \in \mathcal{T}$.

The knowledge base obtained as the result of combining concepts C_H and C_M into the compound concept C is called *C-revised* knowledge base, and it is defined as follows:

$$\mathcal{K}_C = \langle \mathcal{R}, \mathcal{T} \cup \{p : \mathbf{T}(C) \sqsubseteq D\}, \mathcal{A} \rangle,$$

for all D such that either $\mathbf{T}(C_H) \sqsubseteq D$ is entailed in w or $\mathbf{T}(C_M) \sqsubseteq D$ is entailed in w, and p is defined as above. In [2] we have shown that reasoning in the logic $\mathbf{T}^{\mathsf{CL}}$ remains in the same complexity class of standard $\mathcal{ALC}$ Description Logics, namely that reasoning in $\mathbf{T}^{\mathsf{CL}}$ is ExpTime-complete.

3 Generating New Knowledge by Concept Combination in $\mathbf{T}^{\mathsf{CL}}$

We exploit the logic $\mathbf{T}^{\mathsf{CL}}$ in order to tackle the following problem: given a knowledge base $\mathcal{K}$ in the Description Logic $\mathbf{T}^{\mathsf{CL}}$, an intelligent agent has to achieve a goal $\mathcal{G}$ intended as a set of concepts $\{D_1, D_2, \ldots, D_n\}$. More precisely, the agent has to find a *solution* for the goal, namely a concept C such that, for all properties D_i, it holds that either $\mathcal{K} \models C \sqsubseteq D_i$ or $\mathcal{K} \models \mathbf{T}(C) \sqsubseteq D_i$ in the logic of typicality $\mathcal{ALC} + \mathbf{T}^{RaCl}_{\mathbf{R}}$. If $\mathcal{K}$ does not contain any solution for the goal, then the agent tries to generate a *new concept* by combining two existing ones C_1 and C_2 by means of the logic $\mathbf{T}^{\mathsf{CL}}$: C is then considered a solution for the goal if, considering the $(C_1 \sqcap C_2)$-revised knowledge base $\mathcal{K}_C$ extending $\mathcal{K}$, we have that, for all properties D_i, it holds that either $\mathcal{K}_C \models C \sqsubseteq D_i$ or $\mathcal{K}_C \models \mathbf{T}(C) \sqsubseteq D_i$ in the logic of typicality $\mathcal{ALC} + \mathbf{T}^{RaCl}_{\mathbf{R}}$.

This is formally defined as follows:

Definition 1. *Given a knowledge base $\mathcal{K}$ in the logic $\mathbf{T}^{\mathsf{CL}}$, let $\mathcal{G}$ be a set of concepts $\{D_1, D_2, \ldots, D_n\}$ called* goal. *We say that a concept C is a* solution *to the goal $\mathcal{G}$ if either:*

- *for all $D_i \in \mathcal{G}$, either $\mathcal{K} \models C \sqsubseteq D_i$ or $\mathcal{K} \models \mathbf{T}(C) \sqsubseteq D_i$ in the logic $\mathbf{T}^{\mathsf{CL}}$*

or

- *C corresponds to the combination of two concepts C_1 and C_2 occurring in $\mathcal{K}$, i.e. $C \equiv C_1 \sqcap C_2$, and the C-revised knowledge base $\mathcal{K}_C$ provided by the logic $\mathbf{T}^{\mathsf{CL}}$ is such that, for all $D_i \in \mathcal{G}$, either $\mathcal{K}_C \models C \sqsubseteq D_i$ or $\mathcal{K}_C \models \mathbf{T}(C) \sqsubseteq D_i$ in the logic $\mathbf{T}^{\mathsf{CL}}$.*

Let us conclude this section by formalizing the example of the Introduction.

Example 1. In the example of the Introduction, suppose that $\mathcal{K}$ contains the information that, normally, coffee contains caffeine and is a hot beverage; moreover, we have that the chocolate with cream is normally sweet and has a taste of milk, whereas Limoncello is not a hot beverage (normally, it is served chilled). Both coffee and Limoncello are after meal drinks. We can represent these information as follows:

$0.9 :: \mathbf{T}(\mathit{Coffee}) \sqsubseteq \mathit{AfterMealDrink}$
$0.8 :: \mathbf{T}(\mathit{Coffee}) \sqsubseteq \mathit{WithCaffeine}$
$0.85 :: \mathbf{T}(\mathit{Coffee}) \sqsubseteq \mathit{HotBeverage}$
$\mathit{Limoncello} \sqsubseteq \mathit{AfterMealDrink}$
$0.9 :: \mathbf{T}(\mathit{Limoncello}) \sqsubseteq \neg \mathit{HotBeverage}$
$0.65 :: \mathbf{T}(\mathit{ChocolateWithCream}) \sqsubseteq \mathit{Sweet}$
$0.95 :: \mathbf{T}(\mathit{ChocolateWithCream}) \sqsubseteq \mathit{TasteOfMilk}$

Cold winters in Turin suggest to have a hot after-meal drink, also being sweet and having taste of milk. We can then define a goal $\mathcal{G}$ as

$$\mathcal{G} = \{AfterMealDrink, HotBeverage, Sweet, TasteOfMilk\}.$$

None of the concepts in the knowledge base represent a solution for the problem. However, the combination between the concepts *Coffee* and *ChocolateWithCream* represents a solution. Indeed, the revised knowledge base obtained by exploiting the logic $\mathbf{T}^{\mathsf{CL}}$ to combine these concepts allows the agent to extend its knowledge with the following typicality inclusions:

$0.9 :: \mathbf{T}(Coffee \sqcap ChocolateWithCream) \sqsubseteq AfterMealDrink$
$0.85 :: \mathbf{T}(Coffee \sqcap ChocolateWithCream) \sqsubseteq HotBeverage$
$0.65 :: \mathbf{T}(Coffee \sqcap ChocolateWithCream) \sqsubseteq Sweet$
$0.95 :: \mathbf{T}(Coffee \sqcap ChocolateWithCream) \sqsubseteq TasteOfMilk$

providing a solution to the goal corresponding to the famous Turin drink known as *Bicerín* (little glass).

4 The System GOCCIOLA

In this Section we describe GOCCIOLA, a preliminary implementation of a system able to extend the knowledge of an agent in order to fulfill a set of properties representing the goal that the agent wants to achieve. GOCCIOLA is implemented in Python and its current version, along with the files for the examples presented in this paper, are available at http://di.unito.it/gocciola. The architecture of the system GOCCIOLA is shown in Figure 1.

As an example, let us consider the goal: *object, cutting, graspable*, in other words our agent is looking for an object being graspable and which is able to cut. The initial knowledge base is formalized in the language of the logic $\mathbf{T}^{\mathsf{CL}}$ and it is stored in a suitable file. Rigid properties, holding for all individuals of a given class, are stored as pairs object-property, whereas typical properties are formalized as triples object-property-probability. We have considered an extension with probabilities of a portion of the ontology `opencyc`[1] As an example, the concept *Vase* is stored as follows (on the right the corresponding knowledge base in $\mathbf{T}^{\mathsf{CL}}$):

vase, object	$Vase \sqsubseteq Object$
vase, high convexity	$Vase \sqsubseteq HighConvexity$
vase, ceramic, 0.8	$0.8 :: \mathbf{T}(Vase) \sqsubseteq Ceramic$
vase, to put plants, 0.9	$0.9 :: \mathbf{T}(Vase) \sqsubseteq ToPutPlants$
vase, to contain objects, 0.9	$0.9 :: \mathbf{T}(Vase) \sqsubseteq ToContainObjects$
vase, graspable, 0.9	$0.9 :: \mathbf{T}(Vase) \sqsubseteq Graspable$

To run GOCCIOLA, the user has to invoke the Python interpreter on the file *main.py*, which consults the initial knowledge base and the goal to achieve

[1] https://github.com/asanchez 75/opencyc/blob/master/opencyc-latest.owl.gz.

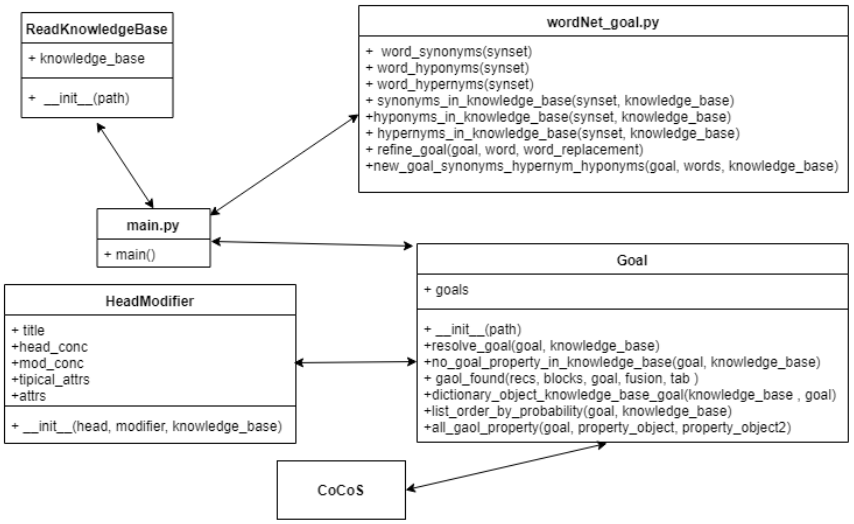

Fig. 1. The architecture of the system GOCCIOLA.

(classes *ReadKnowledgeBase* and *Goal*, respectively). First of all, the method *resolve_goal* checks whether the knowledge base contains a concept C immediately satisfying it, i.e. exhibiting all the concepts of the goal either as rigid or typical properties: in this case, the system is done, and GOCCIOLA ends its computation by suggesting such a concept C as the solution. In case C does not exist, the system GOCCIOLA tries to extend the knowledge base of the agent by looking for at least two concepts, C_1 and C_2, whose combination via $\mathbf{T}^{\text{CL}}$ generates a concept C' satisfying the goal. More in detail:

- GOCCIOLA computes a list of concepts of the initial knowledge base satisfying at least a property of the goal. As an example, suppose that the following inclusions belong to the knowledge base:

  $Spoon \sqsubseteq Graspable$
  $0.85 \;::\; \mathbf{T}(Spoon) \sqsubseteq \neg Cutting$
  $0.9 \;::\; \mathbf{T}(Vase) \sqsubseteq Graspable$
  $Vase \sqsubseteq Object$

  Both *Vase* and *Spoon* are included in the list of candidate concepts to be combined;
- for each item in the list of candidate concepts to be combined, GOCCIOLA computes a rank of the concept as the sum of the probabilities of the properties also belonging to the goal, assuming a score of 1 in case of a rigid property. In the example, *Vase* is ranked as $0.9 + 1 = 1.9$, since both *Graspable* and *Object* are properties belonging to the goal: for the former we take the probability 0.9 of the typicality inclusion $\mathbf{T}(Vase) \sqsubseteq Graspable$, for the latter we provide a score of 1 since the property $Vase \sqsubseteq Object$ is rigid.

Concerning the concept *Spoon*, GOCCIOLA computes a rank of 1: indeed, the only inclusion matching the goal is the rigid one $Spoon \sqsubseteq Graspable$;
- GOCCIOLA checks whether the concept obtained by combining the two candidate concepts with the highest ranks, C_1 and C_2, is able to satisfy the initial goal. GOCCIOLA computes a double attempt, by considering first C_1 as the HEAD and C_2 as the MODIFIER and, in case of failure, C_2 as the HEAD and C_1 as the MODIFIER.

In order to combine the two candidate concepts C_1 and C_2, GOCCIOLA exploits the system COCOS [8], a tool generating scenarios and choosing the selected one(s) according to the logic $\mathbf{T}^{\mathsf{CL}}$. The current version of the system is implemented in Pyhton and exploits the translation of an $\mathcal{ALC} + \mathbf{T}_{\mathbf{R}}^{RaCl}$ knowledge base into standard $\mathcal{ALC}$ introduced in [3] and adopted by the system RAT-OWL [9]. COCOS makes use of the library `owlready2` [2] that allows one to rely on the services of efficient DL reasoners, e.g. the HermiT reasoner. GOCCIOLA also exploits WordNet sysnsets in order to extend its search space in case of a failure. In detail, if the goal contains properties not belonging to the initial knowledge base, GOCCIOLA looks for hypernyms or hyponyms in order to rewrite such properties.

We have tested GOCCIOLA by asking it to solve some well established and paradigmatic examples from the literature. We have considered a knowledge base extending `opencyc` and including, among others, the following inclusions:

$Stone \sqsubseteq MineralAggregate$
$0.7 :: \mathbf{T}(Stone) \sqsubseteq Roundish$
$0.7 :: \mathbf{T}(Stone) \sqsubseteq Greyish$
$0.7 :: \mathbf{T}(Stone) \sqsubseteq BuildingArrowHeads$
$0.8 :: \mathbf{T}(Stone) \sqsubseteq ShapingObjects$
$0.7 :: \mathbf{T}(Stone) \sqsubseteq Cutting$
$0.6 :: \mathbf{T}(Stone) \sqsubseteq Support$
$0.8 :: \mathbf{T}(Stone) \sqsubseteq StrikeAtDistance$
$0.9 :: \mathbf{T}(Stone) \sqsubseteq Graspable$
$0.7 :: \mathbf{T}(Stone) \sqsubseteq Narrow$

$Shelf \sqsubseteq Object$
$0.8 :: \mathbf{T}(Shelf) \sqsubseteq Wood$
$0.9 :: \mathbf{T}(Shelf) \sqsubseteq Rectangular$
$0.8 :: \mathbf{T}(Shelf) \sqsubseteq Containment$
$0.8 :: \mathbf{T}(Shelf) \sqsubseteq Support$

$0.8 :: \mathbf{T}(Stump) \sqsubseteq Wood$
$0.7 :: \mathbf{T}(Stump) \sqsubseteq Medium$
$0.8 :: \mathbf{T}(Stump) \sqsubseteq Linear$
$0.7 :: \mathbf{T}(Stump) \sqsubseteq LiftingFromTheGround$
$0.7 :: \mathbf{T}(Stump) \sqsubseteq Support$

$RubberBand \sqsubseteq Object$
$RubberBand \sqsubseteq Plastic$
$0.9 :: \mathbf{T}(RubberBand) \sqsubseteq Propeller$
$0.9 :: \mathbf{T}(RubberBand) \sqsubseteq LaunchingObjectsAtDistance$
$0.7 :: \mathbf{T}(RubberBand) \sqsubseteq Small$

We have first asked GOCCIOLA to find a solution for the goal

$$\mathcal{G}_1 = \{Object, Cutting, Graspable\},$$

i.e. our intelligent agent is looking for a graspable object able to cut. In this case, GOCCIOLA proposes the combination $Stone \sqcap Stump$ as a solution, thus

[2] https://pythonhosted.org/Owlready2/

suggesting a combined concept resembling a knife with a wood handle. It is worth noticing that, if the initial knowledge base would have contained the inclusion $Stone \sqsubseteq Object$, then GOCCIOLA would have suggested that the concept *Stone* is a straightforward solution for $\mathcal{G}_1$: indeed, all concepts in $\mathcal{G}_1$ would be either rigid or typical properties of such a concept.

We have then queried GOCCIOLA with the goal

$$\mathcal{G}_2 = \{Object, Graspable, LaunchingObjectsAtDistance\},$$

i.e. the agent is looking for a graspable object able to launch at distance. In this case, GOCCIOLA asks COCOS to combine the concepts *Stone* and *RubberBand*, being those with the highest rank with respect to $\mathcal{G}_2$. The ($Stone \sqcap RubberBand$)-revised knowledge base suggested by adopting *Stone* as the HEAD is such that all the properties of both concepts are considered, with the exception of *Support*, therefore the knowledge base of the agent is extended (among the others) by the following inclusions:

$$0.9 \;::\; \mathbf{T}(Stone \sqcap RubberBand) \sqsubseteq Graspable$$
$$0.9 \;::\; \mathbf{T}(Stone \sqcap RubberBand) \sqsubseteq LaunchingObjectsAtDistance$$

and the combination $Stone \sqcap RubberBand$ is a solution for the goal $\mathcal{G}_2$ (it is worth noticing that $Stone \sqcap RubberBand \sqsubseteq Object$ is inherited since $Stone \sqcap RubberBand \sqsubseteq RubberBand$ and $RubberBand \sqsubseteq Object$ is a rigid property).

We have considered a third goal

$$\mathcal{G}_3 = \{Object, Support, LiftingFromTheGround\},$$

for which GOCCIOLA provides a solution corresponding to the concept obtained by combining *Shelf* and *Stump*. Notice that also $Stump \sqcap RubberBand$ would be a solution: however, GOCCIOLA gives preference to the concept *Shelf* because it has a higher rank with respect to the goal, being also, normally, a member of the concept *Support*.

5 Conclusions

In this short paper we have presented GOCCIOLA, a first implementation of a procedure whose aim is to dynamically extend a Description Logics knowledge base by exploiting conceptual combination. GOCCIOLA relies on COCOS, a tool for combining concepts in the logic $\mathbf{T}^{\mathsf{CL}}$. In future research, we aim at studying the application of optimization techniques in [10, 11] in order to improve the efficiency of COCOS and, as a consequence, of GOCCIOLA.

The logic $\mathbf{T}^{\mathsf{CL}}$ underlying GOCCIOLA is also able to combine more than two concepts at a time, as well as to involve compound concepts (and not only atomic ones) in a concept combination. We aim at extending GOCCIOLA in order to also exploit this features. Moreover, in future works, we plan to consider the case in which GOCCIOLA is able to provide a partial solution, satisfying a proper subset of the initial goals.

References

1. Aha, D.W.: Goal reasoning: Foundations, emerging applications, and prospects. AI Magazine **39**(2) (2018)
2. Lieto, A., Pozzato, G.L.: A description logic of typicality for conceptual combination. In: Proc. of ISMIS 2018. Volume 11177 of LNAI., Springer (2018) 189–199
3. Giordano, L., Gliozzi, V., Olivetti, N., Pozzato, G.L.: Semantic characterization of Rational Closure: from Propositional Logic to Description Logics. Artif. Intelligence **226** (2015) 1–33
4. Riguzzi, F., Bellodi, E., Lamma, E., Zese, R.: Reasoning with probabilistic ontologies. In: Proc. of IJCAI 2015, AAAI Press (2015) 4310–4316
5. Lehmann, D., Magidor, M.: What does a conditional knowledge base entail? Artificial Intelligence **55**(1) (1992) 1–60
6. Frixione, M., Lieto, A.: Towards an extended model of conceptual representations in formal ontologies: A typicality-based proposal. J. UCS **20**(3) (2014) 257–276
7. Osherson, D.N., Smith, E.E.: On the adequacy of prototype theory as a theory of concepts. Cognition **9**(1) (1981) 35–58
8. Lieto, A., Pozzato, G.L., Valese, A.: COCOS: a typicality based concept combination system. In: Proc. of CILC 2018. Volume 2214 of CEUR Workshop Proceedings. (2018) 55–59
9. Giordano, L., Gliozzi, V., Pozzato, G.L., Renzulli, R.: An efficient reasoner for description logics of typicality and rational closure. In: Proc. of Description Logics 2017. Volume 1879 of CEUR Workshop Proceedings., CEUR-WS.org (2017)
10. Alberti, M., Bellodi, E., Cota, G., Riguzzi, F., Zese, R.: cplint on SWISH: probabilistic logical inference with a web browser. Intelligenza Artificiale **11**(1) (2017) 47–64
11. Bellodi, E., Lamma, E., Riguzzi, F., Zese, R., Cota, G.: A web system for reasoning with probabilistic OWL. Journal of Software: Practice and Experience **47**(1) (2017) 125–142