The Learning and Question Answering Modes in the Dolphin System for the Transparent Intensional Logic

Andrej Gardoň and Aleš Horák

Faculty of Informatics, Masaryk University Brno
Botanická 68a, 602 00 Brno, Czech Republic
`xgardon@fi.muni.cz`, `hales@fi.muni.cz`
`http://nlp.fi.muni.cz/projects/dolphin`

Abstract. In this paper, we present the two modes of operation of the developed Dolphin system, which implements a knowledge base for the Transparent Intensional Logic formalism.
We offer several examples of the system input with detailed explanation of how these inputs are parsed and stored in the system and how the system reuses the knowledge base information for answering simple questions offering a kind of simple inference.

Key words: Dolphin, transparent intensional logic, TIL, knowledge base

1 Introduction

The transparent intensional logic (TIL) is a higher order typed logical system designed for representing meaning of human language in a computer. In comparison with traditional logic systems, TIL is able to distinguish so called intensions, i.e. entities dependent on the reference possible world and time moment, and extensions. Using the higher order types, TIL allows us to express *attitudes*[1] to other natural languages objects in a declarative manner. For more detailed information about TIL you can see [1,2].

In the following text, we will present details of an ongoing project called Dolphin, which is an implementation of an efficient knowledge base built over the TIL formalism. The basic ideas of Dolphin have been published in [3].

2 The TIL Knowledge Base Architecture

The Dolphin system is designed to process the output of the syntactic parser `synt` [2,4], which is able to produce syntactic trees as well as logical representations of input sentences in the form of TIL constructions. In Dolphin, the constructions are parsed and stored in the knowledge base (KB), which takes

[1] e.g. *Tom believes that Peter says that the earth is flat.*

the form of a semantic network. The system is then able to process the facts from KB in the way allowing to answer *yes/no* questions and questions about firmly established objects relations in the form of *The apple is red. The cube is red. What red objects do you know?*. Although current version of the system does not support all TIL capabilities in full breadth, we hope the design of it will make it easy to add them in the future.

2.1 How the Dolphin Database Works

The Basic Dolphin idea is a separation of the *language layer* from the *logical layer*. Human words are just a way how to describe some objects. There are many languages but all of them work over the same set of objects. Therefore Dolphin works with this set and is ready to understand everything what his teacher teaches him. This process lies in transforming teacher words to the KB objects mentioned above. Here `synt` plays its role as it produces TIL transcription of sentence. As a young child who gets in touch with word *apple* for the first time, Dolphin after obtaining *apple* on its input takes its TIL description and creates new object in its object layer – let it be object #1. Next time we mention this particular *apple*, Dolphin realizes object #1 in its KB and whatever is told about it is appended to this object.

2.2 The Role of the Language Layer

The linkage from the apple and the object #1 is driven by the language layer. The word *apple* is stored in the appropriate language file and is connected to the object #1. If the system is switched to the *question answering mode* (see the Section 2.3), whenever the word *apple* appears, it is replaced by object #1 for the purpose of the logical inference. The current implementation of the language layer supports multiple languages but the rest of the system works, at the moment, with one selected language. The reason for this is that the language learning needs to employ specific inference. The language layer also offers techniques for synonyms and homonyms but again complex inference is needed to use it. The implementation of the language layer uses an adapted library of special B-trees and thus provides effective transcription of a word to the equivalent object.

2.3 The Input Sentence Processing

Let us take an example input sentence (in Czech, as this is the current input language of `synt`):

Jablko je červené. (An apple is red)

and its corresponding TIL transcription:

$$\lambda w_1 \lambda t_2 (\exists i_3)([{}^0\text{jablko-0}_{w_1 t_2}, i_3] \ \wedge \ [{}^0\text{červený-2}_{w_1 t_2}, i_3]) \ldots \pi$$

At first, variables are identified and single applications are isolated. Variables that are not covered by λ-abstraction are replaced by new constants named after the variables. The λ-abstracted variables are currently handled in a simplified way – when the λ-abstraction is used without quantification, the abstracted variables usually go over the ω (possible worlds) or τ (time moments) types. In case of an ω-variable, the Dolphin's world $w_{Dolphin}$ is assigned to it. All time variables (of type τ) are, in the current version, replaced by an object representing a "general time independent object," so as we can see the current version does not support time processing at all. This feature will be the main goal of our future work. Other quantified variables are initialized differently according to the system running mode. There are two running modes currently available – the *learning mode* and the *question answering mode.*

The Learning Mode processes and stores new facts and it is activated by a full stop mark at the end of a sentence. In this mode, each uninitialized variable is replaced by a new object named after the variable. If we want to add new facts about an object previously mentioned in the conversation with Dolphin, we have to stress this with the demonstrative pronoun (*ten*, *ta*, *to* – in Czech this corresponds to the definite article in English). The object is then marked with an exclamation mark (!) and is looked up in the knowledge base and the variable is replaced by the stored object. For example if we want to add the fact that

To jablko je červené. (The apple is red.)

the *apple* is analysed as $^{0}\mathsf{!jablko\text{-}0}/(oi)_{\tau\omega}$ and then found as the previously stored object #23 and the construction would thus contain object $^{0}\#23$ instead of the i_3 variable displayed above. Currently, the system supports only the existential quantification since the universal quantification defines new inference rules and the complex inference has not been implemented so far.

We may mentioned the way, how possible fact conflicts are handled – if the input fact assigns something to an existing KB object and the fact is in conflict with the KB content then an error message is raised. For demonstration let us have a sentence

The apple is red.

stored in KB. Now we would like to store the sentence

The apple is not red.

This raises an error message and the second sentence is not stored. Today there is no tool for saying to Dolphin:

A fact in KB is wrong, I have the correct one.

so whatever is stored in KB will remain unchanged until a reset of the whole database.

The Question Answering Mode works similarly to the learning mode but there is no unification of free variables with new objects. Instead, the first application containing an uninitialized variable suggests a way how to unify this variable with a particular value. If we take our example sentence as a question

Je nějaké jablko červené? (Is there any red apple?)

first application containing free variable i_3 (variables w_1 and t_2 are initialized as described above) will be

$$[{}^0\mathsf{jablko\text{-}0}_{w_1 t_2}, i_3]$$

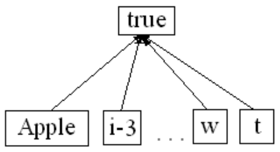
Thanks to this application, i_3 is unified with an object that is stored in Dolphin and is an Apple (is linked to the class construction of apples). The following application (saying that the object is *red*) posts a second requirement to check. If the object unified with i_3 is not Red (there is no relation between the unified object and the class of red objects), the system returns to the state when i_3 was initialized and tries another possibility. If all possible ways were checked and there is no object that is Apple and Red concurrently the answer *NO* is returned. In case all the requirements are fulfilled, the answer *YES* is returned with the selected i_3 value. Again there is no universal quantification feature yet, but simple questions such as

Jaké červené objekty znáš? (What red objects do you know?)

can be processed correctly.

2.4 How the Objects are Stored

Objects are essential elements of logical analysis and the TIL construction processing. In the Dolphin knowledge base, all data are objects related with each other. It does not matter if we store an individual or a function in KB, we are still working with one object. For example, the class of apples (Apple) is represented as the characteristic function of the class, i.e. as a (world and time dependent) function which returns an o-object (boolean *true/false*) for each ι-object (an individual) given as its argument. Each such ι-object is stored separately from Apple, of course.



There is an information that Apple and the ι-object i_3 are in relation and this information is shared among all participants. Each object is represented as a separate file. This can be considered as very ineffective while each time we need an object we have to read a separate file from disc. In fact, there is no solution that does not use disc reading as there is so many objects and absolutely no chance to store them all in operating memory. On the other hand simple file for each object offers some benefits. In the future, a specialized file system technology can be developed or archiving methods can be incorporated in the storage process without the database functionality limitation or reimplementation.

3 A Complex Example

Let us take three example sentences:

1. *Toto je jablko. (This is an apple.)*
2. *Tato kostka je červená. (This cube is red.)*
3. *To jablko je červené. (The apple is red.)*

and their TIL transcriptions provided as inputs to Dolphin, step by step:

$$\lambda w_1 \lambda t_2([{}^0\textsf{jablko-0}_{w_1 t_2}, Toto]) \ldots \pi$$

The word *toto* (this) makes the resulting construction an open construction (with the *free* variable *Toto*) which needs to receive so called *pragmatic anchor*.[2] This input creates a new object in the knowledge base and Dolphin prints the assigned object number:

```
> Toto je jablko. (This is an apple.)
> stored as object 6.
```

The second sentence

```
> Tato kostka je červená. (This cube is red.)
```

with the corresponding TIL construction

$$\lambda w_1 \lambda t_2([{}^0\textsf{!kostka-0}_{w_1 t_2}, i_3] \ \wedge \ [{}^0\textsf{červený-2}_{w_1 t_2}, i_3]) \ldots \pi$$

where the variable w_1 is unified with object #2 (representing the Dolphin's world), the variable t_2 is unified with object #3 (representing the General time object) and the variable i_3 is now initialized with a new object of type ι (let it be #7), since no previously mentioned object from the class *kostka* was found.

Each trivialization asks the language layer whether it knows the word. If the word is not found, it is stored and a new object is created with connection to this word.

So $\wedge$ (AND) in our transcription causes that search in language layer is performed and object #10 is returned (we suppose that AND was previously stored).

Applications are represented as relations among objects that are participating in the application. Thus in Dolphin, the partial application

$$[{}^0\textsf{červený-2}/(oi)_{\tau\omega} w_1]$$

[2] see [2, the Section 5.2.4] or [5, the Section 7.1]

is represented as a relation between *červený* and the object of the variable w_1. The information about the relation is stored both in object *červený* and w_1 object and of course in the final object – the result of this application is a new $(o\iota)\tau$-object (a *chronology* of the class of objects which are červený/red) which is than applied on the general time object. The final object of the application

$$[{}^0\mathsf{červený\text{-}2}_{w_1 t_2}, i_3] \ldots o$$

is then created as an o-object (a truth value) and in the learning mode the object receives the *true* value.

The third example sentence is stored similarly as the second one with the i_3 variable definition difference.

```
> To jablko je červené. (The apple is red.)
```

TIL transcription

$$\lambda w_1 \lambda t_2([{}^0\mathsf{!jablko\text{-}0}_{w_1 t_2}, {}^0\mathsf{\#6}] \;\wedge\; [{}^0\mathsf{červený\text{-}2}_{w_1 t_2}, {}^0\mathsf{\#6}]) \ldots \pi$$

Semantic network for three sentences is displayed in the Figure 1. Note that there are many objects with *true* value in the network. This is because each application leads to a unique object and if this object is not yet in KB, it is created. Thanks to this, the λ-abstraction can be done in a straightforward way. It is worth saying that negation is a special function over the o-object that does not create a new object but it replaces the existing *true* value with *false*. If we ask the question

Je to jabko červené? (Is the apple red?)

TIL transcription is in form of a match:

$$x \ldots o : ([{}^0\mathsf{!jablko\text{-}0}_{w_{Dolphin} t_{now}}, {}^0\mathsf{\#6}] \;\wedge\; [{}^0\mathsf{červený\text{-}2}_{w_{Dolphin} t_{now}}, {}^0\mathsf{\#6}]) \ldots o$$

Basically it asks whether the previously mentioned object #6 is in relation with Apple and Red concurrently. It is easy to answer as we have this information in our knowledge base already. System just obtains the result of the applications on the right side of the match (common applications of objects) and checks whether final object of the right side is *true.* In this case, the answer is *yes* but what happens if we have the question

Je to jablko zelené? (Is the apple green?)

Since we are in the question answering mode, the basic trivialization of *green* will fail as we do not have such object in KB yet. The system does not follow the predicate logic *closed world assumption*, thus the answer is “I DO NOT KNOW” instead of “NO.” The question

Které červené objekty znáš? (What red objects do you know?)

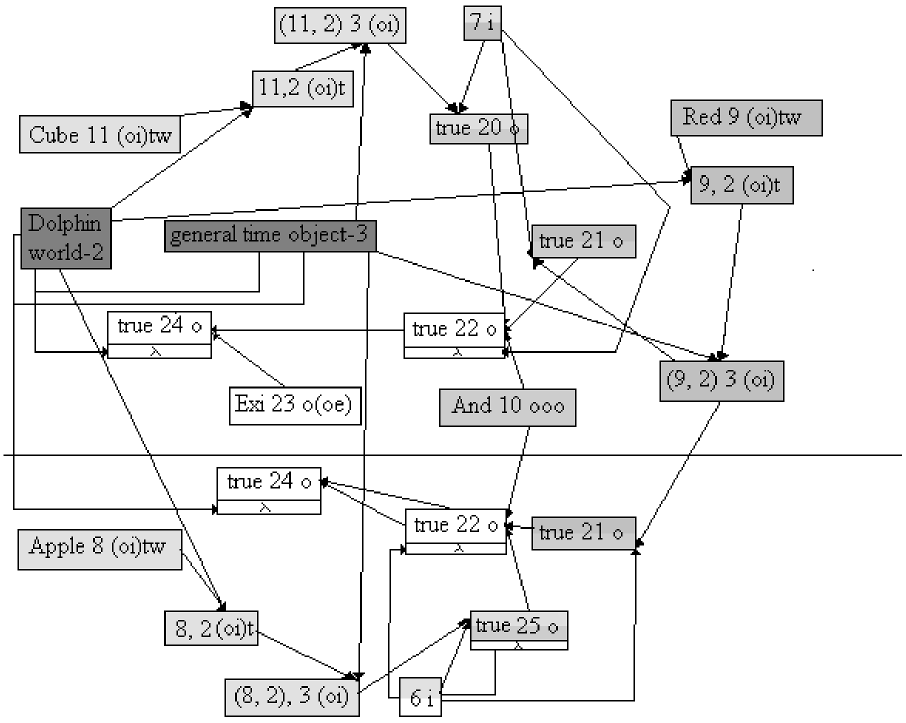


Fig. 1. The semantic network for the 3 example sentences.

is again analysed as a match

$$s \ldots (oi) : {}^0\text{červený-1}_{w_{Dolphin} t_{now}} \ldots (oi)$$

Here the system evaluates all possible objects which are stored in KB as related to the class *červený* (applied on the Dolphin's world and the general time object) obtaining the class of all red objects known to Dolphin. Thanks to system design, this operation consists in fast searching through the semantic network. As can be seen in the Figure 1, the Red object applied on the Dolphin's world and the General time object ((oi)-object with `(9,2) 3` label) has connections to the *true* value. Now (simplified) it is enough to look what object is at the end of the connection that runs out from the place of the *true* value where the connection from `(9, 2) 3` object ended. In this way we find out that objects #6 and #7 are red.

4 Conclusions and Future Work

We have presented the current state of development of the Dolphin knowledge base for the Transparent Intensional Logic. The system is currently able to store TIL constructions in an efficient semantic network structure with offers basic question answering ability.

The Dolphin system is able to parse and store the `synt` output in the form of TIL constructions, to check the consistence of the database and to answer simple questions. The final aim of the Dolphin development is the full support of working with possible worlds and time moments and the provision of a complex inference tool. The directly following version will be directed to the time span support. The plan is that every mentioned time moment will be represented as a separate object, which will contain contain information about objects that are in relation with it. Through this system a simple fact will have different truth value in various time moments and it will be possible to answer questions like:

What is true in the time moment XXX?

References

1. Tichý, P.: The Foundations of Frege's Logic. de Gruyter, Berlin, New York (1988)
2. Horák, A.: The Normal Translation Algorithm in Transparent Intensional Logic for Czech. PhD thesis, Masaryk University, Brno (2002)
3. Gardoň, A., Horák, A.: Dolphin – a Knowledge Base for Transparent Intensional Logic. In: Proceedings of the Seventh International Workshop on Computational Semantics (IWCS-7), Tilburg, The Netherlands (2007) 316–319
4. Horák, A., Kadlec, V.: New Meta-grammar Constructs in Czech Language Parser synt. In: Proceedings of Text, Speech and Dialogue 2005, Karlovy Vary, Czech Republic, Springer-Verlag (2005) 85–92
5. Materna, P.: Concepts and Objects. Volume 63 of Acta Philosophica Fennica. The Philosophical Society of Finland, Helsinki (1998)