Abstract In this paper we discuss two formal models apt for a search and communication in a ‘multi-agent world’, namely TIL and EL@. Specifying their intersection, we are able to translate and switch between them. Using their union, we extend their functionalities. The main asset of using TIL is a fine-grained rigorous analysis and specification close to natural language. The additional contribution of EL@ consists in modelling multi-criterion aspects of user preferences. Using a simple example throughout the paper, we illustrate the aspects of a multi-criterion search and communication by their analysis and specification in both the systems. The paper is an introductory study aiming at a universal logical approach to the ‘multi-agent world’, which at the same time opens new research problems and trends.

1. Introduction and motivation.

In this paper we discuss two formal models that are relevant in the area of search and communication in the multi-agent world, namely Transparent Intensional Logic (TIL) and a fuzzy variant EL@ of the existential description logic EL (see [2]). Since TIL has been introduced and discussed in the EJC proceedings and EL is a well-known logical system, we are not going to introduce in details the technicalities of them. Instead, we provide just a minimal necessary introduction to keep the paper self-contained and concentrate on the analytic and specification role of these systems in the area of a semantic web search that takes into account specific user fuzzy criteria. By comparing the two formalisms we aim at providing a clue to their integration. Last but not least we’d like to illustrate the assets of a rigorous logical approach to the problem.

The main asset of using TIL is a fine-grained rigorous analysis and specification close to natural language. The additional contribution of EL@ consists in modelling multi-criterion aspects of user preferences. The paper is an introductory study aiming at a universal logical approach to the ‘multi-agent world’, which at the same time opens new research problems and trends. The EL@ logic is a many-valued version of the existential description logic EL (see [2]) where fuzzification concerns only concepts and the logic is enriched with aggregation (see [21]). Specifying the intersection of TIL and EL@, viz. the TIE@L, we are able to translate
and switch between the two systems. Using their union, $\text{TIL} + \text{DL}$, we extend their functionalities. Throughout the paper we use a simple example in order to illustrate basic principles, common features, as well as differences of the two systems.

**Example** Consider a simple communication between three agents, $A$, $B$ and $C$. The agents can be computational, like web services, database engines, query engines, pieces of software, or even human ones. The agent $A$ sends a message to $B$ asking to find a hotel suitable for $A$ (the structure of the message and the meaning of ‘suitable’ will be discussed later). After obtaining an answer the agent $A$ chooses a hotel and sends another message to the agent $C$ asking to seek a suitable parking place close to the chosen hotel. The criteria of $A$ are: hotel price (e.g., as low as possible), hotel distance to a beach (should be as close as possible), hotel year of building (not too old), parking place price and parking place distance (to the hotel). We are going to describe this scenario simultaneously in two formal models: $\text{TIL}$ (Transparent Intensional Logic) and $\text{DL}$ (Description Logic).

Of course, the model can be made more realistic by considering a larger number of agents searching for specific attribute values (this approach is motivated by Fagin in [10]). When needed, we will switch between the levels of granularity in order to go into more details.

Using the DL and/or database notation we are thus going to consider agents of the type User, and the attributes Hotel_Price, Hotel_Beach_Distance, Hotel_Year_of_Construction, Parking_Price, Parking_Distance. Let the values of the attributes (results of the search) be:

<table>
<thead>
<tr>
<th>Hotel</th>
<th>price</th>
<th>distance</th>
<th>yoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>h1</td>
<td>150</td>
<td>300</td>
<td>1980</td>
</tr>
<tr>
<td>h2</td>
<td>200</td>
<td>450</td>
<td>2000</td>
</tr>
</tbody>
</table>

Particular attribute preferences of a user $U$ can be evaluated by assigning the preference degree, a real number in the interval $[0,1]$, to the attribute values. For instance, $\text{cheap}_U(150) = 0.75$, $\text{close}_U(300) = 0.6$, $\text{new}_U(1980) = 0.2$, and similarly for the other values. In this way we obtain fuzzy subsets $\text{cheap}_U$, $\text{close}_U$, $\text{new}_U$ of the attribute domains, which can be recorded in a fuzzy database operation table (see [15]):

<table>
<thead>
<tr>
<th>Hotel</th>
<th>cheap_U</th>
<th>close_U</th>
<th>new_U</th>
</tr>
</thead>
<tbody>
<tr>
<td>h1</td>
<td>0.75</td>
<td>0.6</td>
<td>0.2</td>
</tr>
<tr>
<td>h2</td>
<td>0.5</td>
<td>0.3</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Our reasoning and decision making is driven not only by the preferences we assign to the values of attributes, but also by the weight we assign to the very criteria of interest. For instance, when being short of money, the low price of the hotel is much more important than its closeness to the beach. When being rich we may prefer a modern high-tech equipped hotel situated on the beach. On the other hand a hotel close to the beach may become totally unattractive in a tsunami-affected area. The multi-criterion decision is thus seldom based on a simple conjunctive or disjunctive combination of the respective factors, and we need an algorithm to compute global user preferences as a composition of particular weighted fuzzy values of the selection criteria. The algorithm can be rather sophisticated. However, for the sake of simplicity, let it be just a weighted average:
\[
@_U(\text{cheap}_U, \text{close}_U, \text{new}_U) = \frac{2 \ast \text{cheap}_U + 3 \ast \text{close}_U + \text{new}_U}{6}
\]

Computing the global degree of preferences of the hotel h₁ for the user U, we obtain:

\[
@_U(0.75, 0.6, 0.2) = \frac{2 \ast 0.75 + 3 \ast 0.6 + 0.2}{6} = \frac{3.5}{6} = 0.58...
\]

Since this value is higher than the value of the hotel h₂, the user U is going to choose h₁. Of course, another user can have different preferences, and also the preferences of one and the same user may dynamically change in time.

Besides the fact that in a multi-agent world we work with vague, fuzzy or uncertain information, we have to take into account also the demand on robustness and distribution of the system. The system has to be fully distributive, and we have to deal with value gaps because particular agents may fail to supply the requested data. On the other hand, in critical and emergency situations, which tend to a chaotic behaviour, the need for an adequate data becomes a crucial point. Therefore the classical systems which are based on the Closed World Assumption are not plausible here. We have to work under the Open World Assumption (OWA), and a lack of knowledge must not yield a collapse of the system.

For instance, it may happen that we are not able to retrieve the distance of the hotel h₁ to the beach, and the available data are as follows:

<table>
<thead>
<tr>
<th>Hotel</th>
<th>price</th>
<th>distance</th>
<th>yoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>h₁</td>
<td>150</td>
<td></td>
<td>1980</td>
</tr>
<tr>
<td>h₂</td>
<td>200</td>
<td>450</td>
<td>2000</td>
</tr>
</tbody>
</table>

There are several possibilities of dealing with lacking data. We may use default values (e.g., average, the best or the worst ones), or treat the missing values as value gaps of partial functions. From the formal point of view, \text{TIL} is a hyper-intensional partial \(\lambda\)-calculus. By ‘hyper-intensional’ we mean the fact that the terms of the ‘language of \text{TIL} constructions’ are not interpreted as the denoted functions, but as algorithmically structured \textit{procedures}, known as \textit{\text{TIL constructions}}, producing the denoted functions as outputs. Thus we can rigorously and naturally handle the terms that are in classical logics ‘non-denoting’, or undefined.\(^1\) in \text{TIL} each term is denoting a full-right entity, namely a construction. Hence (well-typed) terms never lack semantics. It may just happen (in well defined cases) that the denoted procedure fails to produce an output function. And if it does not fail it may happen that the produced function fails to have a value at an argument. These features of \text{TIL} are naturally combined with and completed by the \text{\text{EL}@ fuzzy tools}, in particular the aggregation algorithms.

The paper is organized as follows: Chapter 2 contains brief introductory remarks on \text{TIL}. Chapter 3 introduces the \text{\text{EL}@ description logic}, and Chapter 4 is devoted to the formal description of our motivating examples, which gives us a flavour of the common features of both the models. As a result, in concluding Chapter 5 we outline a possible hybrid system and specify the trends of future research.

\(^1\) For the logic of definedness see [11].
In this Chapter we provide just a brief introductory explanation of the main notions of Transparent Intensional Logic (TIL). For exact definitions and details see, e.g., [5], [7], [8], [19], [20]. TIL approach to knowledge representation can be characterised as the ‘top-down approach’. TIL ‘generalises to the hardest case’ and obtains the ‘less hard cases’ by lifting various restrictions that apply only higher up. This way of proceeding is opposite to how semantic theories tend to be built up. The standard approach consists in beginning with atomic sentences, then proceeding to molecular sentences formed by means of truth-functional connectives or by quantifiers, and from there to sentences containing modal operators and, finally, attitudinal operators.

Thus, to use a simple case for illustration, once a vocabulary and rules of formation have been laid down, a semantics gets off the ground by analysing an atomic sentence as follows:

(1) “Charles selected the hotel $h$”: $S(a, h)$

And further upwards:

(2) “Charles selected the hotel $h$, and Thelma is happy”: $S(a, h) \land H(b)$

(3) “Somebody selected the hotel $h$”: $\exists x S(x, h)$

(4) “Possibly, Charles selected the hotel $h$”: $\Diamond S(a, h)$

(5) “Thelma believes that Charles selected the hotel $h$”: $B(b, S(a, h))$.

In non-hyperintensional (i.e., non-procedural) theories of formal semantics, attitudinal operators are swallowed by the modal ones. But when they are not, we have three levels of granularity: the coarse level of truth-values, the fine-grained level of truth-conditions (propositions, truth-values-in-intension), and the very fine-grained level of hyper-propositions, i.e., constructions of propositions. TIL operates with these three levels of granularity. We start out by analysing sentences from the uppermost end, furnishing them with a hyperintensional\(^2\) semantics, and working our way downwards, furnishing even the lowest-end sentences (and other empirical expressions) with a hyperintensional semantics. That is, the sense of a sentence such as “Charles selected the hotel $h$” is a hyper-proposition, namely the construction of the denoted proposition (i.e., the instruction how to evaluate the truth-conditions of the sentence in any state of affairs).

When assigning a construction to an expression as its meaning, we specify a procedural know-how, which must not be confused with the respective performatory know-how. Distinguishing performatory know-how from procedural know-how, the latter could be characterised “that a knower $x$ knows how $A$ is done in the sense that $x$ can spell out instructions for doing $A$.”\(^3\) For instance, to know what Goldbach Conjecture means is to understand the instruction to find whether ‘all positive even integers $\geq 4$ can be expressed as the sum of two primes’. It does not include either actually finding out (whether it is true or not by following a procedure or by luck) or possessing the skill to do so.\(^4\)

Furthermore, the sentence “Charles selected the hotel $h$” is an ‘intensional context’, in the sense that its logical analysis must involve reference to empirical parameters, in this case both possible worlds and instants of time. Charles only contingently selected the hotel; i.e., he did so only at some worlds and only sometimes. The other reason is because the analysans must

---

2 The term ‘hyperintensional’ has been introduced by Max Cresswell, see [4].
3 See [16, p.6]
4 For details on TIL handling knowledge see [8].
be capable of figuring as an argument for functions whose domain are propositions rather than truth-values. Construing ‘S(a,h)’ as a name of a truth-value works only in the case of (1) and (2). It won’t work in (5), since truth-values are not the sort of thing that can be believed. Nor will it work in (4), since truth-values are not the sort of thing that can be possible.

Constructions are procedures, or instructions, specifying how to arrive at less-structured entities. Being procedures, constructions are structured from the algorithmic point of view, unlike set-theoretical objects. The ‘TIL’ language of constructions is a modified hyper-intensional version of the typed λ-calculus, where Montague-like λ-terms denote, not the functions constructed, but the constructions themselves. Constructions qua procedures operate on input objects (of any type, even on constructions of any order) and yield as output (or, in well defined cases fail to yield) objects of any type; in this way constructions construct partial functions, and functions, rather than relations, are basic objects of our ontology. The choice of types and of constructions is not given once for ever: it depends on the area to be analyzed.

By claiming that constructions are algorithmically structured, we mean the following: a construction C—being an instruction—consists of particular steps, i.e., sub-instructions (or, constituents) that have to be executed in order to execute C. The concrete/abstract objects an instruction operates on are not its constituents, they are just mentioned. Hence objects have to be supplied by another (albeit trivial) construction. The constructions themselves may also be only mentioned: therefore one should not conflate using constructions as constituents of composed constructions and mentioning constructions that enter as input into composed constructions, so we have to strictly distinguish between using and mentioning constructions. Just briefly: Mentioning is, in principle, achieved by using atomic constructions. A construction is atomic if it is a procedure that does not contain any other construction as a used subconstruction (a constituent).

There are two atomic constructions that supply objects (of any type) on which complex constructions operate: variables and trivializations.

Variables are constructions that construct an object dependently on valuation: they v-construct, where v is the parameter of valuations. When X is an object (including constructions) of any type, the Trivialization of X, denoted 0X, constructs X without the mediation of any other construction. 0X is the atomic concept of X: it is the primitive, non-perspectival mode of presentation of X.

There are two compound constructions, which consist of other constructions: Composition and Closure. Composition is the procedure of applying a function f to an argument A, i.e., the instruction to apply f to A to obtain the value (if any) of f at A. Closure is the procedure of constructing a function by abstracting over variables, i.e., the instruction to do so. Finally, higher-order constructions can be used twice over as constituents of composed constructions. This is achieved by a fifth construction called Double Execution.

TIL constructions, as well as the entities they construct, all receive a type. The formal ontology of TIL is bi-dimensional. One dimension is made up of constructions, the other dimension encompasses non-constructions. On the ground level of the type-hierarchy, there are entities unstructured from the algorithmic point of view belonging to a type of order 1. Given a so-called epistemic (or ‘objectual’) base of atomic types (α-truth values, ι-individuals, τ-time moments / real numbers, ω-possible worlds), mereological complexity is increased by the induction rule for forming partial functions: where α, β₁,…,βₙ are types of order 1, the set of partial mappings from β₁×…×βₙ to α, denoted (α β₁…βₙ), is a type of order 1 as well.5

5 TIL is an open-ended system. The above epistemic base {ο, τ, ω} was chosen, because it is apt for natural-language analysis, but the choice of base depends on the area to be analysed.
Constructions that construct entities of order 1 are constructions of order 1. They belong to a type of order 2, denoted by *₁. This type *₁ together with atomic types of order 1 serves as a base for the induction rule: any collection of partial functions, type (α β₁…βₙ), involving *₁ in their domain or range is a type of order 2. Constructions belonging to a type *₂ that identify entities of order 1 or 2, and partial functions involving such constructions, belong to a type of order 3. And so on ad infinitum.

**Definition (Constructions)**

i) Variables x, y, z, ... are constructions that construct objects of the respective types depending on valuations v; they v-construct.

ii) Trivialization: Where X is an object whatsoever (an extension, an intension or a construction), ^X is a construction called trivialization. It constructs X without any change.

iii) Composition: If X v-constructs a function F of a type (α β₁…βₙ), and Y₁,…,Yₙ v-construct entities B₁,…,Bₙ of types β₁,…,βₙ, respectively, then the composition [X Y₁ … Yₙ] is a construction that v-constructs the value (an entity, if any, of type α) of the (partial) function F on the argument (B₁, …, Bₙ). Otherwise the composition [X Y₁ … Yₙ] does not v-construct anything: it is v-improper.

iv) Closure: If x₁, x₂, …,xₘ are pairwise distinct variables that v-construct entities of types β₁, β₂,…, βₘ, respectively, and Y is a construction that v-constructs an entity of type α, then [λx₁…xₘ Y] is a construction called closure, which v-constructs the following function F of the type (α β₁…βₘ), mapping β₁ ×…× βₘ to α: Let B₁,…,Bₘ be entities of types β₁,…,βₘ, respectively, and let v(B₁,x₁,…,Bₘ/xₘ) be a valuation differing from v at most in associating the variables x₁,…xₘ with B₁,…,Bₘ, respectively. Then F associates with the m-tuple (B₁,…,Bₘ) the value v(B₁,x₁,…,Bₘ/xₘ)-constructed by Y. If Y is v(B₁,x₁,…,Bₘ/xₘ)-improper (see iii), then F is undefined on (B₁,…,Bₘ).

v) Double execution: If X is a construction that v-constructs a construction X', then ^X is a construction called double execution. It v-constructs the entity (if any) v-constructed by X'. Otherwise the double execution ^X is v-improper.

vi) Nothing is a construction, unless it so follows from i) through vi).

The notion of construction is a notion that is the most misunderstood notion of those ones used in TIL. Some logicians ask: Are constructions formulae of type-logic? Our answer: No! Another question: Are they denotations of closed formulae? Our answer: No! So a pre-formal, ‘pre-theoretical’ characteristics is needed: constructions are abstract procedures. Question: Procedures are time-consuming, how can they be abstract? Answer: The realization of an algorithm is time-consuming, the algorithm itself is timeless and spaceless.

Question: So what about your symbolic language? Why do you not simply say that its expressions are constructions? Answer: These expressions cannot construct anything they serve only to represent (or encode) constructions.

Question: But you could do it like Montague did: To translate expressions of natural language into the language of intensional logic, and then interpret the result in the standard manner. What you achieve using ‘constructions’ you would get using metalanguage?

Answer(s):

First, Montague and other intensional logics interpret terms of their language as the respective functions, i.e., set-theoretical mappings. However, those mappings are the outputs of executing the respective procedures. Montague does not make it possible to mention the

---

* For details on Montague system see, e.g., [12, pp. 117-220].
procedures as objects *sui generis*, and to make thus a semantic shift to hyperintensions. Yet we do need a hyperintensional semantics. Notoriously well-known are attitudinal sentences which no intensional semantics can properly handle, because its finest individuation is equivalence.\(^7\) Second, our logic is universal: we do not need to work as part-time linguists. Using the ‘language of constructions’ we directly encode constructions.

**Definition ((α)-intension, (α)-extension)**

(α)-intensions are members of a type (αω), i.e., functions from possible worlds to the arbitrary type α. (α)-extensions are members of the type α, where α is not equal to (βω) for any β, i.e., extensions are not functions from possible worlds.

**Remark on notational conventions:** An object A of a type α is called an α-object, denoted A/α. That a construction C \(\rightarrow\) constructs an α-object is denoted C\[\rightarrow\]α. We will often write ‘\(\forall x A\)’, ‘\(\exists x A\)’ instead of ‘\(\forall^0 x A\)’, ‘\(\exists^0 x A\)’, respectively, when no confusion can arise. We also often use an infix notation without trivialisation when using constructions of truth-value functions \(\land\) (conjunction), \(\lor\) (disjunction), \(\supset\) (implication), \(\equiv\) (equivalence) and negation (\(\neg\)), and when using a construction of an identity.

Intensions are frequently functions of a type ((ατω), i.e., functions from possible worlds to chronologies of the type α (in symbols: \(ατω\)), where a chronology is a function of type (ατ). We will use variables \(w, w_1, w_2, \ldots\) as \(\rightarrow\)-constructing elements of type \(ω\) (possible worlds), and \(t, t_1, t_2, \ldots\) as \(\rightarrow\)-constructing elements of type \(τ\) (times). If \(C \rightarrow ατω\) \(\rightarrow\)-constructs an α-intension, the frequently used composition of a form \([[(C w) t]]\), \(\rightarrow\)-constructing the intensional descent of the α-intension, will be abbreviated as \(C wt\).

Some important kinds of intensions are:

- **Propositions**, type \(οτω\). They are denoted by empirical (declarative) sentences.
- **Properties of members of a type α**, or simply α-properties, type \((οα)τω\). General terms (some substantives, intransitive verbs) denote properties, mostly of individuals.
- **Relations-in-intension**, type \((οβ_1…β_m)τω\). For example transitive empirical verbs, also attitudinal verbs denote these relations. Omitting \(τω\) we get the type \((οβ_1…β_m)\) of relations-in-extension (to be met mainly in mathematics).
- **α-roles, offices**, type \(ατω\), where α ≠ \((οβ)\). Frequently \(τω\). Often denoted by concatenation of a superlative and a noun (“the highest mountain”). Individual roles correspond to what Church in [3] called “individual concept”.

The role of the above defined constructions in a communication between agents will be illustrated in Chapter 4, in particular in Paragraph 4.5. Just a note to elucidate the role of Trivialisation and empirical parameters \(w \rightarrow ω, t \rightarrow τ\): The TIL language is not based on a fixed alphabet: the role of formal constants is here played by Trivialisations of non-constructional entities, i.e., the atomic concepts of them. Each agent has to be equipped with a basic ontology, namely the set of atomic concepts he knows. Thus the upper index \(' 0',\) serves as a marker of the atomic concept (like a ‘key-word’) that the agent should know. If they do not, they have to learn it. The lower index \(' w',\) can be understood as an instruction to execute an empirical inquiry (search) in order to obtain the actual current value of an intension, for instance by searching agent’s database or by asking the other agents, or even by means of agent’s sense perception.

---

\(^7\) See [12, p.73]

\(^8\) Collections, sets, classes of ‘α-objects’ are members of type (οα); TIL handles classes (subsets of a type) as characteristic functions. Similarly relations (-in-extension) are of type(s) \((οβ_1…β_m)\).
3 The $\mathcal{EL}^\oplus$ description logic

In a multi-agent world like the semantic web we need to retrieve, process, share or reuse information which is often vague or uncertain. The applications have to work with procedures that deal with the degree of relatedness, similarity or ranking. These motivations lead to the development of the fuzzy description logic (see, [18]). In this chapter we briefly describe a variant of the fuzzy description logic, namely $\mathcal{EL}^\oplus$ (see [21]).

One of the principal sources of fuzziness is user evaluation (preference) of crisp values of attributes. For instance, the hotel price is crisp but user evaluation may lead to a fuzzy predicate like a cheap, moderate, or expensive hotel. User preferences are modelled by linearly ordered set of degrees $T = [0, 1]$ extending classical truth-values. Thus we have:

$$0 = False = \bot = \text{the worst} \in T$$

and

$$1 = True = T = \text{the best} \in T$$

Now when searching a suitable object we have to order the set of available objects according to the user degrees assigned to object-attribute values. Practical experiences have shown that the ordering is seldom based on a conjunctive or disjunctive combination of particular scores. Rather, we need to work with a fuzzy aggregation function that combines generally incomparable sets of values.

The $\mathcal{EL}^\oplus$ logic is in some aspects a weakening of Straccia fuzzy description logic and in some other aspects a strengthening. The restrictions concern using just crisp roles and not using negation. Moreover, quantification is restricted to existential quantifiers. The extension concerns the application of aggregation functions. Thus we loose the ability to describe fuzziness in roles but gain the ability to compute a global user score.

The $\mathcal{EL}^\oplus$ alphabet consists of (mutually disjoint) sets $N_C$ of concept names containing $T$, $N_R$ role names, $N_I$ instance names and constructors containing $\exists$ and a finite set $C$ of combination functions with an arity function $ar : C \rightarrow \{n \in \mathbb{N} : n \geq 2\}$.

Concept descriptions in $\mathcal{EL}^\oplus$ are formed according to the following syntax rules (where $@ \in C$)

$$C \rightarrow T|A|@((C_1, \ldots, C_n)|\exists r.C')$$

The interpretation structures of our description logic $\mathcal{EL}^\oplus$ are parameterized by an ordered set of truth-values $T$ (the degrees of membership to a domain of a fuzzy concept) and a set of $n$-ary aggregation functions over $T$. An interpretation structure $\mathcal{T}$ is thus an algebra

$$\mathcal{T} = \{T, \geq, \{@\oplus : @ \in C\}\},$$

where $(T, \geq, T)$ is an upper complete semilattice with the top element $T$, and $@\oplus : T^{ar(@)} \rightarrow T$ is a lattice of totally continuous (order-preserving) aggregation functions.

A $\mathcal{T}$-interpretation is then a pair $I = \langle \Delta^I, \bullet^I \rangle$, with a nonempty domain $\Delta^I$ and the interpretation of language elements

$$a^I \in \Delta^I, \text{ for } a \in N_I$$

$$A^I : \Delta^I \rightarrow T, \text{ for } A \in N_C \text{ (concepts can be fuzzy, like a suitable hotel)}$$

$$r^I \subseteq \Delta^I \times \Delta^I, \text{ for } r \in N_R$$

---

9 For details on fuzzy description logic see [18].
(Roles remain crisp; however, users may interpret these data in a fuzzy way. We assume that fuzziness should not be attached to the data from the very beginning).

The extension of the \( T \)-interpretation \( I \) to the composed \( \mathcal{EL}@ \) concepts is given by
\[
(@(C_1, \ldots, C_n)_I(x) = @_I(C_1^I(x), \ldots, C_n^I(x))
\]
and
\[
(\exists r.C)_I(x) = \sup\{C^I(y): (x, y) \in r_I\}
\]

The \( \mathcal{EL}@ \) is a surprisingly expressive language with good mathematical properties. It opens a possibility to define declarative as well as procedural semantics of an answer to a user query formulated by means of a fuzzy concept definition. The discussion on the complexity of particular problems, like satisfiability, the instance problem, the problem of deciding subsumption and the proof of soundness and completeness are, however, out of scope of the present paper. For details, see, e.g., [21].

4 \( TIL \) and \( \mathcal{EL}@ \) combined.

Using the example from the outset we are now going to outline the way of integrating the two systems. We illustrate the work with a typed and / or non-typed language, and the role of basic pre-concepts like a type, domain, concept and role. As stated above, \( TIL \) is a typed system. The basic types serve as the pre-concepts.

4.1 Pre-concepts

a) Basic types

\((TIL)\): The epistemic base is a collection of: \( \mathfrak{O} \) – the set of truth-values \{\text{T}, \text{F}\}, \( \mathfrak{T} \) – the universe of discourse (the set of individuals), \( \mathfrak{T} \) – the set of times (temporal factor) and / or real numbers, \( \mathfrak{W} \) – the set of possible worlds (modal factor)

\((\mathcal{EL}@)\): Basic pre-concepts are \( T \) and \( \Delta^I \), as specified in Chapter 3.

The description logic does not work explicitly with the temporal and modal factor. However, there is a possibility to distinguish between necessary ex definitione (T-boxes) and contingency of attribute values (A-boxes). Moreover, \( \mathcal{EL}@ \) contributes the means for handling user preference structures – the preference factor.

\((TIL)\): The universe of discourse is the (universal) set of individuals.

\( \mathcal{EL}@ \) works with varying domains of interpretation \( \Delta^I \).

b) Functions and relations

\( TIL \) is a functional system: Composed (functional) types are collections of partial functions; \( \alpha \)-sets and \( (\alpha\beta) \)-relations are modelled by their characteristic functions, objects of types \( (\mathfrak{O}_\alpha) \), \( (\mathfrak{O}_\alpha\beta) \), respectively.

\( \mathcal{EL}@ \): Being a variant of description logic, \( \mathcal{EL}@ \) is based on the first-order predicate logic where \( n \)-ary predicates are interpreted as \( n \)-ary relations over the universe. However, in \( \mathcal{EL}@ \) this is true only for \( n = 2 \): binary predicates are crisp roles. In the other aspects \( \mathcal{EL}@ \) is actually functional; it deals with (crisp) \( n \)-ary aggregation functions, and unary predicates (concepts) are interpreted as fuzzy sets by their fuzzy characteristic functions \( \Delta^I \rightarrow \text{T} \).
4.2 Assortment of the individuals in the universe

(TIL) properties

In order to classify individuals into particular sorts, we use properties of individuals. They are intensions, namely functions that depending on the states of affairs (the modal parameter \( \omega \)) and time (the parameter \( \tau \)) yield a population of individuals \((\omega)\) that actually and currently have the property in question.

Example: \( h_1, h_2, h_3 \) / \( \tau \) are individuals with the property \( \text{Hotel} / (\omega)_{\tau \omega} \) of being a hotel. In the database setting these individuals belong to the domain of the attribute “hotel”, or, these individuals may instantiate the entity set \( \text{HOTEL} \). That \( h_1, h_2, h_3 \) are hotels is in TIL represented by the constructions of the respective propositions

\[
\lambda w \lambda t \left[ \omega \right] \text{Hotel} \left[ \omega \right] h_1, \lambda w \lambda t \left[ \omega \right] \text{Hotel} \left[ \omega \right] h_2, \lambda w \lambda t \left[ \omega \right] \text{Hotel} \left[ \omega \right] h_3,
\]

where the property \( \text{Hotel} / (\omega)_{\tau \omega} \) is first intensionally descended \( \left[ \omega \right] \text{Hotel} \left[ \omega \right] h_i \) and then ascribed to an individual: \( \left[ \omega \right] \text{Hotel} \left[ \omega \right] h_i \). Finally, to complete the meaning of ‘\( h_i \) is a hotel’, we have to abstract over the modal and temporal parameter in order to construct a proposition of type \( \omega_{\tau \omega} \) that \( h_i \) is a hotel: \( \lambda w \lambda t \left[ \omega \right] \text{Hotel} \left[ \omega \right] h_i \).

Gloss the construction as an instruction for evaluating the truth-conditions: In any state of affairs of evaluation \( \lambda w \lambda t \) check whether the individual \( \left[ \omega \right] h_i \) currently belongs to the actual population of hotels \( \left[ \omega \right] \text{Hotel} \left[ \omega \right] h_i \).

(EL@) equivalents.

Names of properties correspond to the elements of \( N_C \) and \( N_R \). The above propositions are represented by membership assertions:

\[ \text{Hotel}(h1), \text{Hotel}(h2), \text{Hotel}(h3). \]

The example continued. Let \( A, B, C \) / \( \tau \) are individuals with the property of being an agent. In the database setting these individuals belong to the domain of the attribute “user”, or, these individuals may instantiate the entity set \( \text{AGENT} \). However, in order to be able to represent \( n \)-ary properties of individuals by means of binary ones, we need to identify particular users. Of course, in case of a big and varying set of users it is not in general possible to identify each user, and we often have to consider (a smaller number of) user profiles.

(TIL): That \( A, B, C \) are agents is represented by the constructions of the respective propositions:

\[
\lambda w \lambda t \left[ \omega \right] \text{Agent} \left[ \omega \right] A, \lambda w \lambda t \left[ \omega \right] \text{Agent} \left[ \omega \right] B, \lambda w \lambda t \left[ \omega \right] \text{Agent} \left[ \omega \right] C,
\]

where the property \( \text{Agent} / (\omega)_{\tau \omega} \) is intensionally descended and then ascribed to an individual: \( \left[ \omega \right] \text{Agent} \left[ \omega \right] A_i \). Finally, in order to construct a proposition, we have to abstract over the parameters \( w, \tau \): \( \lambda w \lambda t \left[ \omega \right] \text{Agent} \left[ \omega \right] A_i \).

Gloss: In any state of affairs of evaluation check whether the individual \( A \) currently belongs to the actual population of agents.

(EL@): The above propositions are represented by membership assertions:

\[ \text{Agent}(A), \text{Agent}(B), \text{Agent}(C). \]

(TIL): Parking / (\omega)_{\tau \omega}, the property of an individual of being a parking place. For instance, the proposition that \( p_1, p_2, ..., p_n \) / \( \tau \) are individuals with the property of being a parking place, is constructed by
\[\lambda \omega \lambda t \ [(^0\text{Parking}_{\omega t}^0 p)].\]

\((\mathcal{EL}^{\mathcal{O}})\): These individuals belong to the extension of the concept:

Parking\((p_i)\).

### 4.3 Attributes – criteria

In general, attributes are empirical functions, i.e. intensions of a type \((\alpha \beta)_{\tau_0}\). For instance, ‘the President of (something)’ denotes a (singular) attribute. Dependently on the modal factor \(\omega\) and time \(\tau\) the function in question associates the respective country with the unique individual playing the role of its President. But, for instance, George W. Bush might not have been the President of the USA (the modal dependence), and he has not always been and will soon not be\(^{10}\) the President (the temporal dependence).

\((\mathcal{TIL})\) Price \((\tau_1)_{\tau_0}\); an empirical function associating an individual (of type \(\tau\)) with a \(\tau\)-number (its price); to obtain a price of a hotel \(h_i\), we have to execute an empirical procedure:

\[\lambda \omega \lambda t \ [(^0\text{Price}_{\omega t}^0 h)].\]

\((\mathcal{EL}^{\mathcal{O}})\) the value of the attribute Price can be obtained, e.g., by an SQL query

```
SELECT Price FROM Hotel WHERE Hotel.Name=\(h_i\)
```

or by using a crisp atomic role hotel_price.

\((\mathcal{TIL})\) Distance \((\tau_1 \tau_1)_{\tau_0}\); an empirical function assigning a \(\tau\)-number (the distance) to a pair of individuals, for instance:

\[\lambda \omega \lambda t \ [(^0\text{Distance}_{\omega t}^0 h_i^0 p_i)].\]

\((\mathcal{TIL})\) Dist\(E\) \((\tau_1)_{\tau_0}\); the empirical function assigning to an individual a \(\tau\)-number (its distance to another chosen entity \(E\) – a beach, a hotel, ...).

\((\mathcal{EL}^{\mathcal{O}})\) Database point of view: Assuming we have a schema Distance(Source, Target, Value), this is the value of the attribute Distance.Value. It can be obtained, e.g., by the SQL query

```
SELECT Distance.Value FROM Distance, Hotel WHERE Hotel.Name=\(h_i\) AND Hotel.Address=\(x\) AND Distance.Source=\(x\) AND Distance.Target=E
```

\(DL\) point of view: In DL we meet a problem here, because the relation Distance is of arity 3 and DL is a binary conceptual model. For each individual \(E\) we can consider an atomic role hotel_distance_from_E. (Of course, in practical applications we can combine these approaches).

\((\mathcal{TIL})\) Year \((\tau_1)_{\tau_0}\); an empirical function assigning to an individual a \(\tau\)-number (its year of building).

\((\mathcal{EL}^{\mathcal{O}})\) Database and \(DL\) points of view similar as above

\((\mathcal{TIL})\) Appertain-to \((\omega \theta)_{\tau_0}\); the binary relation between individuals. For example, a parking place \(p_i\) belonging to a hotel \(h_i\):

\[\lambda \omega \lambda t \ [(^0\text{Parking}_{\omega t}^0 p_i] \land [(^0\text{Hotel}_{\omega t}^0 h_i] \land [(^0\text{Appertain-to}_{\omega t}^0 h_i^0 p_i]).\]

\((\mathcal{EL}^{\mathcal{O}})\) the relation between a particular hotel and a parking; a crisp role

\(^{10}\) Written in January 2007
4.4 Evaluation of criteria by combining user preferences

The procedural semantics of TIL makes it possible to easily model the way particular agents can learn by experience. An agent may begin with a small ontology of atomic primitive concepts (trivialisations of entities) and gradually obtain pieces of information on more detailed definitions of the entities. In TIL terminology each composed construction yielding an entity \( E \) is an ontological definition of \( E \). For instance, the agent \( A \) may specify the property of being a Suitable (for-\( A \)) hotel by restricting the property Hotel. To this end the property Suitable hotel is defined by the construction composing the price, distance and year attribute values and yielding the degree greater than 0.5.

\[(TIL): \text{Suitable-for } / ((\tau \iota \tau \tau \tau)_{\omega \iota}) \text{ – an empirical (parameters } \tau, \omega \text{) function that applied to an individual (of type } \iota \text{) and a property (of type } (\omega \iota)_{\omega \iota} \text{) returns a property (of type } (\omega \iota)_{\omega \iota} \text{). For instance, the agent } A \text{ may specify the property of being a suitable hotel for the agent } A \text{ can be defined by:}

\[
\lambda w \lambda t \left[ 0^a \text{Suitable-for}_{wt} 0^a A 0^a \text{Hotel} \right] =
\]
\[
\lambda w \lambda t \lambda x \left[ [0^a \text{Hotel}_{wt} x] \land [0^a \text{Evaluate}_{wt} 0^a A [0^a \text{Price}_{wt} x] [0^a \text{DistE}_{wt} x] [0^a \text{Year}_{wt} x]] \geq 0.5 \right].
\]

By way of further refining, we can again define the atomic concept \( 0^a \text{Evaluate} \). To this end we enrich the ontology by \( 0^a \text{Aggregate} \) and \( 0^a \text{Apt-for} \), which can again be refined. And so on, theoretically ad infinitum.

\( \text{Evaluate} / (\tau \iota \tau \tau \tau)_{\omega \iota} \text{ – an empirical function that applied to an individual } a \text{ and a triple of } \tau \text{-parameters (e.g., price, distance, year) returns a } \tau \text{-number } \in [0,1], \text{ which is the preference degree of a particular hotel for the agent } a. \)

\[
[0^a \text{Evaluate}_{wt} a \text{ par}_1 \text{ par}_2 \text{ par}_3] =
\]
\[
[0^a \text{Aggregate} [0^a \text{Apt-for}_{wt} a \text{ par}_1] [0^a \text{Apt-for}_{wt} a \text{ par}_2] [0^a \text{Apt-for}_{wt} a \text{ par}_3]].
\]

\( \text{Aggregate} / (\tau \tau \tau) \text{ – the aggregation function that applied to the triple of } \tau \text{-numbers returns a } \tau \text{-number = the degree of appropriateness.} \)

\( \text{Apt-for} / (\tau \iota \tau \tau \tau) \text{ – an empirical function that applied to an individual } a / \iota \text{ and a } \tau \text{-parameter } \text{par}_i \text{ (e.g., price, distance, and so like) returns a preference scale of the respective parameter } \text{par}_i \text{ for the user } a. \text{ The scale is a } \tau \text{-number } \in [0,1]. \)

For instance,
\[
[0^a \text{Evaluate}_{wt} 0^a A [0^a \text{Price}_{wt} x] [0^a \text{DistE}_{wt} x] [0^a \text{Year}_{wt} x]] = [0^a \text{Aggregate}
\]
\[
[0^a \text{Apt-for}_{wt} A [0^a \text{Price}_{wt} x] [0^a \text{Apt-for}_{wt} A [0^a \text{DistE}_{wt} x] [0^a \text{Apt-for}_{wt} A [0^a \text{Year}_{wt} x]]].
\]

The empirical function \( \text{Evaluate} \) is the key function here. Applied to an individual agent (user) and particular criteria it returns the agent’s preference-degree of a particular object. Each agent may dynamically (parameter \( \tau \)) choose (parameter \( \omega \)) its own function \( \text{Evaluate} \). The algorithm computing the preference-degree of an object consists of two independent sub-procedures:

i) user preference scale \( \text{Apt-for}_{wt} \) of the \( TIL \)-type (\( \tau \tau \)), or using the \( \mathcal{EL}^\oplus\) notation: \( \langle \text{user, par}_i \rangle \rightarrow [0,1], \text{ where } \text{par}_i \text{ is the value of a particular criterion (for instance price, distance, etc.).} \)

Here the additional role of \( \mathcal{EL}^\oplus \) comes into play. The \( \mathcal{EL}^\oplus \) logic makes it possible to choose an appropriate scale algorithm. It can be a specific function for a particular user \( U_1 \), e.g.:
\[
\text{fclose}_U(x) = \max\left(0, 1 - \frac{x}{1000}\right), \quad \text{fcheap}_U(x) = \max\left(0, 1 - \frac{x}{2000}\right)
\]

and

\[
\text{fnew}_U(x) = \max\left(0, 1 - \frac{2005 - x}{20}\right)
\]

\[\text{ii)}\quad \text{the aggregation function } \text{Aggregate} \text{ of the } \mathcal{TIL} - \text{type } (\tau \tau \tau), \text{ or in the } \mathcal{EL}^\oplus \text{ notation (understood as a many valued connective) } @: [0,1]^3 \rightarrow [0,1], \text{ computing the global preference degree.}
\]

Here we consider the \text{Aggregate} function as not being user-dependent, but rather \text{system}-dependent (therefore, in \mathcal{TIL} –notation there is no \(\tau\omega\)-parameter). In other words, it is a system algorithm of computing the general user preference. Of course, we might let each user specify his/her/its own algorithm but in practice it suffices to consider different user profiles associated with each aggregation function. Thus the system may test several algorithms of aggregation, e.g., those that were used for users with a similar profile, in order to choose the suitable aggregation. It does not seem to be necessary to further refine the specification in \mathcal{TIL}. Instead we either call at this point a software module, or make use of the \mathcal{EL}^\oplus \text{ logic. In the example above we used the weighted average:}

\[
@_{U_1}(\text{close}, \text{cheap}, \text{new}) = \frac{2 \times \text{close} + 3 \times \text{cheap} + \text{new}}{6}
\]

### 4.5 Communication of agents; messages

The communication aspects are not elaborated in \(\mathcal{EL}^\oplus\) from the semantic point of view. Hence it represents the added value of \mathcal{TIL} when integrating with \(\mathcal{EL}^\oplus\). However, in SQL we have ORDER BY command and when dealing with preferences we work with the notion of the best, top-k, respectively, answers. The \(\mathcal{EL}^\oplus\) many valued logic setting understood as a comparative tool (numerical values do not matter) is an appropriate tool for evaluating fuzzy predicates. It provides a good semantics for ordering preferences of answers (see [13]).

The \mathcal{TIL}-philosophy is driven by the fact that natural language is a perfect logical language. Hence the \mathcal{TIL}-specification is close to an ordinary human reasoning and natural-language communication. On the other hand, however, the high expressive power of the \mathcal{TIL} language of constructions may sometimes be an obstacle to an effective implementation. This problem is dealt with by the step-by-step refinement as discussed above. At the first step we specify just a coarse-grained logical form of a message; the execution is left to particular Java modules. Then a more fine-grained specification makes it possible to increase agent’s “intelligence” by letting him dynamically decide which finer software modules should be called. To this end we combine Java modules, Prolog, fuzzy Prolog Ciao, etc.

\((\mathcal{TIL})\): The general scheme of a message is:

\[
\text{Message} / (\text{o} \text{t} \text{t} \text{o}_\omega)_{\text{to}} \\
\lambda w\lambda t [^0\text{Message}_w] [^0\text{Who}_\text{Whom}] [^0\text{Type}_w] [^0\text{What}],
\]

where
Who /ι, Whom /ι, What (content)/ατο, Type / (οατο)το.

What – the subject of the message is a specification of an intension (usually a proposition of type ατο).

(EL): The description logic does not incorporate a specific semantic logical description of messages. It is usually handled by an implementation component (generally in the Software Engineering part) by dealing with exceptions, deadlocks, etc.

(TIL): There are three basic types of messages that concern propositions; i.e., these types are properties of propositions, namely Order, Inform, Query/(οοτο)το. In an ordinary communication act we implicitly use the type Inform affirming that the respective proposition is true. But using an interrogative sentence we ask whether the proposition is true (Query), or using an imperative sentence we wish that the proposition were true (Order).

The content of a message is then the construction of a proposition, the scheme of which is given by:

\[ \lambda w \lambda t \ [0^{Type\_nt}]^{0\What} \rightarrow \alpha_{\tau o}. \]

In what follows we specify in more details possible typical types of messages.

\[ Type = \{ Seek, Query(Yes-No), Answer, Order, Inform, Unrecognised, Refine, \ldots \}; \]

where \( Type_i / (\alpha\alpha_{\tau o})_{\tau o} \) or \( Type_i / (\alpha*\tau_o)_{\tau o} \).

Examples of a content of a message:

\[ [0^{Seek\_nt}]^{0\What}; \] What / \( \alpha_{\tau o} \rightarrow \) send me an answer = the actual \( \alpha \)-value of What in a given state of affairs \( w,t \) of evaluation.

\[ [0^{Query\_nt}]^{0\What}; \] What / \( \alpha_{\tau o} \rightarrow \) send me an answer = the actual \( \alpha \)-truth-value of What in a given state of affairs \( w,t \).

\[ [0^{Order\_nt}]^{0\What}; \] What / \( \alpha_{\tau o} \rightarrow \) manage What to be actually True (in a state of affairs \( w,t \)).

\[ [0^{Inform\_nt}]^{0\What}; \] What / \( \alpha_{\tau o} \rightarrow \) informing that What is actually True

\[ [0^{Answer\_nt}]^{0\What} = a / \alpha]; \] where \( a = [0^{What\_nt}] \); the answer to a preceding query or seek.

\[ [0^{Unrecognised\_nt}]^{00\What};\] the atomic concept \( ^0\What \) has not been recognised; a request for refinement.

Note that Unrecognised is of type \( (\alpha*\tau_o)^{\tau_o} \), the property of a construction (usually an atomic concept). Therefore the content of the message is not the intension What constructed by \( ^0\What \), but the construction \( ^0\What \) itself. The latter is mentioned here by trivialisation, therefore \( ^{00}\What \).

\[ [0^{Refine\_nt}]^{00\What} = C \rightarrow \alpha_{\tau o} ]; \] an answer to the message on unrecognised atomic concept. The construction \( C \) is the respective composed specification (definition) of What, i.e., C and \( ^0\What \) are equivalent, they construct the same entity: \( C = ^0\What \).

For instance, the set of prime numbers can be defined as the set of numbers with two factors: [[0\Refine\_nt]^{00Prime} = 0[λx [0\Card\_nt] λy [0\Div\_nt x y] = 02]], where x, y \( \rightarrow \Nat \) (the type of natural numbers), Div / (οNat Nat) – the relation of being divisible by, Card / (Nat (οNat))– the number of elements of a set.
4.6 Example of communication

Now we continue the simple example from the outset. We will analyse a part of the dialog of the three agents $A$, $B$, $C$. Sentences will be first written in ordinary English then analysed using TIL, transformed into the standardised message, and if needed provided by a gloss. For the sake of simplicity we will omit the specification of TIL-types of particular objects contained in a message. However, since the TIL-type is an inseparable part of the respective TIL-construction, we do not omit it in a real communication of agents. For instance, when building an agent’s ontology, each concept is inserted with its typing.

**Message 1 (A to B):** ‘I wish $B$ to seek a suitable hotel for me.’

$A(\text{TIL})$: \[ \lambda w \lambda t \left[ \begin{array}{c} \text{Wish}_{wt}^0 A \ 	ext{Seek}_{wt}^0 B \ 	ext{Suitable-for}_{wt}^0 A \ 	ext{Hotel}^0 \end{array} \right] \]

$A(\text{TIL}\ m1)$: \[ \lambda w \lambda t \left[ \begin{array}{c} \text{Message}_{wt}^0 A \ 	ext{Seek}_{wt}^0 \text{Suitable-for}_{wt}^0 A \ 	ext{Hotel}^0 \end{array} \right] \]

**Gloss:** The agent $A$ is sending a message to $B$ asking to seek a suitable hotel for $A$.

**Message 2 (B to A):**

However, the agent $B$ does not understand the sub-instruction $^0\text{Suitable-for}_A^0\text{Hotel}$, because he does not have the atomic concept $^0\text{Suitable-for}$ in his ontology. Therefore, he replies a message to $A$, asking to explain:

‘I did not recognise $^0\text{Suitable-for}$.’

$B(\text{TIL}\ m2)$: \[ \lambda w \lambda t \left[ \begin{array}{c} \text{Message}_{wt}^0 B \ 	ext{Unrecognised}_{wt}^0 \ 	ext{Suitable-for}_{wt}^0 \end{array} \right] \]

**Remark** Thus the lower index $^n$ can be understood as an instruction to execute an empirical inquiry (search) in order to obtain the actual current value of an intension, here the property of being a suitable hotel (for instance by searching agent’s database or by asking the other agents, or even by means of agent’s sense perception).

The upper index $^0$ serves as a marker of the primitive (atomic) concept belonging the agent’s ontology. If it does not, i.e., if the agent does not know the concept, he has to ask the others in order to learn by experience.

**Message 3 (A to B):**

The agent $A$ replies by specifying the restriction of the property $\text{Hotel}$ to those hotels which are evaluated with respect to price, distance and the year of building with the degree higher than 0.5:

$A(\text{TIL})$ \[ ^0\text{Suitable-for} / (\begin{array}{c} \text{Price}_{wt}^0 \ 	ext{Distance}_{wt}^0 \ 	ext{Year}_{wt}^0 \end{array}) \rightarrow \begin{array}{c} \text{Suitable-for}_{wt}^0 A \ 	ext{Hotel}^0 \end{array} \]

$A(\text{TIL}\ m3)$: \[ \lambda w \lambda t \left[ \begin{array}{c} \text{Message}_{wt}^0 A \ 	ext{Refine}_{wt}^0 \ 	ext{Suitable-for}_{wt}^0 A \ 	ext{Hotel}^0 \end{array} \right] \]

**Gloss:** The $A$’s answer message should refine the atomic concept $^0\text{Suitable-for}$. Now there is a problem, however. The agent $B$ would have to remember the respective message asking for the refinement in order to apply the property to proper arguments (namely $A$ and $\text{Hotel}$). This would not be plausible in practice, because $A$ is the aid prayer, not $B$. Therefore the answer message contains the smallest constituent containing the refined concept:

$A(\text{TIL}\ m3)$: \[ \lambda w \lambda t \left[ \begin{array}{c} \text{Message}_{wt}^0 A \ 	ext{Refine}_{wt}^0 \ 	ext{Suitable-for}_{wt}^0 A \ 	ext{Hotel}^0 \end{array} \right] = \lambda w \lambda t \left[ \begin{array}{c} \text{Hotel}_{wt}^0 x \ 	ext{Evaluate}_{wt}^0 A \ 	ext{Price}_{wt}^0 x \ 	ext{Distance}_{wt}^0 x \ 	ext{Year}_{wt}^0 x \end{array} \right] \]
In this way the agent $B$ obtains a piece of knowledge what should he look for. Another possibility would be $A$’s sending the original message 1 refined, i.e., the constituent $[^0\text{Suitable-for}_w A]^0\text{Hotel}$ replaced by the new specification:

$$A(\text{TIL} m^3): \lambda w. t \; [^0\text{Message}_w A \; ^0B \; \lambda w. t \; [^0\text{Seek}_w A \; ^0\text{Hotel}_w x] \land \left[ [^0\text{Evaluate}_w A \; ^0\text{Price}_w x] \land [^0\text{Dist}_w E x] \land [^0\text{Year}_w x] \geq 0.5 \right]]$$

However, we prefer the former, because in this way $B$ learned what a suitable hotel for $A$ means. Or rather, he would learn if he understood $^0\text{Evaluate}_w A$, which may not be the case if he received the request for the first time. Thus if $B$ does not have the concept in his /her ontology, he again sends a message asking for explaining:

**Message 4 (B to A):**

$B$: I did not recognise $^0\text{Evaluate}_w A$.

$$B(\text{TIL} m^4): \lambda w. t \; [^0\text{Message}_w A \; ^0B \; \lambda w. t \; [^0\text{Unrecognised}_w A \; ^0\text{Evaluate}_w A]]$$

**Message 5 (A to B):**

$$A(\text{TIL} m^5): \lambda w. t \; [^0\text{Message}_w A \; ^0B \; \lambda w. t \; [^0\text{Refine}_w A \; ^0\text{Price}_w x] \land [^0\text{Dist}_w E x] \land [^0\text{Year}_w x]] = \lambda w. t \; [^0\text{Aggregate}_w A \; ^0\text{Price}_w x] \land [^0\text{Dist}_w E x] \land [^0\text{Year}_w x]]]$$

And so on, the refinement may continue and the agents may learn new concepts (from the theoretical point of view ad infinitum).

Anyway, finally $B$ fully understands the message and attempts at fulfilling the task; recall that he is to seek a suitable hotel for $A$.

Note that the whole process is dynamic, even agents’ learning by the process of refining particular atomic concepts. $B$ knows now that actually and currently a hotel suitable for $A$ is such a hotel the price, distance from the beach and the year of building of which evaluate with respect to $A$’s scaling $[^0\text{Suitable-for}_w A]_w$ with the degree higher than 0.5. But he also knows that it might have been otherwise (the modal parameter $w$ / $\omega$) and it will not have to be always so (the temporal parameter $t$ / $\tau$). In other words, $A$ and $B$ now share common knowledge of the composed concept defining the property of being a suitable hotel for $A$.

When eventually $B$ accomplishes his search he sends an answer to $A$:

**Message 6 (B to A):**

$$A(\text{TIL} m^6): \lambda w. t \; [^0\text{Message}_w A \; ^0B \; \lambda w. t \; [^0\text{Answer}_w A \; ^0\text{Suitable-for}_w A \; ^0\text{Hotel}]]$$

$$= \{ (h1, 0.7), (h5, 0.53) \}$$

**Gloss:** $B$ found out that there are two instances of the property $\nu$-constructed by the construction $[^0\text{Suitable-for}_w A \; ^0\text{Hotel}]$, namely the hotel $h1$ that has been evaluated with the degree 0.7 and $h5$ with the degree 0.53.

Since $h1$ has been evaluated as better than $h5$, $A$ chooses the former.

At this point the communication can continue as a dialogue between $A$ and $C$ in a similar way as above. The aim is now finding a suitable parking close to the chosen hotel $h1$ and then asking to navigate to the chosen parking place:

$$\lambda w. t \; [^0\text{Message}_w A \; ^0C \; \lambda w. t \; [^0\text{Seek}_w A \; ^0\text{Suitable}_w A \; ^0\text{Parking}]]$$

---

11 Here we use the classical set-theoretic notation without trivialisation, for the sake of simplicity.
\[ \lambda w \lambda t \left[ ^0 \text{Message} \right] \theta C \theta A \lambda w \lambda t \left[ ^0 \text{Unrecognised} \right] ^0 \text{Suitable} \]

\[ \lambda w \lambda t \left[ ^0 \text{Message} \right] \theta C \theta A \lambda w \lambda t \left[ ^0 \text{Refine} \right] ^0 \text{Suitable} \theta A ^0 \text{Parking} = \]

\[ ^0 \left[ \lambda w \lambda t \lambda x \left[ ^0 \text{Parking}_w x \right] \land \left[ ^0 \text{Evaluate}_w \theta A ^0 \text{Price}_w x \left[ ^0 \text{DistE}_w x \right] \geq ^0 0.5 \right] \right] \]

\[ \lambda w \lambda t \left[ ^0 \text{Message} \right] ^0 \theta C ^0 \theta A \lambda w \lambda t \left[ ^0 \text{Answer} \right] ^0 \text{Suitable} \theta A ^0 \text{Parking} = \left\{ \{ p2, 0.93 \}, \{ p1, 0.53 \} \right\} \]

The message closing the dialogue might be sent from A to C:

\[ \lambda w \lambda t \left[ ^0 \text{Message} \right] ^0 \theta A ^0 \theta C \lambda w \lambda t \left[ ^0 \text{Order} \right] \lambda w \lambda t \left[ ^0 \text{Navigate-to} \right] ^0 p2 \].

At this point the agent C must have ^0 \text{Navigate-to} in his/her ontology (if he/she does not then the learning process described above begins); C thus knows that he/she has to call another agent D which is a GIS-agent that provides navigation facilities (see [6]).

Concluding this paragraph we again compare the \text{TIL} approach with \text{EL}@. An analogy to the above described means of communication can be found in the DL community. There are heuristics for the top-k search (see [13]). However, these facilities lack any formal / logic / semantic specification. The development of description logic and its variants can be considered as a step forward to the development of languages which extend W3C standards. In [9] a step in this direction is described. In particular the \text{EL}@ variant of the description logic can be embedded into classical two-valued description logic with concrete domains (see [1]), and thus also into OWL (or a slight extension of it). Using the results described in this paper, especially the added value of \text{TIL}, we can expect the extension of W3C based specification of web service languages using the OWL representation.

5. \textbf{Conclusion: A hybrid system}

In the previous chapters, especially by using the parallel description of our motivating example in Chapter 4, we tried to show that \text{TIL} and \text{EL}@ have many features in common. Both the systems can share some basic types, functions, concepts and roles; both the systems distinguish extensional and intensional context (the former being modelled by the intensional descent in \text{TIL} and \text{A-Boxes in DL}, the latter illustrated here by the (user-) definition or specification of a multi-criterion search). These features can form the intersection \text{TIE}@. On the other hand, both the systems can be enhanced by accommodating features of the other system, thus forming a union \text{TIL+E}@. The main contribution of \text{EL}@ is the method of modelling multi-criterion aspects of user preferences (some heuristics have been tested in separate works), and computing global user preferences by means of the aggregation functions and scaling. \text{TIL} contributes to this union the method of a very fine-grained and rigorous knowledge specification closed to natural language, including procedural hyper-intensional semantics. We are convinced that these aspects are crucial for a smooth communication and reasoning of agents in the multi-agent world. Artificial Intelligence is sometimes characterised as a ‘struggle for consistency’. To put it slightly metaphorically, reality is consistent. Only our ‘making it explicit’ in language may lead to paradoxes and inconsistencies due to misinterpretations that are caused by a too coarse-grained analysis of assumptions.

The specification of the formal model of the hybrid system is however still a subject of further research. Currently we plan to perform experiments and tests on real data using the hints described in Chapter 4.
In the team led by M. Duží, working on the project “Logic and Artificial Intelligence for multi-agent systems” (see http://labis.vsb.cz/), we pursue research on multi-agent systems based on TIL. Currently we implemented software modules simulating the behaviour of mobile agents in a traffic system. The agents can choose particular realisations of their pre-defined processes; moreover, they are able to dynamically adjust their behaviour dependently on changing states of affairs in the environment. They communicate by message-exchange system. To this end the TIL-Script language (see [14]) has been designed and it is currently being implemented. We also plan to test some modules with \(\mathcal{E}\mathcal{L}\) features.

The project in which P. Vojtaš is involved (see [17]) deals with theoretical models compatible with W3C standards and experimental testing of multi-criterion search dependent on user preferences. We believe that the TIL features will enhance the system with a rigorous semantic description and specification of the software / implementation parts.

When pursuing the research we soon came to the conclusion that the area of the semantic web and multi-agent world in general is so broad that it is almost impossible to create a universal development method. Instead we decided to develop a methodology comprising and integrating particular existing and/or newly developed methods as well as our fine-grained rigorous logic. The paper is an introductory study aiming at a more universal logical approach to the ‘multi-agent world’, which at the same time opens new research problems and trends. The main challenges are formal measures (soundness and completeness) and implementation measures of the integrated hybrid system.

---

**ACKNOWLEDGEMENTS**

This work has been supported by the project No. 1ET101940420 “Logic and Artificial Intelligence for multi-agent systems” within the program “Information Society” of the Czech Academy of Sciences, and by the “Semantic Web” project No. 1ET100300419 of the Czech IT agency.

**REFERENCES**


The TIL-Script language description is available at the VSB-TIL homepage: http://www.cs.vsb.cz/TIL.


‘Semantic Web’ project of the Czech IT agency 1ET100300419.


