

Selected Aspects of Temporal Knowledge Engineering

Maria Mach-Król
University of Economics
ul. Bogucicka 3,
40-226 Katowice Poland

Email: maria.mach-krol@ue.katowice.pl

Krzysztof Michalik
University of Economics
ul. Bogucicka 3,
40-226 Katowice Poland

Email:
krzysztof.michalik@ue.katowice.pl

Abstract—The paper presents some problems of logical coherence while reasoning temporally. It shows the importance of these problems in some application domains for temporal intelligent systems, e.g. in legal domain. It then presents Logos reasoning tool and its inference techniques, it also shows how Logos can handle temporal rules now, and it points out what should still be done in order to make the system resistant to temporal reasoning logical problems.

I. INTRODUCTION

TIME indispensable representation in many artificial and temporal intelligence reasoning systems [1]. It is so, because time is a basis for reasoning about change and actions. Many AI systems concern the currently changing economic environment (see e.g. [2]). Therefore, if the decisions taken on the basis of system's advice are to be correct, the system has to take into account the temporal dimension of information, the changes of information in time and has to be aware of the nature of those changes [1]. Therefore the need for representing temporal knowledge in artificial intelligence systems is nowadays obvious. Even intuitively, one can feel the need of capturing a temporal aspect of relationships between objects in AI systems.

Representation of knowledge changing in time and temporal reasoning have to be based on some formal foundations. One of such foundations is the language of logic, both the classical as the modal one (see e.g. [3], [4]). The shortest motivation for using the temporal logic may be found in [5], where it is concluded, that “in order to introduce temporal relationships (...) it is necessary to broaden the formal apparatus with the temporal logic” (p. 429).

The paper is organized as follows. In section 2 the general view of temporal knowledge is given. Section 3 contains a few examples of temporal knowledge engineering problems. Section 4 is devoted to the discussion on inference techniques already implemented in the Logos tool. The next section shows the importance of knowledge verification as important knowledge engineering process, and how it is handled by Logos. In section 6 we provide an example of temporal reasoning in the Logos system. The paper ends with summary and conclusions.

II. KNOWLEDGE ENGINEERING PROBLEMS

The tasks for a temporal AI system encompass among others:

- maintaining temporal coherence,
- answering temporal queries,
- explanations,
- prediction, etc.

The most commonly known domains, in which there is a need for an explicit time notion, are: natural language processing (NLP), planning, robotics, image processing, medical diagnosis, and law [1], [2]. In all of the above mentioned domains, change has a primary meaning. Introducing time in an explicit way allows for reasoning about changing domains, also about the economic one. It also allows for a computer simulation of human reasoning process, because people reason about action and change [3]. In particular, there are described such notions, as change, causality or actions, therefore the proper representation of time and temporal reasoning are so important in (among others) artificial intelligence [1]. If an AI system is to simulate intelligent behavior, to adapt to changes in the environment, or to verify its beliefs, it has to be able not only to gain new knowledge, but also to keep its knowledge in an up to date state. Knowledge changes – due to two basic reasons. The first is simply the passage of time. The second reason is due to new information on objects, which possess temporal characteristics [4].

III. KNOWLEDGE ENGINEERING PROBLEMS

There exist several problems which one has to overcome in some way to reason consistently about time and change. In the paper there will be mentioned three problematic issues: the TBS (Tossed Ball Scenario), the DIP (Divided Instant Problem), and finally the FP (Frame Problem).

Tossed Ball Scenario (TBS)

The problem called the Tossed Ball Scenario is connected with the question of temporal primitives in the ontology of time. As temporal primitives there can be chosen [1]:

- time points (also called instants) – as for example in McDermott's logic;
- time periods (also called intervals) – as for example in Allen's interval calculus;
- both primitives.

In logic systems using the notion of time points (instants), there is a question of modeling continuous change (e.g. in the environment). This is considered a problematic issue, because

to do it properly one must be able to describe fluents that hold at an instant, more precisely – an isolated instant.

Divided Instant Problem (DIP)

The problem is similar in its nucleus to the TBS. It consists of establishing logical value (truth-value) of a fluent f at an instant i , if f is true on period p_1 and false on period p_2 , given that p_1 ends at i and p_2 begins at i . Of course we assume that we take instants and periods as temporal primitives. Let us discuss the problem using an example given by [5]. Consider fig. 1:

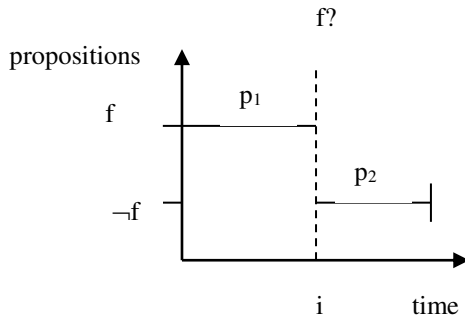


Fig. 1.: Divided Instant Problem [6]; “ f ” denotes fluent, “ p_1 ” and “ p_2 ” denote periods.

Vila gives an example concerning light. If fluent f means “the light is on”, then on p_1 light is on and on p_2 light is off. Question: is the light on or off at i ? More generally: what is a truth-value of the proposition at i ?

Frame Problem (FP)

The frame problem is one of the central theoretical issues of artificial intelligence. The problem arises when logic is used to describe the effects of actions and events. The frame problem(s) appears in all approaches to reasoning about action and change [6].

In order to be historically correct, the first description of this problem has been done by McCarthy and Hayes in 1969 [7]. While working on situation calculus they noted that a problem occurs when there are several actions available, each of which changes certain features of the situation.

The classical examples of such problems are: ‘Yale Shooting Problem’, ‘Stolen Car Problem’ and others [8]. Let’s take a look on the first example. Given the sequence of events at times as denoted by the times of their assertion $t(n)$:

- $t(0)$: I (now) pick up a loaded gun
- $t(1)$: I unload it
- $t(3)$: I point the gun at my head
- $t(4)$: I pull the trigger

The answers to the question: ‘Am I alive?’ should be negative as the representation did not contain the information that the gun was still unloaded (there is no information about the change of fact - gun is unloaded in $t(3)$).

There were already many attempts to solve the above enumerated logical problems, e.g. by [5] or by [9]. Especially the results obtained by Reiter are interesting, as he uses

situation calculus. And this formalism or its mutation called Golog [10] are planned by us to be implemented in the intelligent system Logos. Therefore we hope Logos will be able not only to reason temporally, but to do it in a consistent way, that is the tool will be resistant to temporal logic inference problems. Some previous remarks on reasoning temporally about the legal domain in Logos were published in [11]. Logos now achieved the status of a research prototype and is treated as environment and reasoning engine for temporal knowledge bases experiments.

In the next section we will present currently implemented reasoning strategies in Logos, and we will show how the tool can now handle temporal rules.

IV. INFERENCE IN LOGOS

In present version of Logos system we successfully implemented four variants of inference:

1. Top-down inference using two-valued logic,
2. Top-down inference using Stanford Certainty Factor Algebra,
3. Bottom-up inference using two-valued logic,
4. Bottom-up inference using Stanford Certainty Factor Algebra.

Each of these methods of inference can operate in two modes: with askable or not askable conditions. The top down inference is goal-driven and is similar to that used in logic programs with backtracking mechanism which makes it possible to find all solutions to a given problem. Generally this method of inference is based on resolution principle developed by Robinson [12]. The whole process of inference starts with a given goal:

$$\langle - C_1, \dots, C_m \rangle$$

In procedural interpretation each step of computation relies on matching of a given goal C_i with the head of any procedure in knowledge base. Procedure is Horn clause (rule) of the following form:

$$A \leftarrow B_1, \dots, B_n$$

Where $\{C_1, \dots, C_k, A, B_1, \dots, B_m\}$ are atomic formulas. In our implementation those atomic formulas take the form $\langle O, A, V \rangle$, where O is identifier of object, A is identifier of attribute and V represents value. Value can be symbol (more generally any string), numeric constant or variable.

If the mentioned matching is not possible, we call it failure. If the matching of C_i is successful, then current goal is being reduced to the following form:

$$\langle - (C_1, \dots, C_{i-1}, B_1, \dots, B_n, C_{i+1}, \dots, C_m) \Theta \rangle,$$

where Θ is called matching substitution, mainly concerned with assignment of temporary values to variables.

The inference process finishes when the goal is reduced to empty clause.

The bottom-up inference, as implemented in our system, starts from facts and using rules of knowledge base finishes with generation of set of conclusions.

While building inference module of Logos we assumed that in temporal knowledge base some parts of knowledge (facts or rules) can be to some degree uncertain, so we implemented

Stanford Certainty Factor Algebra. We chose this method because of the simplicity of its use and good practical verification. In this method for given rule R:

$$R: C \text{ if } W \text{ with } CF(R)$$

is assigned certainty factor $CF(R)$ and its value is in the range $\langle -1; 1 \rangle$. The current CF for a conclusion of the rule R is dynamically computed in the following way:

$$CF(C) = CF(R) * CF(W) .$$

If antecedent of a given rule consists of a set of conditions e.g. W_1 and W_2 joined by disjunction or conjunction its value is calculated as follows:

$$CF(W_1 \wedge W_2) = \text{MIN} \{ CF(W_1), CF(W_2) \}$$

$$CF(W_1 \vee W_2) = \text{MAX} \{ CF(W_1), CF(W_2) \}$$

During inference, especially using bottom-up (forward chaining) method it is possible that several rules add the same conclusion to the working memory of the knowledge base. In such situation the CF of the added conclusion have to be dynamically changed. Calculation of the new CF for two rules R_1 and R_2 is as follows:

$$CF(C) = CF(R_1) + CF(R_2) * (1 - CF(R_1))$$

if $CF(R_1) > 0$ and $CF(R_2) > 0$

$$CF(C) = CF(R_1) + CF(R_2) * (1 + CF(R_2))$$

if $CF(R_1) < 0$ and $CF(R_2) < 0$

$$CF(C) = (CF(R_1) + CF(R_2)) / (1 - \text{MIN} \{ |CF(R_1)|, |CF(R_2)| \}) ,$$

when the signs of $CF(R_1)$ and $CF(R_2)$ are different.

The next step of our empirical research will attempt to implement inference of temporal situation calculus method using reasoning schemes of already implemented methods. Problem of uncertainty will be important part of the temporal inference and knowledge representation language.

Logos inference engine is equipped with rich set of explanation facilities, among others:

- How explanations,
- Why explanations,
- Metaphors,
- What is explanations and
- Facts descriptions.

How explanations show the way the conclusion has been derived and can be used after the end of reasoning process. Why explanations explain the reason system asked a question during consultation. In this case Logos shows the current context of reasoning and shows in what way the answer will contribute to solving the problem. Metaphors enable knowledge engineer to attach more textual information about selected rules, what can be useful at initial stage of using the application. What is explanation provide more detailed, textual information about conclusions as well as some questions. It is also possible to attach some explanations to facts, in the form of facts descriptions, e.g.: source of information or availability of deeper/further How explanations (in the blackboard architecture) showing how the fact has been derived during consultation.

The mentioned explanations are equally important in usual knowledge bases and in the temporal ones.

V. KNOWLEDGE VERIFICATION AS IMPORTANT KNOWLEDGE ENGINEERING PROCESS

As it has been already mentioned, one of the main aims of our research is creation of temporal knowledge base using Logos system. To this end we started building special reasoning system called Logos which will be kind of experimental environment. One of our assumptions is that appropriate verification algorithms are necessary to provide solid foundation for temporal reasoning. At present we implemented broad range of knowledge base anomalies detection procedures, among others [13]:

1. Redundant rules,
2. Subsuming rules,
3. Contradictory rules,
4. Recursive rules (circular loop).

Ad. 1.

Two rules we call redundant if

$$R_i \leftarrow C_{i1} \wedge \dots \wedge C_{in} \text{ and } R_j \leftarrow C_{j1} \wedge \dots \wedge C_{jn}$$

where R_i and R_j are conclusions and C are conditions and $i \neq j$, holds: $\{ C_{i1}, \dots, C_{in} \} = \{ C_{j1}, \dots, C_{jn} \}$.

Ad. 2.

If for two different rules: $R_i \leftarrow C_{i1} \wedge \dots \wedge C_{im}$ and $R_j \leftarrow C_{j1} \wedge \dots \wedge C_{jn}$ and $i \neq j$,

holds $\{ C_{i1}, \dots, C_{im} \} \subset \{ C_{j1}, \dots, C_{jn} \}$, then we

say that rule R_i subsumes rule R_j .

Ad. 3.

Two rules are regarded as contradictory if $R_i \leftarrow C_{i1} \wedge \dots \wedge C_{in}$ and $R_j \leftarrow C_{j1} \wedge \dots \wedge C_{jn}$ where $i \neq j$.

Ad. 4.

We distinguish in Logos two kinds of the recursion: direct and indirect. The notion of direct recursion is consistent with notion of circular rule set by Vermesan [14]: a rule set is circular iff the antecedents cannot be derived from any other rule except given rule consequent.

Typical situations are as follows:

$$Q_1 \leftarrow Q_1, P_1, \dots, P_n$$

$$Q_1 \leftarrow P_1, \dots, P_1, Q_1, P_m, \dots, P_n$$

$$Q_1 \leftarrow P_1, \dots, P_n, Q_1$$

In practice the circular loop can be much more complicated and engage some set of rules (see fig. 2). Detection of such situation is more difficult but is necessary the knowledge base to function properly. Exemplification of this kind of indirect recursion can be as follows:

$$Q_1 \leftarrow P_{11} \wedge \dots \wedge P_{1j} \wedge \dots \wedge P_{1x}$$

$$Q_i \leftarrow P_{i1} \wedge \dots \wedge P_{i1} \wedge \dots \wedge P_{iy}$$

$$Q_n \leftarrow P_{n1} \wedge \dots \wedge P_{nm} \wedge \dots \wedge P_{nz}$$

where $P_{1j} = Q_i, P_{i1} = Q_n, P_{nm} = Q_1$.

where “ \Leftarrow ” has more general meaning and denotes beside equality/identity ability of instantiation and matching. Our system detects all levels of recursive interconnections between rules, what is important to guarantee correct behavior of knowledge base in practical use.

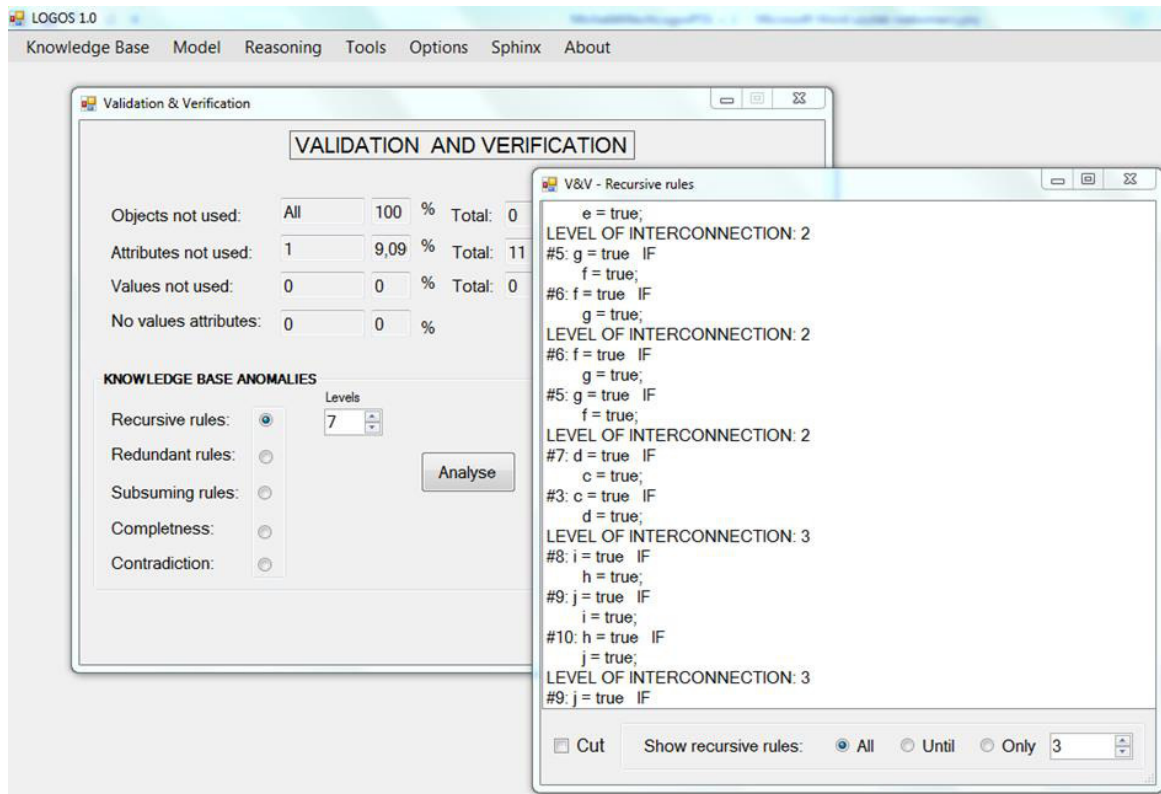


Fig. 2. Sample circular loop in Logos.

Source: own elaboration.

What is very important, we expect that full implementation of temporal knowledge base and temporal inference will require some specific verification algorithms in response to anomalies in temporal knowledge base. At this stage of our empirical research it is difficult to precisely point out these anomalies, it will be subject of further investigations.

VI. TEMPORAL REASONING IN LOGOS

To show how Logos performs temporal reasoning, we have chosen an example arising from Polish act on economic activity. It concerns the obligation to obtain a license for certain economic activities: a certain enterprise wishes to perform an activity, that needs a license obtained from the state.

The act issued in 1988, that has been valid from January 1st 1989 till December 31st, 2000, stated, that “A license is given indefinitely”; while the act issued in 1999, valid from January 1st, 2001 till July 19th, 2004 stated: “A license is given for a specific period of time, not shorter than 2 years and not longer than 50 years”. The actual act of July 2nd, 2004, which is valid from July 20th, 2004 till now, states that “A license is given for a specific period of time, not shorter than 5 years and not longer than 50 years”.

While analyzing the above statements as a temporal legal knowledge, we can distinguish such temporal elements as events (an act of granting an enterprise a license), objects (a

license), temporal relations (not shorter than, not longer than), temporal constants (2 years, 5 years, 50 years).

The basic temporal elements are points (e.g. the date when a license is granted) and intervals (e.g. the period for which a license is valid). We may also consider – in a broader perspective – an event consisting of “license withdrawal” which of course always happens before the period of license validity ends. Therefore while formalizing legal statements one should choose a formalism that keeps total linear order of time, with precedence relation. The formalism has to be a point-interval one, as we have both types of basic temporal elements to be taken into consideration. All the above conditions are fulfilled by the model of calendar time.

If we take into account the regulations coming from both acts, we immediately see two of the three temporal aspects of legal knowledge, discussed earlier. These are namely “law in time” (three periods of legal acts validity) and “time in law” (a period for which a license is granted) aspects. The third aspect – “transitional law” – will not be discussed here.

The above cited law articles of our example may be also written in the form of general rules:

If a license is issued, then it is valid indefinitely (Act of 1988)

If a license is issued, then it is valid for a period not shorter than 2 years and not longer than 50 years (Act of 1999)

If a license is issued, then it is valid for a period not shorter than 5 years and not longer than 50 years (Act of 2004)

The rules may be formalized e.g. in the Legal Temporal Representation (LTR) language [15], and they are as follows:

```
Attributes(license, {who_issues, who_gets
})
Attributes(is_issued, {what})
Attributes(valid, {what}, _,)
Granularity(day)
```

Rules for the 1988 Act:

Rule 1:

```
If TT1:license(issuing_authority,
enterprise)
  TT2: is_issued(TT1)
  Occurs(TT2)
Then Occurs(valid(TT1), instant(TT2))
```

Rule 2:

```
If TT2: valid(TT1)
  Occurs(TT2)
Then Holds_on(valid(TT1), period(TT3))
  Period(TT3) Equals [instant(TT2),
+inf]
```

The above rules state, that if a certain enterprise has been granted a license issued by a certain authority and it happened in the point (day) stamped by token TT1, then the license is valid from this day, and for indefinite time (that is, over an interval from TT1 to infinity).

Rules for the 1999 Act:

Rule 3:

```
If TT1:license(issuing_authority,
enterprise)
  TT2: is_issued(TT1)
  Occurs(TT2)
Then Occurs(valid(TT1), instant(TT2))
```

Rule 4:

```
If TT2: valid(TT1)
```

```
Occurs(TT2)
```

```
Then Holds_on(valid(TT1), period(TT3))
  Period(TT3) Equals [2y, 50y]
```

Rules for the 2004 Act:

Rule 5:

```
If TT1:license(issuing_authority,
enterprise)
  TT2: is_issued(TT1)
  Occurs(TT2)
Then Occurs(valid(TT1), instant(TT2))
```

Rule 6:

```
If TT2: valid(TT1)
  Occurs(TT2)
Then Holds_on(valid(TT1), period(TT3))
  Period(TT3) Equals [5y, 50y]
```

As it can be easily seen, rules 4 and 6 differ from rule 2 only in the length of a period over which a license is valid, while rules 1, 3 and 5 are identical. Example of the use of the discussed rules for the reasoning along with HOW explanations in the Logos system is shown in fig. 3.

VII. SUMMARY AND CONCLUSIONS

As it has been mentioned, in some areas, especially in the legal domain, temporal reasoning is natural part of problem solving. Therefore it should be taken into account while building real-life decision support systems in that domain. Our research is aimed to add temporality to knowledge representation of legal knowledge for building more adequate knowledge base. At present stage of our work we are trying to build temporal knowledge base using Logos reasoning system. Important part of any knowledge-based system should be module for automatic detection of all possible anomalies in the knowledge base, so we implemented wide range of special algorithms for that purpose. The next step it will be adding more temporal relations to our system, and to implement temporal reasoning in the Situation Calculus in order to overcome logical inference problems, pointed out in this paper. If we are succeed as far as building prototype temporal knowledge base, we will try to implement such application into practice.

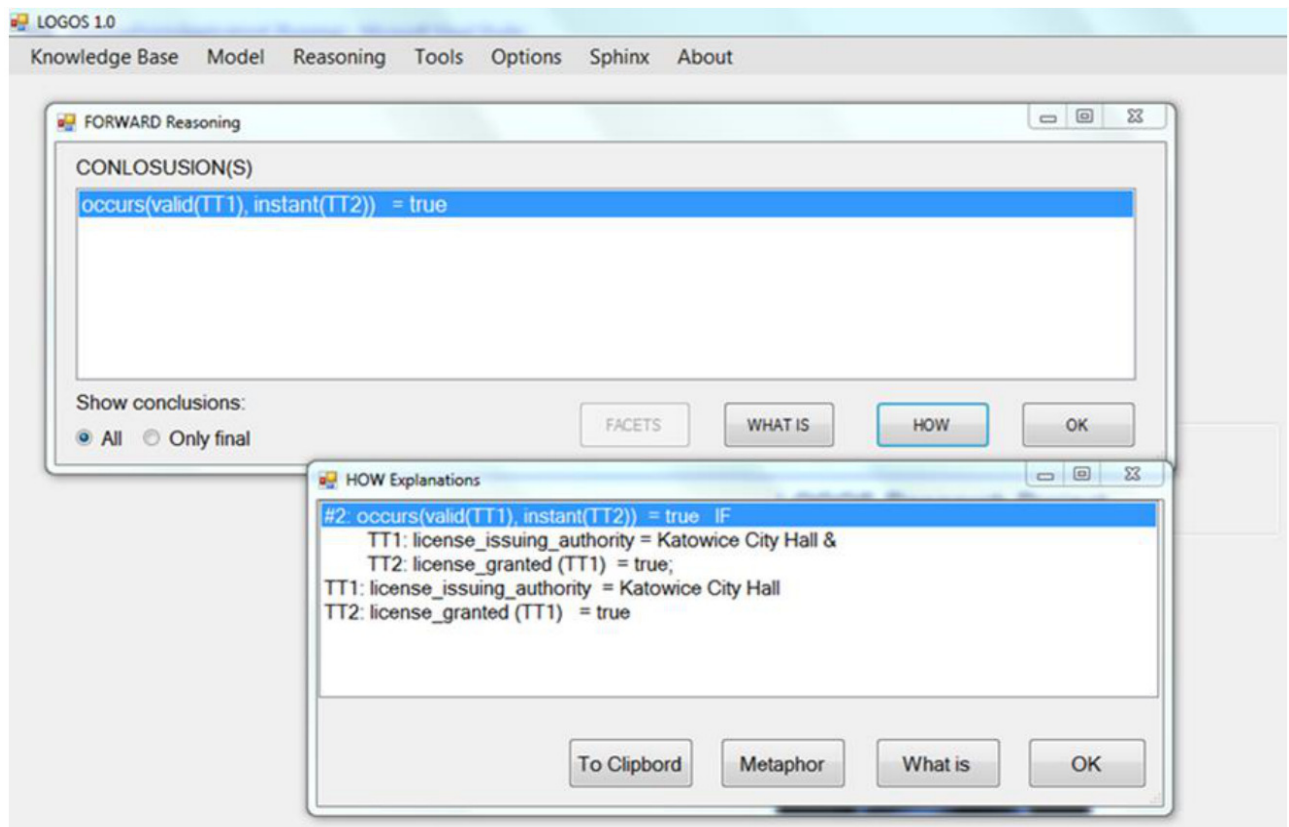


Fig. 3. Example of reasoning using temporal rules and HOW explanation facilities of Logos.
Source: own elaboration.

REFERENCES

- [1] L. Vila, „A Survey on Temporal Reasoning in Artificial Intelligence,” *AI Communications*, nr 7(1), pp. 4-28, 1994.
- [2] T. Bench-Capon and F. Coenen, "The maintenance of legal knowledge based systems," *Artificial Intelligence Review*, vol. 6, no. 2, 1992.
- [3] A. Galton, „Time and Change for AI,” in *Handbook of Logic in Artificial Intelligence and Logic Programming. Vol. 4: Epistemic and Temporal Reasoning*, D. Gabbay, C. Hogger i J. Robinson, Eds., Oxford, Clarendon Press, 1995.
- [4] J. Benthem van, „Temporal Logic,” in *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 4: Epistemic and Temporal Reasoning*, D. M. Gabbay, C. J. Hogger, and J. A. Robinson, Eds., Oxford, Clarendon Press, 1995, pp. 241-350.
- [5] L. Vila and E. Schwalb, "Revisiting Time and Temporal Incidence," in *Proc. of the AAAI'96 Workshop on Spatial and Temporal Reasoning*, 1996.
- [6] M. Fisher, D. Gabbay and L. Vila, Eds., *Handbook of Temporal Reasoning in Artificial Intelligence*, Amsterdam: Elsevier, 2005.
- [7] J. McCarthy and P. Hayes, "Some philosophical problems from the standpoint of artificial intelligence," *Machine Intelligence*, vol. 4, pp. 463-502, 1969.
- [8] A. Zambak, „The Frame Problem,” in *Philosophy and Theory of Artificial Intelligence*, V. Müller, Ed., Berlin Heidelberg, Springer, 2013, pp. 307-319.
- [9] R. Reiter, „The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression,” in *Artificial intelligence and mathematical theory of computation: papers in honor of John McCarthy*, V. Lifschitz, Ed., San Diego, Academic Press Professional, Inc., 1991, pp. 359-380.
- [10] J. Claßen, „Planning and Verification in the Agent Language Golog,” Aachen, 2013.
- [11] M. Mach-Król and K. Michalik, "An Intelligent System with Temporal Extension for Reasoning About Legal Knowledge," in *Looking into the Future of Creativity and Decision Support Systems*, A. Skulimowski, Ed., Kraków, Progress & Business Publishers, 2013, pp. 360-369.
- [12] J. Robinson, „A Machine-Oriented Logic Based on the Resolution Principle,” *Journal of the ACM*, Vol. 12, No. 1, pp. 23-41, Jan. 1965.
- [13] K. Michalik, M. Kwiatkowska and K. Kielan, "Application of Knowledge-Engineering Methods in *Medical Knowledge Management*,” in *Fuzziness and Medicine: Philosophical Reflections and Application Systems in Health Care, vol. III*, R. Seising and M. E. Tabacchi, Eds., Berlin Heidelberg, Springer, 2013, pp. 205-214.
- [14] A. Vermesan, „Foundation and Application of Expert System Verification and Validation,” w *The Handbook of Applied Expert Systems*, J. Liebowitz, Ed., Boca Raton, CRC Press LLC, 1998, pp. 5-1-5-32.
- [15] L. Vila and H. Yoshino, "Time in automated legal reasoning," *Information and Communications Technology Law*, vol. 7, no. 3, 1998.