Evolutionary Algorithms for Reasoning in Fuzzy Description Logics with Fuzzy Quantifiers

Mauro Dragoni
Università degli Studi di Milano
Dipartimento di Tecnologie dell'Informazione
Via Bramante 65, I-26013 Crema (CR), Italy
dragoni@dti.unimi.it

Andrea G. B. Tettamanzi
Università degli Studi di Milano
Dipartimento di Tecnologie dell'Informazione
Via Bramante 65, I-26013 Crema (CR), Italy
andrea.tettamanzi@unimi.it

ABSTRACT

The task of reasoning with fuzzy description logics with fuzzy quantification is approached by means of an evolutionary algorithm. An essential ingredient of the proposed method is a heuristic, implemented as an intelligent mutation operator, which observes the evolutionary process and uses the information gathered to guess at the mutations most likely to bring about an improvement of the solutions. The viability of the method is demonstrated by applying it to reasoning on a resource sheduling problem.

Categories and Subject Descriptors

I.2.3 [**Deduction and Theorem Proving**]: Uncertainty, "fuzzy," and probabilistic reasoning

General Terms

Algorithms

Keywords

Evolutionary Algorithms, Fuzzy Logic, Description Logics, Fuzzy Quantification

1. INTRODUCTION

Representing knowledge about a domain and using that representation to reason about and solve problems in that domain is central to many disciplines of computer science. Recently, a new family of knowledge representation formalisms which strike a delicate balance between expressive power and computational complexity has emerged, namely description logics (DLs) [1]. All reasoning tasks in DLs can be reduced to checking whether a set of assertions is satifiable.

Even more recently, these formalisms have been extended with fuzzy constructs and semantics [14]. The further extension of fuzzy DLs with fuzzy quantifiers [12] greatly increases the expressive power of the language, but makes reasoning particularly hard [13].

GECCO'07 , July 7-11, 2007, London, England, United Kingdom.

This paper proposes an alternative to exact, classical tableau-like reasoning algorithms consisting of the use of evolutionary algorithms for checking the satisfiability of a set of fuzzy assertions.

Evolutionary algorithms for the satisfiability problem in propositional logic (SAT), a well-known NP-complete problem [8], have quite a long history. Recent developments, such as GASAT [10], have been shown to be competitive with the best state-of-the-art algorithms for SAT. Of particular interest is the fuzzy approach by Pedrycz and colleagues [15], which provided a source of inspiration for this work.

The paper is organized as follows: first we present the basic concepts of DLs in Section 2, then we introduce fuzzy quantifiers in Section 3. Section 4 describes the problem of checking the satisfiability of a set of fuzzy DL assertions with fuzzy quantifiers, Section 5 illustrates the evolutionary algorithm used to approach the problem, followed in Section 6 by a case study which demonstrates the viability of the approach. Finally, Section 7 concludes.

2. DESCRIPTION LOGICS

Description Logics is the most recent name for a family of knowledge representation (KR) formalisms that represent the knowledge of an application domain (the "world"), by first defining the relevant concepts of the domain (its terminology), and then using these concepts to specify properties of objects and individuals occurring in the domain (the world description).

As their name indicates, DLs are equipped with a formal, logic-based semantics and support inference patterns named classification of concepts and individuals.

Classification of concepts determines subconcept/superconcept relationships between the concepts of a given terminology, and thus allows one to structure the terminology in the form of a subsumption hierarchy.

Classification of individuals determines whether a given individual is always an instance of a certain concept.

Decidability and complexity of the inference problems depend on the expressive power of the DL at hand. On the one hand, very expressive DLs are likely to have inference problems of high complexity, or they may even be undecidable. On the other hand, very weak DLs may not be sufficiently expressive to represent the important concepts of a given application. Investigating this trade-off between the expressivity of DLs and the complexity of their reasoning problems has been one of the most important issues in DL research.

2.1 Basic Formalism

A KR system based on DLs provides facilities to set up knowledge bases, to reason about their content, and to manipulate them.

A knowledge base (KB) comprises at least two components, the TBox and the ABox. The TBox introduces the terminology, while the ABox contains assertions about named individuals in terms of the TBox. Statements in the TBox and in the ABox are equivalent formulae in first-order logic or, in some cases, a slight extension of it.

Important problems for an ABox are to find out whether its set of assertions is consistent and whether the assertions in the ABox entail that a particular individual is an instance of a given concept description.

Satisfiability checks of descriptions and consistency checks of sets of assertions are useful to determine whether a knowledge base is meaningful at all.

2.2 Fuzzy Description Logics

Typically, DLs are limited to dealing with crisp concepts. However, many useful concepts that are needed by an intelligent system do not have well defined boundaries. That is, more often than not, the concepts encountered in the real world do not have precisely defined criteria of membership, i.e. they are vague concepts rather than precise concepts. Fuzzy extensions of DLs have been proposed for dealing with vague concepts [14].

2.2.1 Fuzzy Sets

A fuzzy set is a class of objects with a continuum of grades of membership. The key is to consider a membership of an element in a set not as an all-or-nothing concept, but as a gradual attribute of elements which can be anything from "definitely not belonging" to "definitely belonging" to a set. In formal terms, this is achieved by replacing the characteristic function with a *membership* function.

The notions of inclusion, union, intersection, complement, relation, etc., are extended to such sets, and various properties of these notions in the context of fuzzy sets are estabilished.

Let U be a universe of objects and denote by x a generic element of U. A *fuzzy set* A in U is defined by its membership function $A : U \to [0, 1]$. The value $A(x)$ is to be understood as the degree to which x belongs to A. If A is a crisp set, A reduces to the usual characterstic function.

A notational convention for fuzzy sets when the universe of discourse U is discrete is, for fuzzy set A,

$$A = \frac{A(x_1)}{x_1} + \frac{\mu_A(x_2)}{x_2} + \cdots = \sum_i \frac{A(x_i)}{x_i}. \tag{1}$$

When U is continuous, the fuzzy set A is denoted by

$$A = \int_{x \in U} \frac{A(x)}{x}. \tag{2}$$

Both notations are nothing more than a formal device and the fractions do not have to be interpreted as divisions but just as ordered pairs, while the + does not stand fo algebraic sum but rather for a function-theoretic union.

3. FUZZY QUANTIFIERS

The concept of fuzzy linguistic quantifier is due to L. A. Zadeh [16]. Fuzzy quantifiers are linguistic labels representing imprecise quantities or percentages. This concept is important since quantification is one of the most employed tools in human reasoning.

It is usual to distinguish two basic types of quantifiers:

- Absolute quantifiers, which express vague quantities (e.g., "around 2") or quantity intervals (e.g., "approximately between 1 and 3"). They are represented as fuzzy subsets of the non-negative integers.
- Relative quantifiers, which express fuzzy percentages and are represented by fuzzy subsets of the real unit interval, although in practice only rational values make sense. This category includes the standard quantifiers $\exists$ and $\forall$ from predicate logic.

There are two ways to interpret the semantics of a relative quantifier. We say a quantifier Q is interpreted in an exclusive or in a wide way, depending on whether we interpret its meaning as "exactly Q" or "at least Q", respectively. However, the wide interpretation is the most employed, not only because it is more common to interpret a quantifier this way intuitively, but probably because many methods work only with monotonically increasing quantifiers.

3.1 Cardinality and Fuzzy Quantification

Crisp quantification is strongly linked to crisp cardinality since a crisp quantifier Q represents a crisp subset of absolute (values in $\mathbb{N}$) or relative (values in $\mathbb{Q} \cap [0, 1]$) cardinalities, we call $S(Q)$. By the same token, fuzzy quantification is strongly linked to fuzzy cardinality.

We may consider two different kinds of cardinalities: *absolute* cardinality measures the number of elements in a set, while *relative* cardinality measures the percentage of elements of one set that also belong in another set (called *referential*).

The close relationship between fuzzy quantification and fuzzy cardinality is discussed in depth in [13].

3.2 Absolute Cardinality

The most widely used definition of fuzzy set cardinality introduced in [11] is the following: given a fuzzy set F,

$$|F| = \sum_{x \in \Delta} F(x), \tag{3}$$

which is, in general, a real number, and not an integer as it is the case with classical set cardinality. This definition lends itself to many objections and leads to paradoxes. A more satisfactory cardinality measure is ED [4]: the fuzzy cardinality of a set G is the set $ED(G)$ defined for each $0 \leq k \leq |\text{supp}(G)|$ as

$$ED(G)(k) = \begin{cases} \alpha_i - \alpha_{i+1} & \alpha_i \in \Lambda(G) \text{ and } |G_{\alpha_i}| = k \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

with $\Lambda(G) = \{\alpha_1, \ldots, \alpha_p\} \cup \{1\}$ the level set of G, and $\alpha_i > \alpha_{i+1}$ for every $i \in \{1, ..., p\}$, and $\alpha_{p+1} = 0$.

3.3 Relative cardinality

An extension to the ED measure to cope with relative cardinality is ER [4]: the fuzzy relative cardinality of a set G with respect to a set F is the set $ER(G/F)$ defined for each $0 \leq q \leq 1$ as

$$ER(G/F)(q) = \sum_{\alpha_i \mid C(G/F, \alpha_i) = q} (\alpha_i - \alpha_{i+1}) \tag{5}$$

with $\Lambda(F) \cup \Lambda(G \cap F) = \{\alpha_1, \ldots, \alpha_p\}$ and $\alpha_i > \alpha_{i+1}$ for every $i \in \{1, \ldots, p\}$, and $\alpha_0 = 1, \alpha_{p+1} = 0$.

3.4 Evaluation of Quantified Sentences

Quantified sentences are natural language sentences involving fuzzy linguistic quantifiers, and therefore they express claims about the (fuzzy) quantity or percentage of elements of a (possibly fuzzy) set that verify a certain imprecise property.

According to Zadeh [16], there are two main types of quantified sentences, whose general structure is the following:

Type I sentences: Q of X are G
Type II sentences: Q of F are G

where Q is a linguistic quantifier, X is a crisp finite set, and F and G are two fuzzy subsets of X that represent imprecise properties.

The evaluation of a quantified sentence is the process of calculating its fuzzy accomplishment degree. There are several methods available in the literature (some methods are discussed in [5], recent developments are [9, 6]).

In order to extend fuzzy DLs with absolute and relative quantifiers, we shall employ method GD [5]. This method calculates the accomplishment degree of a quantified sentence "Q of F are G" as the compatibility degree between the fuzzy relative cardinality measure $ER(G/F)$ [4] and Q (if Q is relative) or between its absolute counterpart $ED(G \cap F)$ [4] and Q (if Q is absolute). A convenient formulation of GD is the following [5]:

The method GD obtains the evaluation of a quantified sentence "Q of F are G" as

$$GD_Q(G/F) = \sum_{\alpha_i \in \Lambda(G/F)} (\alpha_i - \alpha_{i+1})\, Q\left(\frac{|(G \cap F)_{\alpha_i}|}{|F_{\alpha_i}|}\right) \quad (6)$$

for relative quantifiers, and

$$GD_Q(G/F) = \sum_{\alpha_i \in \Lambda(G/F)} (\alpha_i - \alpha_{i+1})\, Q\left(|(F \cap G)_{\alpha_i}|\right) \quad (7)$$

for absolute ones, where $F \cap G$ is computed using the minimum, and $\Lambda(G/F) = \Lambda(G \cap F) \cup \Lambda(F)$. We label these values as $\Lambda(G/F) = \{\alpha_1, \ldots, \alpha_p\}$ with $\alpha_i > \alpha_{i+1}$ for every $i \in \{1, \ldots, p\}$ and $\alpha_{p+1} = 0$.

4. THE PROBLEM

A KR system based on DLs is able to perform specific kinds of reasoning. The purpose of a KR system goes beyond storing concept definitions and assertions. A knowledge base, comprising a TBox and an ABox, has a semantics that makes it equivalent to a set of axioms in first-order predicate logic. Thus, like any set of axioms, it contains implicit knowledge that can be made explicit through inferences.

The different kinds of reasoning performed by a DL system are defined as logical inferences.

When a knowledge engineer models a domain, she constructs a terminology, say $\mathcal{T}$, by defining new concepts, possibily in terms of others that have been defined before. During this process, it is important to find out whether a newly defined concept makes sense or whether it is contradictory. From a logical point of view, a concept makes sense if there is some interpretation that satisfies the axioms of $\mathcal{T}$ such that the concept denotes a nonempty set in that interpretation. A concept with this property is said to be satisfiable with respect to $\mathcal{T}$ and unsatisfiable otherwise.

While in crisp DLs it is true that all reasoning tasks can be reduced to the consistency check for ABoxes, the same is not always necessarily true for fuzzy DLs. Straccia [14] examines three reasoning tasks, namely:

- the Fuzzy Entailment Problem;
- the Fuzzy Subsumption Problem;
- the Best Truth-Value Bound Problem,

and he reduces them to the Entailment Problem, for which he provides a PSPACE-complete calculus based on constraint propagation. An important observation about Straccia's definition of these reasoning tasks is that, because they deal with TVBs, the first two decision tasks are binary, i.e., yes or no, true-or-false, crisp.

Bonatti and Tettamanzi [2] consider a further reasoning task, namely concept consistency check, and they prove some unusual complexity results, some of which are reminiscent of properties of disjunctive logic programs.

Reasoning algorithms for expressive crisp and fuzzy DLs are characterized by exponential time complexity. To be sure, optimized inference engines are available which are capable of carrying out reasoning tasks in most cases with acceptable performance. However, one could wonder to what extent this is due to the fact that knowledge bases are carefully written to avoid known sources of intractability, like, for example, disjunctions.

The injection of fuzzy quantification into fuzzy DLs makes a good case for the observation that as we increase the expressivity of DLs, sooner or later we will get to a point where the use we will be able to make of the knowledge thus expressed will be throttled by the performance of exact reasoning algorithms.

On the other hand, one of the revolutionary ideas of soft computing, of which fuzzy logic is one ingredient, is that by sacrificing some unnecessary precision or exactness, one can still achieve useful results at a lower cost. A direct consequence of this line of reasoning would be to give up the logical completeness of a knowledge representation formalism, which comes at an exponentially increasing computational cost, for some sort of probabilistic decidability of the kind guaranteed, e.g., by an evolutionary approach.

By using an evolutionary approach, the probem of checking the consistency of an ABox can be relaxed into an optimization problem whose goal is to find an interpretation that minimizes the number of violated constraints, where the number of violated constraints is the number of assertions that are not satisfied by a given interpretation.

In the case of fuzzy DLs, a given interpretation may satisfy an assertion to a certain degree. Therefore, the ABox consistency check can be reformulated as an optimization problem, where the fuzzy interpretation is sought for that maximizes the degree of satisfaction of all the assertions in the ABox.

A crucial issue is how one measures the satisfaction of a set of fuzzy assertions. From a logical standpoint, given a fuzzy interpretation $\mathcal{I}$ and an ABox $\mathcal{A}$,

$$\mathcal{I} \models \mathcal{A} \equiv \bigwedge_{\phi \in \mathcal{A}} \mathcal{I} \models \phi, \quad (8)$$

which, in fuzzy terms, if one uses min as the t-norm operator, would yield

$$\text{truth}(\mathcal{I} \models \mathcal{A}) = \min_{\phi \in \mathcal{A}} \{\text{truth}(\mathcal{I} \models \phi)\}. \quad (9)$$

Although logically thorough, this definition cannot be used as the optimization criterion, because it does not allow us to distinguish between interpretations that satisfy different numbers of assertions, as long as the degrees of satisfaction of the least satisfied assertions coincide.

5. THE EVOLUTIONARY ALGORITHM

EAs [7, 3] are a broad class of stochastic optimization algorithms, inspired by biology. An EA maintains a population of candidate solutions for the problem at hand, and makes it evolve by iteratively applying a set of stochastic operators, known as *mutation*, *recombination*, and *selection*.

Mutation randomly perturbs a candidate solution; recombination decomposes two distinct solutions and then randomly mixes their parts to form novel solutions; and selection replicates the most successful solutions found in a population at a rate proportional to their relative quality. The resulting process tends to find, given enough time, globally optimal solutions to the problem much in the same way as in nature populations of organisms tend to adapt to their surrounding environment.

5.1 Fitness Function

The fitness function developed for this research takes into account two criteria used to evaluate to which extent a solution represents an acceptable fuzzy interpretation:

- ABox satisfiability — this is a value related to the number of assertions that are satisfied by an interpretation; this criterion has a fundamental importance, since the goal is to obtain phenotypes that totally satisfy the ABox, thus, selecting solutions that have high satisfiability allows the algorithm to approach the optimal solution;
- size of the interpretation represented by a phenotype — this gives a bias toward simpler fuzzy interpretations.

These two criteria do not have the same influence on the fitness value, given that the ABox satisfiability is much more important than the interpretation size.

The equation of the fitness function can be explained as:

$$F = \tau + e^{-ks}, \quad (10)$$

where τ is the sum of the degrees of satisfaction of all the assertions in the ABox, $k \geq 1$ is a parameter used to tune the impact of the interpretation-size criterion, and s is the size of the interpretation, i.e., the total number of individual names it uses. The term e^{-ks} is always less than one, insuring that, roughly speaking, the satisfaction-degree criterion always has the upper hand over the interpretation-size criterion; the latter becomes relevant only when two solutions are indistinguishable with respect to the former.

5.2 Representation

An individual encodes a (partial) fuzzy interpretation $\mathcal{I}$ for the ABox at hand.

A fuzzy interpretation consists of a finite set Δ of individuals (the ones explicitly appearing in the ABox, plus any number of other "extra" individuals whose existence is postulated) and of assignments of degrees of truth to all possible fuzzy assertions involving the individuals in Δ.

$$\begin{aligned}
\mathsf{Person}(\mathsf{P1}) &= 0.56 \\
\mathsf{Female}(\mathsf{P1}) &= 0.36 \\
\mathsf{Person}(\mathsf{P2}) &= 0.72 \\
\mathsf{Female}(\mathsf{P2}) &= 0.16 \\
\mathsf{Person}(\mathsf{P3}) &= 0.87 \\
\mathsf{Female}(\mathsf{P3}) &= 0.73 \\
\mathsf{Person}(\mathsf{P4}) &= 0.22 \\
\mathsf{Female}(\mathsf{P4}) &= 0.66 \\
\mathsf{hasChild}(\mathsf{P1},\mathsf{P2}) &= 0.25 \\
\mathsf{hasChild}(\mathsf{P1},\mathsf{P3}) &= 0.28 \\
\mathsf{hasChild}(\mathsf{P1},\mathsf{P4}) &= 0.52 \\
\mathsf{hasChild}(\mathsf{P2},\mathsf{P1}) &= 0.32 \\
\mathsf{hasChild}(\mathsf{P2},\mathsf{P3}) &= 0.73 \\
\mathsf{hasChild}(\mathsf{P2},\mathsf{P4}) &= 0.11 \\
\mathsf{hasChild}(\mathsf{P3},\mathsf{P1}) &= 0.63 \\
\mathsf{hasChild}(\mathsf{P3},\mathsf{P2}) &= 0.72 \\
\mathsf{hasChild}(\mathsf{P3},\mathsf{P4}) &= 0.39 \\
\mathsf{hasChild}(\mathsf{P4},\mathsf{P1}) &= 0.06 \\
\mathsf{hasChild}(\mathsf{P4},\mathsf{P2}) &= 0.93 \\
\mathsf{hasChild}(\mathsf{P4},\mathsf{P3}) &= 0.88
\end{aligned}$$

Figure 1: An example of fuzzy interpretation, involving four individuals, two concepts, and one role.

An example of fuzzy phenotype is shown in Figure 1.

The meaning of an assignment like $\mathsf{Person}(\mathsf{P3}) = 0.87$ is that individual $\mathsf{P3}$ is an instance of Person with a truth degree of 0.87, while the meaning of an assignment like $\mathsf{hasChild}(\mathsf{P3},\mathsf{P4}) = 0.39$ is that $\mathsf{P4}$ is a filler of role $\mathsf{hasChild}$ for $\mathsf{P3}$ with a truth degree of 0.39.

An important thing is that all phenotypes have the same number of assignments because in a fuzzy environment we have to specify a truth assignment for all possible assertions.

The number of individuals used to complete the ABox is given by the sum of the minimal number of individuals required to satisfy every fuzzy quantifier present in that ABox:

- for relative quantifiers: given $q_0^Q, q_1^Q \in [0,1] \cap \mathbb{Q}$ such that $Q(q_0^Q) = 0$ and $Q(q_1^Q) = 1$, there exist n_0^Q, d_0^Q, n_1^Q, $d_1^Q \in \mathbb{N}$ such that $q_0^Q = \frac{n_0^Q}{d_0^Q}$ and $q_1^Q = \frac{n_1^Q}{d_1^Q}$; the minimal number of individuals is $lcm(d_0^Q, d_1^Q)$.
- for absolute quantifiers: given $m_0^Q, \ldots, m_n^Q \in \mathbb{N}$ such that $Q(m_0^Q) = 1, \ldots, Q(m_n^Q) = 1$, the minimal number of individuals is $\min_i\{m_i^Q\}$.

In assertions with nested quantifiers, i.e., where fuzzy quantifiers occur and the filler of a quantified role contains in turn a quantified role, the number of individididuals is obtained by multiplying the number of individuals of every nested fuzzy quantifier.

Given a fuzzy interpretation $\mathcal{I}$, with the number of individuals calculated above, for an ABox with c concepts, and r roles,

- for each individual, there must be one assignment for every concept, thus nc concept assignments;
- for each couple of individuals, there must be one degree of membership in every role relation, thus $r\frac{n(n-1)}{2}$.

P1 : Person ⊓ Female ⊓ ∃hasChild.Person
P2 : Person
P3 : Person ⊔ Female

Figure 2: A sample ABox.

In total,

$$\|\mathcal{I}\| = nc + r\frac{n(n-1)}{2} = cO(n) + rO(n^2). \quad (11)$$

5.3 Satisfiability of Fuzzy Interpretations

Once all phenotypes are created, their fitness is calculated according to Equation 10, on the basis of the fuzzy semantics of the basic constructs of the fuzzy DL language:

$$\bot^{\mathcal{I}}(a) = 0; \quad (12)$$

$$\top^{\mathcal{I}}(a) = 1; \quad (13)$$

$$(C \sqcap D)^{\mathcal{I}}(a) = \min\{C^{\mathcal{I}}(a), D^{\mathcal{I}}(a)\}; \quad (14)$$

$$(C \sqcup D)^{\mathcal{I}}(a) = \max\{C^{\mathcal{I}}(a), D^{\mathcal{I}}(a)\}; \quad (15)$$

$$(\neg C)^{\mathcal{I}}(a) = 1 - C^{\mathcal{I}}(a); \quad (16)$$

$$(Q_{\mathrm{abs}} R.C)^{\mathcal{I}}(a) = \int_0^1 Q_{\mathrm{abs}}(|G_a|)\, d\tau; \quad (17)$$

$$(Q_{\mathrm{rel}} R.C)^{\mathcal{I}}(a) = \int_0^1 Q_{\mathrm{rel}}\left(\lim_{n\to\infty} \frac{|G_a|}{|F_a|}\right) d\tau, \quad (18)$$

where

$$G_a = \{b : R^{\mathcal{I}}(a,b) \geq \tau \wedge C^{\mathcal{I}}(b) \geq \tau\} \quad (19)$$

$$F_a = \{b : R^{\mathcal{I}}_n(a,b) \geq \tau\} \quad (20)$$

and $\{R^{\mathcal{I}}_n\}_{n=1,2,\ldots}$ is a sequence of fuzzy relations such that

1. for all $n \geq 1$, $R^{\mathcal{I}}_n \subseteq R^{\mathcal{I}}_{n+1}$;
2. $\lim_{n\to\infty} R^{\mathcal{I}}_n = R^{\mathcal{I}}$;
3. for all $n \geq 1$, $\mathrm{supp}(R^{\mathcal{I}}_n)$ is a finite set.

To understand with an example how this works, we consider the fuzzy interpretation of Figure 1 and we use it to evaluate the assertions contained in the ABox of Figure 2.

As for individual P2, the evaluation of its assertion will be the truth degree of Person(P2), in this case 0.72. To evaluate the assertion involving P3, we apply Equation 15, yielding a truth degree of 0.87.

To evaluate the assertion involving P1, before applying Equation 14, one has to evaluate ∃hasChild.Person(P1). Now, ∃ can be regarded as a relative quantifier such that $\exists(0) = 0$ and $\exists(x) = 1$ for $x > 0$; therefore, we have to apply Equation 18, which in this special case reduces to

$$(\exists R.C)^{\mathcal{I}}(a) = \sup_{b\in\Delta^{\mathcal{I}}} \min\{R^{\mathcal{I}}_a(b), C^{\mathcal{I}}(b)\}. \quad (21)$$

In practice, for every couple $(\mathsf{P1}, b)$, $b \in \Delta^{\mathcal{I}}$, we take the minimum of the truth degrees of hasChild$(\mathsf{P1}, b)$ and Person(b); of all such degrees, we take the greatest, in this case 0.28.

5.4 Selection

Two different selection strategies have been tested:

1. truncation selection with a dynamic threshold, whereby all individuals whose fitness is below the best fitness so far are discarded; this strategy pushes selection pressure to a maximum;
2. the standard fitness-proportionate selection, whereby at all individuals has been assigned a probability of selection proportionate with its fitness value.

5.5 Recombination

It is important to note that all individuals (i.e., fuzzy interpretations) have the same number of "genes" (i.e., truth assignments); furthermore, genes at the same position in different individuals correspond to truth assignments to the same assertion (membership of an individual in a concept or of a pair of individuals in a role).

Three different recombination operators have been tried:

1. fitness-proportionate uniform crossover, whereby a new individual is created by selecting for each position the corresponding gene from either parent with a probability proportional to its fitness;
2. single-point crossover;
3. fitness-weighted intermediate recombination, whereby every gene of the offspring is a weighted average of the corresponding gene in both parents, the weights being proportional to their fitness.

5.6 Mutation

Two different mutation operators have been implemented:

1. one gene is picked at random and is assigned a new truth degree; an interesting point about this mutation strategy is that, by modifying a single truth assignment, useful information about the impact of individual genes on fitness can be gleaned; identifying assignments that have a high impact on fitness drives the application of the intelligent mutation, as explained in Section 5.7;
2. all truth assignments are perturbed by applying small random variations.

5.7 Intelligent Mutation

Preliminary tests showed symptoms of stagnation: after a number of generations, the average fitness of the population started to oscillate without improving. To help evolution escape from these traps, we designed an improvement operator whose intent is to look at what happened in the last n generations, to try to determine those assertions in the ABox whose satisfaction is critical, and to concentrate search on truth assignments that involve those critical assertions.

When such a critical set is identified, the operator carries out a systematic search to determine an assignment that satisfies the critical assertions.

The pseudocode of this operator is shown in Figure 3.

During execution, this algorithm maintains in an array the list of all concepts and, for every concept, the amount of mutation that has been applied on all assertions containing that concept (i.e., if, on concept C, the algorithm applied a mutation of 0.1 in the first generation and of 0.05 in the second generation, the amount will be 0.15).

If the increase of the chosen concept is not positive, the algorithm calls function FreezeCurrentConcept, which marks that concept as "frozen" and uses the remaining concepts. Function SearchConceptToMutate gets the non-frozen concepts that have the greatest mutation amount; the

```
IntelligentMutation(currentGeneration, Δ)
  currentGeneration contains all genotypes
  Δ is the difference between the current and the previous fitness
  Check if current fitness is less of the last fitness value, i.e., the analyzed concept has changed
  if(Δ < 0)
    C is the concept to analyze:
    C ← SearchConceptToMutate()
    Get all assertions that contain the analyzed concept:
    A ← SelectAssertions(C, currentGeneration)
    Apply mutation on all assertions:
    ApplyMutation(A)
    Replace the old assertions with the mutated assertions:
    InsertMutatedIndividuals(A, currentGeneration)
    Save the mutation parameters for the next check:
    SaveMutationHistory()
  else
    If the previous mutation did not increase the generation fitness, the algorithm changes the concept to analyze:
    FreezeCurrentConcept()
  end if
end
```

Figure 3: Pseudocode of the intelligent mutation.

rationale is to exploit all the potential of a concept before freezing it.

Function SelectAssertions retrieves all assertions that involve the selected concept; subsequently, function ApplyMutation applies a small perturbation to all assertions formerly selected.

Function InsertMutatedIndividuals inserts the individuals that have been mutated into the current generation; finally, the algorithm saves, by function SaveMutationHistory, information about the changes made.

6. A CASE STUDY

The case study used for testing our approach considers a resource allocation problem where a company needs to create n teams, where every team consists of a given number of people that have some characteristics; the goal is to find the minimal set of people with the required characteristics for each team.

This kin dof example has been choosen to show in which type of problem, i.e. multi-objects, this approach can be used.

The assumptions made are the following:

- a person may have multiple skills
- a person can be a member of more than one team;
- distinct teams may have the same structure.

The KB used for the case study is shown in Figure 4. The notation used to describe fuzzy quantifiers is the one proposed in [12]: their membership function is expressed by means of one or more "points" of the form $\alpha \triangleleft \beta \triangleright \gamma / x$, where x is the cardinality, β the truth degree of x, while α and γ are the truth degrees of values that tend to x respectively from the left and from the right.

TBox

$$
\begin{aligned}
= 2 &\equiv 0 \triangleleft 1 \triangleright 0/2 \\
\geq 3 &\equiv 0 \triangleleft 1/3 \\
\geq 4 &\equiv 0 \triangleleft 1/4 \\
\leq 3 &\equiv 1 \triangleright 0/3 \\
\leq 4 &\equiv 1 \triangleright 0/4 \\
\mathsf{most} &\equiv 0/u + 0/0 + 1/1 \\
\mathsf{half} &\equiv 0/u + 1/0.5 + 0/1 \\
\mathsf{moreThanHalf} &\equiv 0/u + 1/0.5 \\
\mathsf{exclusive} &\equiv 0/u + 0/0.25 + 1/0.75 + 0.5/1 \\
\mathsf{T1} &\equiv (\mathsf{most})\mathsf{hasCompetence.C1} \sqcap \\
&\quad (= 2)\mathsf{hasCompetence.C2} \sqcap \\
&\quad (\geq 3)\mathsf{hasMember.Person} \\
\mathsf{T2} &\equiv (\mathsf{half})\mathsf{hasCompetence.C8} \sqcap \\
&\quad (= 2)\mathsf{hasCompetence.C4} \sqcap \\
&\quad (\leq 4)\mathsf{hasMember.Person} \\
\mathsf{T3} &\equiv (\mathsf{most})\mathsf{hasCompetence.C1} \sqcap \\
&\quad (= 2)\mathsf{hasCompetence.C5} \sqcap \\
&\quad (\geq 3)\mathsf{hasMember.Person} \\
\mathsf{T4} &\equiv (\mathsf{half})\mathsf{hasCompetence.C8} \sqcap \\
&\quad (= 2)\mathsf{hasCompetence.C5} \sqcap \\
&\quad (\leq 4)\mathsf{hasMember.Person} \\
\mathsf{T5} &\equiv (\mathsf{moreThanHalf})\mathsf{hasCompetence.C7} \sqcap \\
&\quad (\leq 3)\mathsf{hasCompetence.C2} \sqcap \\
&\quad (\geq 4)\mathsf{hasMember.Person} \\
\mathsf{T6} &\equiv (\mathsf{exclusive})\mathsf{hasCompetence.C4} \sqcap \\
&\quad (= 2)\mathsf{hasCompetence.C8} \sqcap \\
&\quad (\leq 3)\mathsf{hasMember.Person}
\end{aligned}
$$

ABox

$$
\begin{aligned}
&\mathsf{T1(TEAM1)} \\
&\mathsf{T2(TEAM2)} \\
&\mathsf{T3(TEAM3)} \\
&\mathsf{T4(TEAM4)} \\
&\mathsf{T5(TEAM5)} \\
&\mathsf{T6(TEAM6)} \\
&\mathsf{T2(TEAMA)} \\
&\mathsf{T5(TEAMB)}
\end{aligned}
$$

Figure 4: The knowledge base of the case study.

Phenotype

$$
\begin{aligned}
\mathsf{T1(TEAM1)} &= 1 \\
\mathsf{T2(TEAM2)} &= 0.731464 \\
\mathsf{T3(TEAM3)} &= 1 \\
\mathsf{T4(TEAM4)} &= 0.842050 \\
\mathsf{T5(TEAM5)} &= 0.902186 \\
\mathsf{T6(TEAM6)} &= 0.866550 \\
\mathsf{T2(TEAMA)} &= 0.822846 \\
\mathsf{T5(TEAMB)} &= 1 \\
F &= 7.165096
\end{aligned}
$$

Figure 5: The assignements for the best interpretation found. This result has been obtained with the operators S1 M1 C2 and with the use of the intelligent mutation. The number of extra individuals used to complete the ABox is 47

6.1 Results

The parameters of the evolutionary algorithm that gave us the best results are:

- Population size: 100
- Number of generations: 1000
- Mutation probability: 20% for all mutation operators
- Absolute mutation variance: 0.2

Table 1 reports a comparison of the results obtained by applying all combinations of the operators to problem of the case study; for every combination 10 runs were executed. Figure 5 shows the fuzzy truth value assigned to each assertion in the ABox by the best interpretation found by the cobination using truncation selection, fitness-proportionate uniform crossover, mutation of type 2, and the intelligent mutation operator.

By observing the results, one can easily notice that, for every combination of operators, the use of the intelligent mutation operator increases the quality of the results. Moreover, it is possible to determine which are the operators that hinder convergence of the algorithm: as a matter of fact, use of type-1 mutation always leades to significantly worse results than use of type-2 mutation; the same occurs as well with fitness-weighted intermediate recombination, which in addition corresponds to the smallest standard deviation of results.

By considering the standard deviation of results it is possible to conclude that type-2 mutation operator is responsible for a greater variability of results than using type-1 mutation.

The best interpretation found, shown in Figure 5, indicates that the ABox is most likely satisfiable, even if not all assertions have truth degree of 1. However, it is well-known evolutionary algorithms are very good at locating the optimum but quite poor at reaching it exactly; even if the evolutionary algorithm is not capable of giving a categorical positive answer, it would be relatively easy to work out an interpretation that completely satisfies the ABox starting from the near-optimal solution it provides.

Ops.	Par.	Max	Avg.	St.Dev.
S1C1M1	Best Fit.	2,857160	2,733131	0,071974
	Avg. Fit.	2,095875	2,008061	0,043817
S1C1M1I	Best Fit.	3,143334	2,985866	0,067993
	Avg. Fit.	2,466893	2,435623	0,024624
S1C1M2	Best Fit.	5,920843	5,584489	0,225852
	Avg. Fit.	4,983412	4,724199	0,191450
S1C1M2I	Best Fit.	**7,165096**	**6,430284**	0,394379
	Avg. Fit.	**6,361265**	**5,707091**	0,353380
S1C2M1	Best Fit.	2,742702	2,669760	0,047594
	Avg. Fit.	2,202552	2,042425	0,065414
S1C2M1I	Best Fit.	3,038692	2,935052	0,061584
	Avg. Fit.	2,504489	2,438439	0,043745
S1C2M2	Best Fit.	5,239907	4,675295	0,330141
	Avg. Fit.	4,578072	4,084134	0,315970
S1C2M2I	Best Fit.	6,039121	5,591465	0,290325
	Avg. Fit.	5,371177	4,972441	0,244340
S1C3M1	Best Fit.	3,232268	3,186808	**0,022853**
	Avg. Fit.	2,718579	2,688568	0,012567
S1C3M1I	Best Fit.	3,525225	3,441470	0,045593
	Avg. Fit.	3,018134	2,987624	0,022938
S1C3M2	Best Fit.	3,374401	3,257314	0,068882
	Avg. Fit.	3,069478	2,981942	0,045513
S1C3M2I	Best Fit.	4,333799	3,700037	0,333557
	Avg. Fit.	4,015075	3,386439	0,301728
S2C1M1	Best Fit.	2,433477	2,375808	0,049278
	Avg. Fit.	1,699291	1,655869	0,021764
S2C1M1I	Best Fit.	2,978694	2,801559	0,093780
	Avg. Fit.	2,323473	2,264725	0,049713
S2C1M2	Best Fit.	5,771483	5,184973	0,351993
	Avg. Fit.	4,837855	4,412332	0,260642
S2C1M2I	Best Fit.	6,333782	5,903895	0,352390
	Avg. Fit.	5,394255	5,017405	0,280357
S2C2M1	Best Fit.	2,448946	2,355289	0,049234
	Avg. Fit.	1,645381	1,628530	**0,010597**
S2C2M1I	Best Fit.	2,862535	2,780701	0,053279
	Avg. Fit.	2,346400	2,284865	0,043611
S2C2M2	Best Fit.	4,442869	3,855138	0,331556
	Avg. Fit.	3,803897	3,312906	0,296881
S2C2M2I	Best Fit.	5,904936	5,477814	0,324424
	Avg. Fit.	4,871351	4,560966	0,304289
S2C3M1	Best Fit.	3,230038	3,162692	0,038278
	Avg. Fit.	2,662676	2,642710	0,012890
S2C3M1I	Best Fit.	3,633676	3,455184	0,105414
	Avg. Fit.	2,983431	2,962759	0,016261
S2C3M2	Best Fit.	3,333886	3,198483	0,066056
	Avg. Fit.	3,167546	3,002575	0,080804
S2C3M2I	Best Fit.	4,116326	3,460513	0,414641
	Avg. Fit.	3,857482	3,241055	0,384085

Table 1: Result obtained from the execution of all tests. Where Sx is the selection operator used, Cx is the crossover operation used, Mx is the mutation operator used and [I] if the Intelligent mutation is used or not. For every combination we execute 10 runs.

Run	Max	Run	Max
1	2,066617	6	2,237126
2	2,299563	7	2.174939
3	2,107354	8	1.955346
4	2,057115	9	1.999077
5	2,011992	10	1.964541

Table 2: Results obtained from the execution of 10 runs of the Monte Carlo method. Every run uses 100,000 individuals, i.e., the number of individuals that are evaluated overall by the evolutionary algorithm in every case study run.

It is impossible to carry out a significant comparison of the above-described approach with other methods, because no alternative exact or approximated algorithm for checking the consistency of an ABox in fuzzy DLs with fuzzy quantification has been proposed so far. In order to assess the effectiveness of the approach, however, an option is to compare it with a pure Monte Carlo method which generates, for every run, the same number of individuals that are evaluated in one run of the evolutionary algorithm.

The results of such Monte Carlo method are shown in Table 2. What those results tell is, at least, that our approach performs significantly better than a purely random search, given a fixed amount of computational resources.

7. CONCLUSIONS

The ABox consistency check problem for fuzzy DLs with fuzzy quantification has been approached by means of evolutionary algorithms, and the viability of the method has been demonstrated. An essential ingredient of the proposed method is a heuristics, implemented as an intelligent (i.e., problem-aware) mutation operator, which observes the evolutionary process and uses the information gathered to guess at the mutations most likely to bring about an improvement of the solutions.

The approach has been validated, with promising results, on a test case abstracted from a real-world resource scheduling problem arising in a software development setting. The next step will be to apply this approach to non-trivial real-world problems, in order to obtain further insights leading to an improvement of the heuristics.

8. REFERENCES

[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook: Theory, implementation and applications*. Cambridge, 2003.

[2] P. A. Bonatti and A. G. B. Tettamanzi. Some complexity results on fuzzy description logics. In A. P. V. Di Gesù, F. Masulli, editor, *WILF 2003 International Workshop on Fuzzy Logic and Applications*, LNCS 2955, Berlin, 2004. Springer Verlag.

[3] K. A. DeJong. *Evolutionary Computation: A unified approach*. MIT Press, Cambridge, MA, 2002.

[4] M. Delgado, M. Martín-Bautista, D. Sánchez, and M. Vila. A probabilistic definition of a nonconvex fuzzy cardinality. *Fuzzy Sets and Systems*, 126(2):41–54, 2002.

[5] M. Delgado, D. Sánchez, and M. Vila. Fuzzy cardinality based evaluation of quantified sentences. *International Journal of Approximate Reasoning*, 23:23–66, 2000.

[6] F. Díaz-Hermida, A. Bugarín, P. Cariñena, and S. Barro. Voting-model based evaluation of fuzzy quantified sentences: a general framework. *Fuzzy Sets and Systems*, 146(1):97–120, 2004.

[7] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, Berlin, 2003.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[9] I. Glöckner. Fundamentals of fuzzy quantification: Plausible models, constructive principles, and efficient implementation. Technical Report TR2002-07, Technical Faculty, University Bielefeld, 33501 Bielefeld, Germany, 2002.

[10] F. Lardeux, F. Saubion, and J.-K. Hao. GASAT: A genetic local search algorithm for the satisfiability problem. *Evolutionary Computation*, 14(2):223–253, 2006.

[11] A. D. Luca and S. Termini. A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory. *Information and Control*, 20:301–312, 1972.

[12] D. Sánchez and A. G. B. Tettamanzi. Generalizing quantification in fuzzy description logics. In *Proceedings 8th Dortmund Fuzzy Days*, Dortmund, Germany, 2004.

[13] D. Sánchez and A. G. B. Tettamanzi. Fuzzy quantification in fuzzy description logics. In E. Sanchez, editor, *Fuzzy Logic and the Semantic Web*, Capturing Intelligence. Elsevier, Amsterdam, 2006.

[14] U. Straccia. Reasoning within fuzzy description logics. *Journal of Artificial Intelligence Research*, 14:137–166, 2001.

[15] G. S. Witold Pedrycz and O. Shai. Genetic-fuzzy approach to the boolean satisfiability problem. *IEEE Transactions on evolutionary computation*, 6(5), 2002.

[16] L. A. Zadeh. A computational approach to fuzzy quantifiers in natural languages. *Computing and Mathematics with Applications*, 9(1):149–184, 1983.