Knowledge Base for Transparent Intensional Logic and Its Use in Automated Daily News Retrieval and Answering Machine

A. Gardoň and A. Horák

***Abstract*—This paper describes the design of a knowledge representation and reasoning system, named Dolphin, which is based on higher-order temporal Transparent Intensional Logic (TIL). An intelligent agent (NAM), that is able to read newspaper headlines from specialized internet server and allows users to ask questions about various world situations is chosen to demonstrate Dolphin features. Temporal aspects play an essential role in natural language therefore we present how this phenomenon is handled in the system. Reasoning capabilities of the agent are divided into three individual strategies and described in the text. As a result we compare NAM answers to one of the most used search engines nowadays.**

***Index Terms*—Transparent Intensional Logic, Knowledge Base, Dolphin, Inference, Temporal aspect, News answering machine**

I. Introduction

Today's web searching tools are mainly based on full-text search methods which detect key words in the input text and then look them up by fast database algorithms [6]. Such approach is efficient in speed, but it inevitably ignores time information and natural form of human communication.

The Dolphin knowledge base system that is presented in this paper is based on higher-order typed Transparent Intensional Logic (TIL) that is capable of analyzing natural language (NL) and representing its meaning in an algorithmically accessible form. In contrast to other logic systems, TIL is designed to properly analyze time, personal attitudes and belief sentences which makes it a perfect tool for NL sentences meaning representation and its algorithmic semantic analysis [1].

Dolphin uses TIL as a bridge between human language and a computer. Early stages of the project can be found in [2], where we present the first architecture design and concept. The current analysis uses more complex and powerful approaches but the original ideas still provide good learning points about Dolphin. For the presentation of actual current features of the system, we have chosen a real application based on daily news – News answering machine (NAM).

Fig. 1 displays basic parts of NAM. The news are represented by a dedicated web server that holds a set of daily news – as far as the system is not yet capable of analyzing all natural language phenomena, only certain subset of NL sentences is allowed. NLD is a module, which translates news from the form of NL text to a meaning representation language readable by a computer. The language is defined by the theory of TIL and is implemented as text-based DOLLY language that allows to effectively process lambda-calculus entities [3, p. 6-12]. An example of a NL input, it's TIL translation and the corresponding DOLLY script follows:

Apple is red.

$x\ldots\iota: \lambda w\lambda t$ [$\mathbf{red}_{wt}\, x$] [$\mathbf{Apple}_{wt}\, x$]

{**True**/o := \w\t [[[**And** [[[**red** w_Dolphin] time] x]] […]]}

The translation mechanism from NL sentences to DOLLY is modular. The currently tested implementation is based on SYNT [4]. SYNT takes a NL sentence in the Czech or English language and thanks to syntax, semantic and corpus processing; it produces the DOLLY transcription(s) of the sentence. The Dolphin NAM itself does not contain any other language specific limitation than those impacted by the selected NL-to-DOLLY translation module.

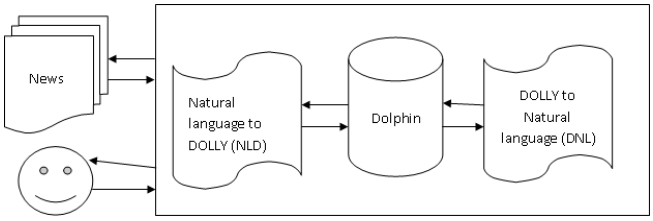

Fig. 1. Basic parts of News answering machine (NAM).

II. The Dolphin Internals

TIL works with objects of universe and their constructions instead of words. As you can see in Fig. 2A, a word depicts a construction which represents the meaning procedure pointing to the referent object. The sentence itself is, of course, also analyzed with a construction constructing a proposition (possible world and time dependent truth value, see Fig. 2B). Treating the (structural) meaning as constructions (and sub-constructions) is than clear and allows us to connect attributes and methods with the meaning objects or their algorithmic form called Dolly Construction (DC). In this way a sentence "Space is unlimited" is stored as three interconnected DCs.

A question may arise why three DCs are used as only two words (*Space, unlimited*) are present. Other interpretation of the sentence "It is true that Space is unlimited" clearly shows the answer – sentence as a whole constructs the (intensional)

Manuscript received May 24, 2012; revised July 27, 2012. This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 and by the Czech Science Foundation under the project P401/10/0792.

The authors are with Masaryk University, Faculty of Informatics, Czech republic (e-mail: xgardon@fi.muni.cz; xhorak@fi.muni.cz).

True object and dynamically defines unlimited as an attribute for the Space DC. Actually, there are more DCs that take part in the mentioned sentence - in a real analysis Dolphin/TIL works with referent possible world (w of type ω) and time moments (t of type τ) but to keep things simple we have omitted them in this example.

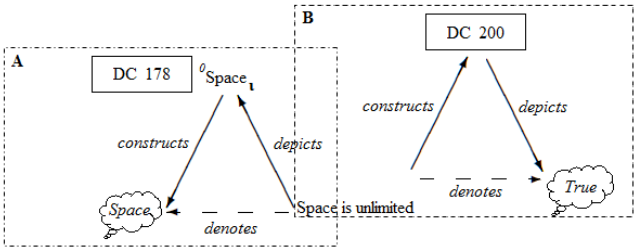

Fig. 2. Extraction of the sentence meaning.

DCs methods can provide powerful tool for actions executed by the system reflecting provided input. For example a sentence "Turn of the computer" can be connected with appropriate commands stored as a C++ method within a DC identified by the sentence. Moreover learning algorithms for automated coding of methods can be developed.

Labels (words of a language) are separated from constructions what makes the translation between different natural languages much easier. When you put a sentence in NAM, Dolphin will process it and identify the corresponding DCs. In fact, all DCs referenced by subparts of the input are presented as results of the input analysis. Let us have a sentence "The house is white" and its representation as shown in Fig. 3 (the figure presents a semantic network for the sentence). A NL translation can be obtained by switching the language database and asking DNL for output. The module will look for words in the selected language and use the appropriate syntactic rules that will produce the output.

The DNL module produces human readable output. At present, the module provides basic sentence forms but techniques from NLD can be adapted to obtain neat sentences.

Thanks to the meaning extraction such outputs can be very close to human language. As an example the question "Who is the US president?" is presented.

The TIL transcription (t1) contains the variable *X*. During the NL-to-DOLLY translation, NLD analysis the syntax and other useful language properties of the sentence.

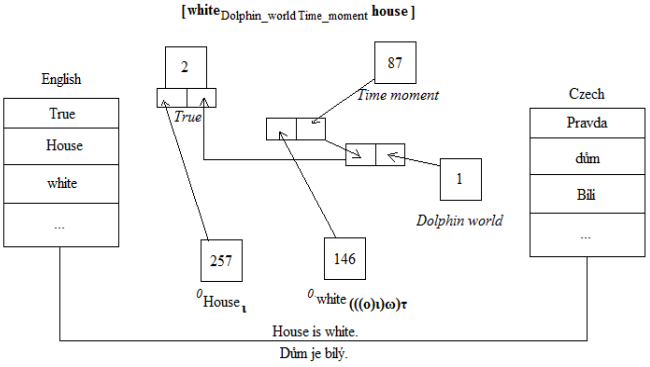

Fig. 3. Translation of the sentence in the form of semantic network.

DNL can reuse this information and combine it with the resulting value of *X* to produce the answer sentence "Barrac Obama is the US president".

$$[\,\mathbf{US\_president}_{\text{Dolphin\_world Current\_time\_moment}}\,X\,] \qquad \text{(t1)}$$

III. Variables

The last example introduces basic capability of the system– inference rules for variables. When a new variable is declared, it is considered to cover all objects of the universe that Dolphin actually knows taking into account the type of variable. So if the system knows 4 persons and we declare a variable Y of type ι (iota - individuals), this variable is connected with the set of those persons as its possible values. The concept is similar to the notion of Valuation [5, p. 51] and within the frame of Dolphin it is called the Universal Valuation. During the process of inferring the answer, the "ambiguous" Universal Valuation is used for computing the classic Valuation where each variable obtains exactly one value. The ambiguity reduction is obtained either by new input facts or by specific choice made by the user. If there is a variable in a sentence that tells something new to the system, the user must provide a way to get exactly one possible value of this variable to force the system to learn the new fact. Let us suppose that we have a system that knows two houses – one which is red and the other one is blue. Now the user enters the sentence "House is big". As the system processes the input, it finds out that there are two houses known to the system and it is not possible to identify a single object for this construction. The situation is solved by replacing "house" in the sentence with variable which has been assigned both possible objects in the Universal valuation. During the learning phase, an error is produced as Dolphin does not know which house is actually big. The user needs to provide another sentence "Red House is big" to teach the system this new piece of information.

IV. Time Oriented Projects

Time is an essential part of almost each NL sentence and within the development of NAM time plays an important role. Today's database systems do not have the capabilities of handling time information naturally. Several projects try to bring up search engines with time support. WolframAplha[1] is an example of such a project. Although it is able to process temporal aspects, it is not able to communicate in natural language and it does not seem to do so in the future. TrueKnowledge[2] is another example, much closer to natural language and it has some neat inference abilities. This project is focused on a search engine and its answers lack module NL synthesis as the DNL module in Dolphin. Another important difference of Dolphin, when compared with these projects, is the fact that Dolphin is designed to run on standard computer hardware.

[1] http://www.wolframalpha.com/about.html
[2] http://www.trueknowledge.com/technology

V. Time in Dolphin

TIL uses a specialized type for time information – τ (tau, [5]). This type represents the time continuum, thus it is isomorphic with real numbers. The logical analysis of NL verbs is viewed through a series of frames called Events [5, p. 64] that together form the whole process called an Episode [5, p. 68]. Every single event is connected with a particular time moment and the number of events in Dolphin depends on the smallest unit we want to handle. Let us say the unit is one second. Action lasting one hour needs 3600 events to be analyzed properly when using discrete events. As a solution, Dolphin works with time intervals and simulate all theoretical parts of TIL. In this way a sentence "I've been at work for two hours" can be processed as one action connected with interval "two hours" instead of a set of 7200 events. The pitfalls of this solution lay in the (Dolphin specific implementation of the) interval types. The last example uses a continuous time interval but in a sentence "I've have been there once during last two days" we have an interval which has a continuous interval in it. That one represents a particular part of the last two days when the activity took time. The position of this continuous interval is unknown as the sentence does not provide it but as can be seen, the sentence has some time information

(Fig. 4). Such interval is represented by **Onc** object [5, p. 78-85].

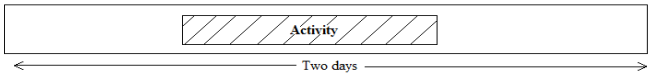

Fig. 4. *"Two days"* interval with one continuous interval in it.

The current Dolphin implementation works with continuous intervals. Processing such intervals requires an inference mechanism to avoid situations where two sentences produce a paradox. Let us have two sentences

It was raining during the whole last hour. (s1)
It was not raining during the last ten minutes. (s2)

It is clear that one of the sentences must be false and the system should produce an error message in case of learning both sentences. Supposing sentence *(s1)* was learned first, the second one *(s2)* must be rejected and the user should be informed about the contradiction with the first one. The time inference rules were developed to avoid such situations and they work in a natural way. The rule set includes rules for interval join, parent interval search, compatibility, intersect paradox check, time questions answering and time variable solving. In a theoretical form we have rules for all types of intervals but the presented Dolphin system processes continuous intervals inference logic only [3, p. 46].

VI. Tenses in Dolphin

In the previous parts, we have presented the basic time support in the system. However, natural language utterance needs a way of handling grammatical tenses. Everyday communication includes a lot of time aspects hidden in those grammatical phenomena. It is the task of the NLD (NL analysis) module to analyze each input properly and to provide correct formal interpretation. The theory behind it can be read from [5] where all English tenses are also discussed.

Briefly, the most important part of the temporal information is formed by an actor, an activity and a time interval. A special object/function **Does** (or **Do**) is used to connect them with a cooperation of the tense specific object. An example sentence "Peter killed the mole" in the notation of TIL [5, p. 81] shows us such an interconnection.

$$\lambda w \lambda t \left[\mathbf{P}_t \left[\mathbf{Onc}_w \lambda w_1 \lambda t_1 \left[\mathbf{Does}_{w_1 t_1} \mathbf{Petr} \left[\mathbf{Perf}_{w_1} [\mathbf{zničit\ Krtek}]_{w_1} \right] \right] \right] \mathbf{Anytime} \right]$$

The **Does** object is a good example of an *Atomic inference* that is discussed further. It provides rules for deduction of facts concerning action events and the time interval of the sentence. The **P** object stands for Past tense and is used to modify the **Anytime** object representing a time interval. As the sentence is in the past tense it makes sure that only the past part of **Anytime** is used (we cannot take any future point into account). To achieve this, the parameter *t* representing the actual time moment must be provided. The modified **Anytime** interval is then processed by the object named **Onc**, which controls the type of interval with regard to the frequency. In this case **Onc** produces an interval meaning "once upon anytime something happened". Finally the **Does** object connects the input interval with the action built from **Petr** as the actor, **kill** as the verb and **Mole** as an object of killing. Behind it there are rules for avoiding the time paradox, interval joining etc. as mentioned in the *Time in Dolphin* section. The whole process is rather complex and an interested reader should consult [5] for the theory introduction and [3] for the implementation details.

VII. Inference

Every logic oriented system should provide an inference module that can deduce new facts from the stored ones. The actual solution of this task depends heavily on the design of the system architecture. The implementation can use the technique of coding the inference rules in the form of a High-level computer language, or the functional paradigm for inference rules definitions can be employed. Both the approaches have their pros and cons. The Dolphin system tries to take advantage of both of them. Dolphin deduction uses three ways of inference:

Atomic inference – case of the direct coding technique. The user has an opportunity to enter specific steps for deducing new information in the form of C++ procedure. This way allows writing fast inference rules as programs, which are compiled to the language of the desired hardware. A drawback is the need of specifying exact steps which are not dynamic but discrete. It is clear, that the natural language is not discrete, but in some situations the Atomic inference provides the best way of fast deduction.

Nullary inference – the simplest inference rule. When an input is processed not all sentence specific DCs are identified. The best way to understand the Nullary inference is to take a sentence: "An apple is red" and its TIL transcription

$$\lambda w \lambda t\, \lambda X\, [\, \mathbf{Red}_{wt}\, X\,] \wedge [\, \mathbf{Apple}_{wt}\, X] \qquad (t2)$$

The **Apple** has the type of $(((o)\iota)\tau)\omega$. This can be interpreted as a function waiting for a parameter of type ω and then returning an DC of type $((o\iota)\tau)$ back. By an application of the **Apple** function to the **Dolphin_world**/ω, a new DC is created. The sentence says nothing specific about it but we know it exists as it is than applied to a time object and the result is applied on an individual object. This also shows the difference between a referent object and a DC – an object of the universe is undetermined, but the DC can be created for further processing. *Nullary inference* creates a new DC in the knowledge base which can be described by other sentences or even can be joined with another DC. This happens with the final DC obtained from the sentence analysis. We declare the sentence as True, which can be represented by an equation in Fig. 5. Due to the internal logic of the Dolphin system, both parts of this equation are processed separately. This means we need real DCs one each side. The *Nullary inference* produces a new DC on the right side which is then joined and replaced by the **True** object.

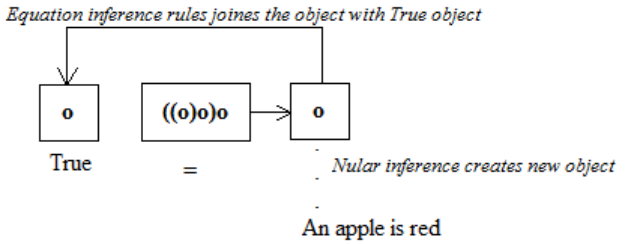

Fig. 5. Example of using inference rules to deduce new fact.

Complex inference – is based on rules and techniques similar to Prolog language (except the fact Prolog is first order logic system only and Dolphin tries to overcome this restraint). The *Complex inference* provides a way for handling sentences like "Every apple is red. This is an apple. Is this apple red?"

Classic forms of the mentioned inference are time consuming as all rule premises must be identified and the best rule for deducing must be chosen. The aim of the Complex inference is to use tree like structures for each input, its objects and also inference rules. Tree similarity algorithms are then used to build a set of the best inference rules with possible premises unification during the first contact with an input. If we look back on the sentence "This is an apple" and assume we already know the fact that "Every apple is red", further processing then produces a set containing this fact. The goal can be achieved using the word "apple" from the input.This technique is in early stage and details are the question of future articles.

VIII. NAM PRESENTATION

To show the abilities of the Dolphin system we have chosen newspaper articles as a good area for question-answering. We have adapted news from www.bbc.com on 26th and 30th of September 2010 with the concentration on headlines and prepared a set of sentences to be stored in the NLD module. The standard Dolphin system uses SYNT in its NLD module that is capable to provide TIL transcriptions for Czech language.

In order to obtain an example NLD module for English, the output of such NLD module was simulated by human transcription to the DOLLY script in the database. The obtained DOLLY script is then processed by Dolphin.

At start NLD contains those sentences:

Virus affected nuclear power plant in Iran on 26th of September. (s3)
Chavez faced election challenge on 26th of September. (s4)
Election took place in Venezuela on 26th of September. (s5)
Hugo Chaves is a president of Venezuela. (s6)
Jewish group sets sail for Gaza on 26th of September (s7).

To present the Dolphin abilities as clear as possible we assume all sentences to be informing about news that took place during the whole day of 26th of September.

To demonstrate the output of the NLD module, the DOLLY script of the first sentence follows:

$$x = \mathbf{Power\_plant}\ldots\iota;\ \mathbf{Iran}\ldots\iota;\ \mathbf{in}\ldots(((((((o)\iota)\tau)\omega)\iota)\tau)\omega)$$
$$\mathbf{True} :: [\ \mathbf{in}_{w\_Dolphin\ time}\ \mathbf{Iran}\ ]\ x$$
$$\mathbf{True} :: \lambda w \lambda t\, [\ \mathbf{P}_t\, [\ \mathbf{Thr}_{w\_Dolphin}\, \lambda w_1 \lambda t_1 [\ \mathbf{Does}_{w1t1}\ \mathbf{Virus}\ [\ \mathbf{Imp}_{w1}\ [\ \mathbf{affect}\ x\ ]_{w1}\ ]\ ]\ ]$$
$$\mathbf{26^{th}\ of\ September}\ ] \qquad (t2)$$

Now let's ask questions:

What affected nuclear power plant on 26th of September? (q1)
Where is the power plant? (q2)
What did Chavez face on 26th of September? (q3)
Where was election on 26th of September? (q4)
Who is Hugo Chaves? (q5)
Who did set sail on 26th of September? (q6)

The first question *(q1)* is processed by the NLD module with the result almost identical to the *(t2)* TIL script:

$$x = \mathbf{Power\_plant}\ldots\iota$$
$$\mathbf{True} :: \lambda w \lambda t\, [\ \mathbf{P}_t\, [\ \mathbf{Thr}_{w\_Dolphin}\, \lambda w_1 \lambda t_1 [\ \mathbf{Does}_{w1t}\, y\ [\ \mathbf{Imp}_{w1}\ [\ \mathbf{affect}\ x\ ]_{w1}\ ]\ ]\ ]\ \mathbf{26^{th}}$$
$$\mathbf{of\ September}\ ] \qquad (t3)$$

The only difference is in variable *Y* that replaced the **Virus** object in the original sentence and the definition of the *X* variable which is not refined by place (*in Iran*). By solving this input with cooperation of rules defined in the *Variables* section, the output of "Virus" is provided.

The second question is coded as:

$$x = \mathbf{Power\_plant}\ldots\iota;\ \mathbf{True} :: [\ \mathbf{in}_{w\_Dolphin\ time}\ y]\ x$$

and we again have a script similar to the original one excluding the **Iran** object replaced by variable *Y*. The resulting answer is: *Iran*.

As can be seen, many traditional questions can be solved by replacing particular objects by a variable. So when we want to analyze the sentence "Where is the power plant", the NLD module translates it into the form "Power plant is in X" with variable *X* and then analyzes it as an input phrase. The Dolphin internal mechanism tries to refine the *Universal*

valuation using the actual knowledge and to find the particular object represented by *X*. This approach allows neat answers by replacing the variable *X* with a solution – "Power plant is in Iran".

The rest of the example questions produce outputs: "Election" for *(q3)*; "Venezuela" for *(q4)*; "president of Venezuela" for *(q5)* and "Jewish group" for *(q6)*.

To compare Dolphin to an existing search engine, on 30th of September Google produced the following answers to questions *(q1, q3, q4):*

Q1: FOXNews.com – Computer Worm Affects Computers at Iran's First
Q3: Hugo Chávez - Wikipedia, the free encyclopedia
Q4: Venezuelan parliamentary election, 2010 - Wikipedia

Results are close to desirable answers but one has to seek more in the provided links, especially for question *(q3)*. Moreover *(q2)* cannot be answered in Google as it is context dependent.

Dolphin has the ability to distinguish personal attitudes. Let's have a sentence:

USA thinks Yuan is kept low. (s7)

We can ask then "Is Yuan kept low?" or "Does China think Yuan is kept low?" and get the answer: "I do not know". This is due to the fact that only USA thinks that and no one else (as far as only sentence *(s7)* is provided). From the theoretical point of view, this phenomenon is solved with the notion of possible worlds [5, p. 42].

Another capability of the Dolphin system is represented by sentences:

Tony Curtis is a film star. (s8)
George Clooney is a film star. (s9)
Film star died on 30th September 2010. (s10)

Respecting the order, we first enter *(s8)* than (*s9)* and finally *(s10)* in the NLD module. During the processing of *(s10)* by the Dolphin knowledge base, the system identifies more than one DCs connected with the construction of "film star". It replaces the construction with a variable having all individuals (*Tony Curtis, George Clooney*) in the *Universal valuation*. The processing continues in the standard way until the information that "Film star died" is going to be learned. In this situation the system needs to know the actual subject and as we did not provided any further sentence about *Film star*, the system asks: "Which film star?" Just after the user enters "Tony Curtis", the new information is learned.

IX. Future Research

The Dolphin NAM system is in an early stage and it is continuously developed. The first goal to reach is the final implementation of the complex time support including X-times intervals and all natural language tenses. The full inference module is an upcoming task as *Atomic* and *Nullar* inference techniques are not sufficient for all natural language phenomena. The user communication is essential for the system usability so an intensive research and development is also going on with the DNL module.

References

[1] P. Tichý, *The foundations of Frefes Logic*, Berlin: De Gruyter, 1988, ISBN 978-3-11-011668-7.
[2] A. Gardoň, "Design of Knowledge Base and Basic Inference Machine for TIL," Bachelor thesis in Slovak, Faculty of Informatics, Masaryk University, Brno, Czech Republic, pp. 61, 2007.
[3] A. Gardoň, "Querying of Temporal Information over Knowledge in Transparent Intensional Logic," Master thesis in Slovak, Faculty of Informatics, Masaryk University, Brno, Czech Republic, pp. 87, 2010.
[4] A. Horák, *Computer Processing of Czech Syntax and Semantics*, 1st edition, Brno, Czech Republic: Librix.eu, pp. 241, 2008.
[5] A. Horák, "The Normal Translation Algorithm in Transparent Intensional Logic for Czech," PhD thesis, Faculty of Informatics, Masaryk University, Brno, Czech Republic, pp. 155, 2001.
[6] Ian R Winship, "World-Wide Web searching tools: an evaluation," *VINE*, vol. 25, no. 2, pp.49 – 54, 1995.

A. Gardoň studied Computer Science at the Masaryk University (MU) in Brno, Czech republic and received M. Sc in natural language processing and knowledege representation in 2010.

Now he is an student of doctoral degree programme in combined form at the Faculty of Informatics MU and he researches Transparent intensional logic with its practical applications. The aim of his work is to develop a system for time and space inference over knowledge extracted from natural language sentences. He works as a SAP ABAP programmer in Bratislava, Slovakia.

A. Horák studied Computer Science at the Masaryk University (MU) in Brno, Czech republic and received Ph.D. in syntactic and logical analysis of natural language in 2002.

Now, he is working as an associate professor at the Faculty of Informatics MU teaching Artificial Intelligence and Computational Linguistics. He also works as a senior researcher at the Natural Language Processing Centre of MU leading teams working on syntactic analysis of free-word-order languages, logical analysis using the Transparent Intensional Logic system and the development of the DEB (Dictionary Editor and Browser) lexicographic platform that is used by hundreds of people and teams all over the world including teams developing national WordNet semantic network resources.