Fundamenta Informaticae 124 (2013) 133–151
DOI 10.3233/FI-2013-828
IOS Press

Automatic Learning of Temporal Relations Under the Closed World Assumption

M. C. Nicoletti
DC, UFSCar, S. Carlos & FACCAMP, C. L. Paulista, SP, Brazil
carmo@dc.ufscar.br

F. O. S. S. Lisboa
IFSC, USP, S. Carlos, SP, Brazil
flavia@ifsc.usp.br

E. R. Hruschka Jr.
DC, UFSCar, S. Carlos, SP, Brazil
estevam@dc.ufscar.br

Abstract. Time plays an important role in the vast majority of problems and, as such, it is a vital issue to be considered when developing computer systems for solving problems. In the literature, one of the most influential formalisms for representing time is known as Allen's Temporal Algebra based on a set of 13 relations (basic and reversed) that may hold between two time intervals. In spite of having a few drawbacks and limitations, Allen's formalism is still a convenient representation due to its simplicity and implementability and also, due to the fact that it has been the basis of several extensions. This paper explores the automatic learning of Allen's temporal relations by the inductive logic programming system FOIL, taking into account two possible representations for a time interval: (i) as a primitive concept and (ii) as a concept defined by the primitive concept of time point. The goals of the experiments described in the paper are (1) to explore the viability of both representations for use in automatic learning; (2) compare the facility and interpretability of the results; (3) evaluate the impact of the given examples for inducing a proper representation of the relations and (4) experiment with both representations under the assumption of a closed world (CWA), which would ease continuous learning using FOIL. Experimental results are presented and discussed as evidence that the CWA can be a convenient strategy when learning Allen's temporal relations.

Address for correspondence: DC-UFSCar, Via Washington Luiz, km. 235, 13565-905, S. Carlos, SP, Brazil
e_mail: carmo@dc.ufscar.br

Keywords: time representation, Allen's temporal algebra, automatic learning of temporal relations, inductive logic programming, FOIL, time points, time intervals, learning under closed world assumption.

1. Introduction

The notion of time is present everywhere, embedded or permeating almost every single aspect of the world as well as human reasoning. An increasing number of computational systems have been pushed into dealing with the temporal aspects of the applications they implement, aiming at fulfilling their goals and avoiding obsolescence. The notion of time (as well as its representation, properties, etc.) is a recurrent and very complex issue, subject to many discussions and approaches in different areas of human knowledge, such as Philosophy, Natural Language Processing (NLP), Medicine, Engineering, Computer Science, etc. (see e.g. [1, 2, 11, 19, 26, 44]) and, as such, approached from different perspectives and goals. Particularly in Artificial Intelligence (AI), the temporal aspect of situations and events is a decisive aspect to be considered when implementing intelligent systems [12]. Many of the existing AI systems can take advantage of temporal perception and representation to enhance their performance on knowledge representation, knowledge acquisition and inference tasks.

As pointed out by Vila in [46], several application areas that employ AI techniques can profit from efficient temporal treatments, particularly those related to: (i) *medical diagnosis and explanation* – information about when and in what sequence the symptoms have occurred and how long they have persisted are mandatory for prescribing the correct treatment; (ii) *planning* – in the development of a plan, not only the duration of its actions and tasks should be carefully considered but also how they can be temporally ordered considering their many possible interactions over time; (iii) *industrial process supervision* – the control of an industrial process involves many temporal aspects such as past states, variable values evolution over time, the point of time particular sub-processes start (or end) etc.; (iv) *natural language understanding* – mainly focusing on the verb tense in a sentence.

Due to computer developments and their increasing use in the last years, computational systems are being pushed into providing solutions which implement efficient ways of representing and reasoning in time-related tasks. The recent proposed approaches for learning from the Web, such as the never-ending learning – NEL – model [14], have pushed even further the pressure for devising efficient ways for implementing automatic learning systems capable of dealing with temporality. The inherent nature of the NEL model, which continuously extracts information from the Web and learns from it requires, as a pre-condition for its success, an efficient mechanism for dealing with the strong temporal aspects embedded in the Web.

The many proposals found in the literature for representing and dealing with time are evidence that the issue is still an open and ongoing research area, without much consensus about what is the best choice (among the proposals currently available) and what would be the more promising ones, to invest on. Several reviews present and discuss different formalisms for representing and reasoning about time, e.g. [15, 17, 23, 46]. Of particular relevance in representing temporal knowledge are the temporal logic languages, which belong to the family of description logics, characterized by high expressivity combined with good computational properties [10, 28]. The situation calculus [27] and the event calculus [24] are among the most popular logic formalisms for reasoning about actions.

An interesting and still influential formalism nowadays is the Temporal Interval Relations (TIR) [3]. In spite of receiving a few criticisms (e.g [18, 29, 30]), TIR is still a popular formalism, mainly due to its

simplicity and implementability characteristics; it has been extended and modified in a few ways. The main goal of this paper is to investigate how efficiently an automatic learning system can learn the basic temporal relations of TIR. Of particular interest is to explore the use of the Closed World Assumption (CWA), when providing the input to the learning system. The learning experiments described in the paper were conducted using the FOIL system [37, 38, 39], which uses Horn clauses as the language for representing and inducing knowledge.

The reminder of this paper is organized as follows: Section 2 describes the main concepts and the basic temporal relations that define TIR considering two different primitive concepts it could be based upon, namely *time interval* and *time point*. Section 3 introduces the main advantages of using inductive logic programming (ILP) and gives the main motivations for using ILP systems in automatic learning. Particularly provides the basic information about FOIL, the ILP system used for learning basic TIR temporal relations. Section 4 presents and discusses the results of the learning experiments conducted with FOIL taking into consideration time interval defined as: (a) a primitive concept (Subsection 4.1) and (b) a composite concept based on the primitive concept of time points (Subsection 4.2). Finally, in Section 5 a few directions of current and future work are presented.

2. Temporal Interval Relations (TIR) and the Theory of Time

Temporal Interval Relations (TIR) (or Allen's Interval Algebra) is a formalism proposed by Allen in [3, 4, 5] to represent and reason about temporal knowledge, based on binary interval relations. Since its proposal it has inspired several research works involving its use in different domains and applications as well as various extensions [16, 25, 32, 43, 45]. TIR is a simple linear model of time based on a unique primitive concept referred to as *time period*, and a primitive binary relation between two time periods named *meets*. A time period can be thought of as the time associated with some event occurring or some property holding in the world. Although the intuitive idea associated with the *meets* relation is very clear, its description and representation as a formal concept (aiming at its implementation) is not trivial, since it involves a previous definition of the adopted granularity for time, as well as the adopted representation for time period [33].

The appropriateness of using time points as opposed to time intervals, as the primitive concept for temporal representation and reasoning is still an open issue. Discussions involving the two different ways of representing the notion of time have been going on for a long time without much consensus, as can be evidenced in the many proposals of temporal theories available in the literature (e.g., [16, 18, 23]) claiming for richer structures of time, using either interval-based or point-based representations and, sometimes, both. Depending on the representation and the choice of primitive (intervals or points, or both), a variety of different temporal relations between times can be defined.

A few years after his initial proposal Allen considered the introduction of the concept of time points to represent time intervals as well [6, 20]. In spite of TIR taking temporal intervals as primitive, the formalism contemplates the possibility of, assuming a model consisting of a fully ordered set of points of time, representing an interval T as an ordered pair of points of time $< t_-, t_+ >$, satisfying the pre-condition given by $t_- < t_+$. Still solutions for dealing with open and closed intervals need to be included into this representational model.

In the initial proposal [3] Allen defined a set of 5 basic binary relations that can hold between two intervals: *meets, before, overlaps, during, equal*. The set was later extended by the inclusion of two

others: *starts* and *finishes* [4]. Reminding that if R is a binary relation, the converse (reversed) relation of R, written R^{-1}, is a relation such that $yR^{-1}x$ if and only if xRy. The 7 basic relations (pictorially shown in Figure 1) can be expanded into 13, if their reversed relations are considered (the reversed relation of *equal* is itself). Table 1 names the reversed relations associated to 6 basic temporal interval relations (equal excluded).

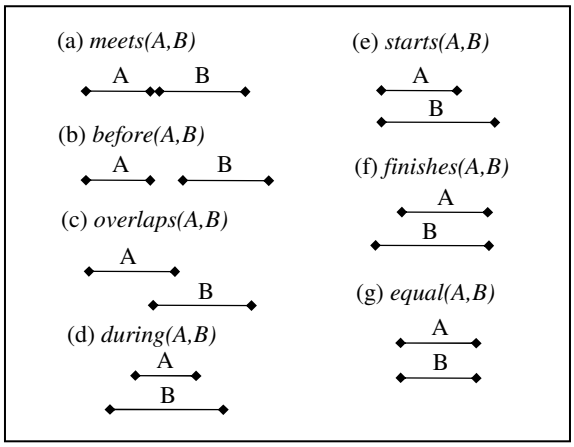

Figure 1. Pictorial representations of Allen's 7 basic temporal relations.

Table 1. Temporal basic interval relations (*equal* excluded), their corresponding reversed relations and the equivalence relation between each pair of them.

Interval Relation	*Reversed Relation*	*Equivalence*
meets	met-by	meets(A,B) $\equiv$ met-by(B,A)
before	after	before(A,B) $\equiv$ after(B,A)
overlaps	overlapped-by	overlaps(A,B) $\equiv$ overlapped-by(B,A)
starts	started-by	starts(A,B) $\equiv$ started-by(B,A)
finishes	finished-by	finishes(A,B) $\equiv$ finished-by(B,A)
during	contains	during(A,B) $\equiv$ contains(B,A)

As presented and discussed in [6], by taking into account all the TIR possible relations between two time invervals, they can constructively be redefined exclusively in terms of the *meets* relation. Table 2 shows the five basic temporal relations which are the focus of this work, represented in terms of the *meets* relation.

The redefinition subsidized the evolution of TIR into the proposal of a Temporal Theory (TT), initially described in [6] and later revised in [7, 8, 9], based on a non-empty class, I, of time intervals (assumed as primitive concept). TT's is axiomatized using a unique binary temporal relation, *meets*, that may happen between two time intervals, and considered a primitive concept. Intuitively two time invervals x and y meet if and only if x precedes y, yet (1) there is no time between x and y and (2) x and y do not overlap. The proposed axiomatization subsumes TIR and its axioms are described next, where i, j, k, l, m and n are logical variables restricted to the domain I.

Table 2. Allen's 5 basic binary temporal interval relations defined in terms of the primitive relation *meets* (*equal* and *meets* excluded), considering two time intervals i_1 and i_2.

Temporal relation	*Corresponding meets-based definition*
$before(i_1, i_2)$	$\exists i \in I \mid (meets(i_1, i) \wedge meets(i, i_2))$
$starts(i_1, i_2)$	$\exists i, j, k \in I \mid (meets(i, i_1) \wedge meets(i_1, j) \wedge meets(j, k) \wedge meets(i, i_2) \wedge meets(i_2, k))$
$finishes(i_1, i_2)$	$\exists i, j, k \in I \mid (meets(i, j) \wedge meets(j, i_1) \wedge meets(i_1, k) \wedge meets(i, i_2) \wedge meets(i_2, k))$
$overlaps(i_1, i_2)$	$\exists i, j, k, l, m \in I \mid (meets(i, i_1) \wedge meets(i_1, l) \wedge meets(l, m) \wedge meets(i, j) \wedge meets(j, i_2) \wedge meets(i_2, m) \wedge meets(j, k) \wedge meets(k, l))$
$during(i_1, i_2)$	$\exists i, j, k, l \in I \mid (meets(i, j) \wedge meets(j, i_1) \wedge meets(i_1, k) \wedge meets(k, l) \wedge meets(i, i_2) \wedge meets(i_2, l))$

1. (A_1) : *Uniqueness of meeting places* – states that the 'place' where two intervals meet is unique and closely associated with the intervals. If two intervals both meet a third, then any interval met by one must also be met by the other, formally stated as:

   $\forall i, j, k, l \in I \; (meets(i, j) \wedge meets(i, k) \wedge meets(l, j) \rightarrow meets(l, k))$

2. (A_2) : *Meeting places are totally ordered* – meaning that if interval i meets interval j and interval k meets interval l, then exactly one of the following holds: (a) i meets l; (b) $\exists$ interval m such that i meets m and m meets l and (c) $\exists$ interval n such that k meets n and n meets j, symbolically represented below, where $\oplus$ means exclusive or.

   $$\begin{array}{lll} \forall i, j, k, l \in I \; (meets(i, j) \wedge meets(k, l) & \rightarrow & meets(i, l) \\ & \oplus & \exists m \mid meets(i, m) \wedge meets(m, l) \\ & \oplus & \exists n \mid meets(k, n) \wedge meets(n, j)) \end{array}$$

3. (A_3) : *Time continuity* – there is no beginning or ending of time; every interval has at least one neighbouring interval preceding it and another succeding it, formally stated as:

   $\forall i \in I \; \exists j, k \in I \; (meets(j, i) \wedge meets(i, k))$

4. (A_4) : *Equal intervals* – if two intervals both meet the same interval and another interval meets both of them, then the two initial intervals are the same, formally stated as:

   $\forall i, j \in I \; \exists k, l \in I \; (meets(k, i) \wedge meets(k, j) \wedge meets(i, l) \wedge meets(j, l) \rightarrow i = j)$

5. (A_5) : *Composing intervals* – intervals can be composed producing a larger interval. For any two intervals that meet, there is a third interval which is the concatenation of the two, formally described by:

   $\forall i, j, k, l \in I\ (meets(i, j) \wedge meets(j, k) \wedge meets(k, l)) \rightarrow \exists m(meets(i, m) \wedge meets(m, l))$

3. Learning with FOIL (*First Order Inductive Learner*)

This section is organized into two subsections; the first gives the motivations for using Inductive Logic Programming (ILP) in automatic learning and presents some of its basic concepts. The second briefly describes the ILP system FOIL (First Order Inductive Learner), used in the experiments described in Section 4, for learning TIR basic temporal relations.

3.1. Inductive Logic Programming (ILP)

The most widely used model for symbolic learning is the *inductive learning from examples*, where the learning task consists of building a general concept description – *hypothesis* – from a given set of training examples – *positive examples* – and counterexamples – *negative examples* – of the concept. Machine learning systems employ formal languages for describing examples and concepts referred to as *example description language* ($\mathcal{L}_\mathcal{E}$) and *concept description language* ($\mathcal{L}_\mathcal{C}$) respectively. To describe the training examples many of the existing inductive learning algorithms use *attribute-based* languages. Concept description languages used for representing induced hypotheses, typically production rules or decision trees, can be treated as variants of attribute-based languages. As variants of the propositional calculus, both languages have only propositional representativeness power. Despite their success, attribute-based learning methods are not only constrained by the languages they employ but also by the very limited and inexpressive role the background knowledge ($\mathcal{K}$) (described by language $\mathcal{L}_\mathcal{K}$) plays in the learning process. Only concepts expressible in propositional logic are candidates to be learnt by systems using attribute-based languages. This strong restriction prevents representing structured objects as well as relations among objects or among their components. Thus relevant aspects of training examples, which somehow could characterize the concept to be learnt, cannot be represented.

To overcome some of the representational limitations imposed by attribute-based languages, learning using more powerful representations, such as variants of the first-order logic, have been considered. It is well known that logic programming is a traditional and sound research area, focused on the use of first-order logic to define computer programs. Logic programming can be broadly defined as the use of symbolic logic for the explicit representation of problems and their associated knowledge bases, together with the use of controlled logical inference for the effective solution of these problems. Logic programming is generally understood in more specific terms: the problem-representation language is a particular subset (Horn-clause form) of classical first-order predicate logic, and the problem-solving mechanism is a particular form (resolution) of classical first-order inference [42]. Logic programming in its narrow sense is based on Horn clause and it is often identified with Prolog.

Inductive Logic Programming – ILP – is a research area of Machine Learning (ML) focusing on the integration of techniques already available and established for logic programming in a framework of learning, aiming to induce first-order logic programs from examples, using background knowledge. In ILP the system's knowledge consists of examples and $\mathcal{K}$, represented as logic programs – the expressive-

ness of logic programs and the use of background knowledge have promoted ILP as a powerful inductive inference paradigm. The adoption of more powerful languages, however, gives rise to various difficulties that need to be overcome. Since the learning of logical definitions requires the exploration of very large space of hypotheses, restrictions should be imposed on both, the background and the hypotheses space in order to make learning a feasible task. Ways of confining the hypotheses space can be implemented *via* restricting the hypotheses and background knowledge description languages, for instance. By reducing the representative power of the languages employed, the search carried out by a learning system can be both: better controlled and limited. The languages employed by learning systems can be restricted in different ways as prescribed by the following definitions [21, 22, 47]. Usually background knowledge can be restricted by only allowing the use of ground background knowledge, as defined in Definition 3.1.

Definition 3.1. A background knowledge $\mathcal{K}$ is ground if and only if it is described by ground unit clauses (i.e., no variables involved) only.

Definition 3.2. A background knowledge $\mathcal{K}$ is of bounded arity if and only if the maximum arity of the predicates in $\mathcal{K}$ is bounded by some constant j.

The bounded-arity restriction is also used by Page and Frish [34], to prove the PAC-learnability of a special kind of concept language $\mathcal{L}_{\mathcal{C}}$ called *constrained clauses*.

Definition 3.3. A clause is constrained if and only if all variables in the body of the clause also occur in its head.

A common restriction applied to both, $\mathcal{L}_{\mathcal{K}}$ and $\mathcal{L}_{\mathcal{C}}$, is the restriction to function-free clauses.

Definition 3.4. A clause is function-free if and only it has no function symbols; i.e. all of its arguments are either variables or constants (function symbols of arity 0).

Two common restrictions shared by some ILP systems are implemented by only allowing ground $\mathcal{K}$ and ground unit clauses as examples; both are also adopted by FOIL, briefly described next.

3.2. A Brief Description of FOIL

FOIL *(First Order Inductive Learner)*, proposed and developed by Quinlan [37, 39], is a non-incremental ILP system which induces function-free Horn clause representations of relations; one target relation is considered at a time. FOIL is subsidized by several ideas that were proved effective in attribute-based learning systems, which were customized to the restricted first-order languages adopted.

Within the FOIL framework the considered task is to learn a function-free Horn clause definition of a target relation in terms of the target and other given relations, from extensional definitions of relations. The extensional definition of the target relation is given as positive examples (tuples for which the relation holds) and negative examples (tuples for which it does not hold). If all the positive examples are known, the negative examples can be generated under the Closed World Assumption (CWA) – an aspect which is of particular interest in the experiments described in Section 4. The 'other relations' referred before constitute the so-called background knowledge ($\mathcal{K}$), also known as domain theory – a set of presumably known relations which are fed to the system in their extensional form, i.e., as ground facts, and can be used by the system for inferring the final expression of the target relation. FOIL learns a description of

the target relation represented in terms of the target and, eventually, of the other relations given as $\mathcal{K}$. So, given the positive and negative examples that define the relation to be learnt and examples of each relation included in $\mathcal{K}$, FOIL's task is to learn a function-free Horn clause definition of the given target relation. When learning the definition of a relation, FOIL uses a covering approach; it repeatedly learns a clause and then removes the positive examples that are covered by it from the set of positive examples to be covered. Once the definition is complete (all the positive examples are covered), FOIL prunes out redundant clauses. For learning a clause FOIL employs an information-based heuristic (similarly to the one used by ID3 [36]) for assessing the contribution a literal has, as the next component to be included in the right-hand side of the clause being constructed.

As Quinlan mentions in one of his papers [38], *FOIL is based on ideas that have proved effective in attribute-value learning systems, but extends them to a first-order formalism.* Later on in the same paper, Quinlan claims that *FOIL has no notion of proof – the validity of a clause is investigated by looking for specific counter-examples in the training set. Moreover, the system looks for maximally general descriptions that allow positive examples to be differentiated from negative ones, rather than maximally specific descriptions that express all shared properties of subsets of positive examples. FOIL cannot learn anything from $\oplus$-tuples alone and needs either explicit $\ominus$-tuples or the closed-world assumption.*

Referring to FOIL's approach, Cameron-Jones and Quinlan in [41] comment: *This and similar approaches in ILP are amongst those most closely related to the traditional (zeroth order) inductive literature, in contrast to those such as Muggleton and Feng's GOLEM [31] (which draws more strongly upon ideas from logic programming).* As pointed out in [40] proof-based algorithms are relatively slow and most of their execution time is consumed in theorem-proving mode, so that they are able to analyze only small training sets (compared to those not proof-based methods), such as FOIL and GOLEM; GOLEM uses Plotkin's relative least general generalization (rlgg) [35] to form clauses while FOIL uses a divide-and-cover approach adapted from zeroth-order learning.

Input data to FOIL follows the required syntax where the header of a relation, other than the target relation, begins with an '*' and consists of the relation name, argument types and optional keys, followed by a sequence of lines containing examples. Each example consists of constants separated by commas (argument separator) and must appear in a single line. Notice that the restriction of having each example in a single line is not always followed in the tables presented in Section 4 due to space limitations. A line with the character '.' indicates the end of the extensional description of the corresponding relation (and has not been included in the tables, for simplification reasons). When referring to predicates, their arity (i.e., number of arguments) will be omitted for the sake of simplicity, when that does not introduce ambiguity. For the experiments described in this paper, the version 5.1 of FOIL was used, downloaded from http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/learning/systems/foil/foil5. The FOIL system runs on a Virtual Machine with Linux Ubuntu version 10 with gcc compiler. For more details on FOIL see also [40].

4. Learning TIR Relations Using FOIL

This section describes the necessary information – positive examples and background knowledge ($\mathcal{K}$) – when using FOIL to learn each of Allen's basic temporal relations, considering two representations: (1) time interval as a primitive concept (Subsection 4.1) and (2) time interval as a composite concept defined in terms of the primitive concept *time point* (Subsection 4.2). The goal of the two subsections is to

investigate (1) the influence of the representation adopted for time interval when learning TIR relations, (2) the use of the CWA in both situations and (3) how the background knowledge ($\mathcal{K}$) and the set of positive examples (PE) influence the relation induced expressions. The experiments intended also to determine a minimal data domain (where positive examples and $\mathcal{K}$ would be extracted from) that would still provide FOIL with enough information to induce the expected expression (for each relation). For the experiments described in both subsections, only five out of the seven basic relations are considered, since *meets* is the primitive relation used for expressing the others and *equal* has no representation in terms of *meets*. The automatic learning of each of basic relation in terms of *meets* is considered successful if the resulting expression agrees with the one proposed by Allen and described in Table 2.

Several of the following tables adopt the notation: ID – identification number of the experiment, PE – positive examples of the relation to be learnt, |NE| – number of negative examples automatically generated under the CWA and IC – relation expression induced by FOIL.

4.1. Learning TIR Relations Based on Time Interval

This subsection describes the necessary information (positive examples and background knowledge ($\mathcal{K}$)) when using FOIL to learn each of Allen's basic temporal relations in terms of the primitive relation *meets*, as described in the second column of Table 2, assuming *time interval* as primitive. Table 3 contains the extensional description of the argument type required by FOIL, specifying objects representing time intervals. The number of objects varies depending on the learning situation; particularly depends on the number of predicates that should be part of the body of induced clauses, to consistently represent the relation. The given background knowledge ($\mathcal{K}$) depends on the argument types used for learning each relation as well as on the ground facts satisfying relation *meets*; $|\mathcal{K}|$ gives the number of background instantiations of the unique relation (*meets*) that defines $\mathcal{K}$. For instance, for learning relation *before* (Table 4), the eight facts that satisfy the *meets* relation have been given to FOIL as $\mathcal{K}$, namely: meets(a,c), meets(a,d), meets(b,f), meets(b,g), meets(c,f), meets(c,g), meets(d,e), meets(g,e). In experiment #3 of the same table the eight facts have also been given as PE of relation *before*. The identification of the number of required positive examples that allowed FOIL to induce the appropriate expression for each relation was determined by a recurrent procedure, starting with a reasonable number of PE and, at each iteration, decreasing the number by one until success was reached. Figure 2 shows the situations that subsidized the input data to FOIL.

Table 3. Type definition required by FOIL used in the experiments for learning temporal relations using *time interval*, where |Intervals| represents the number of temporal intervals used.

Relation	Argument type (representing time intervals)	|Intervals|	$\|\mathcal{K}\|$
before(X,X)	X: a,b,c,d,e,f,g	7	8
starts(X,X) finishes(X,X)	X: a,b,c,d,e,f,g,h,i,j,k,l	12	19
overlaps(X,X) during(X,X)	X: a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p	16	28

Table 4 shows the results of three experiments for learning relation *before*, based on the data domain pictured in Figure 2. In experiment #1 FOIL was fed with five positive examples of relation *before* and

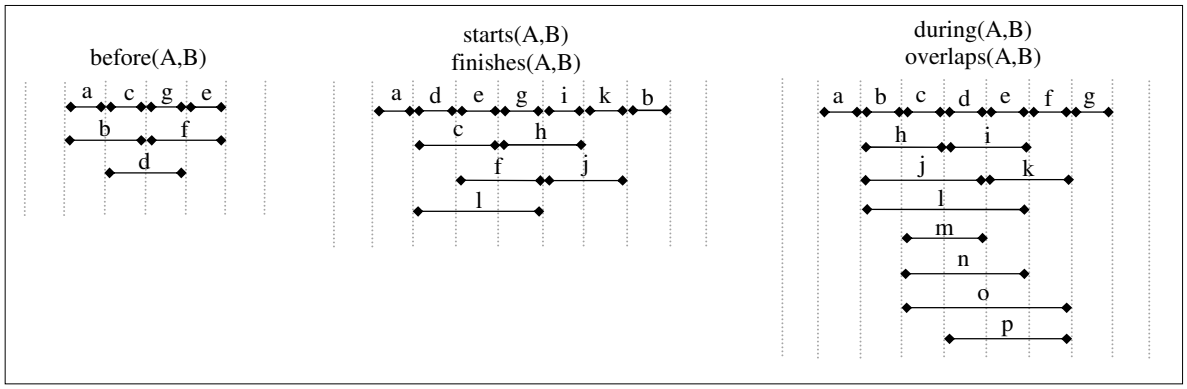

Figure 2. Pictorial examples used for learning the 5 basic temporal relations based on primitive relation *meets*.

Table 4. Learning *before* using time inverval.

ID	PE	|NE|	IC
#1	a,e a,f a,g b,e c,e	44	before(A,B):- meets(A,C), meets(C,B).
#2	a,e a,f a,g b,e	45	before(A,B):- FOIL does not induce any expression, and prompts: *** Warning: the following definition *** does not cover 4 tuples in the relation
#3	a,e a,f a,g b,e c,e **a,c a,d b,f b,g c,f c,g d,e g,e**	36	before(A,B):- meets(A,B). before(A,B):- meets(A,C), before(C,B).

eight instantiations of relation *meets* as $\mathcal{K}$; the system successfully inferred the expected representation of the target relation. In #2 one positive example was removed from the set of positive examples given in #1 (increasing the number of negative examples by one). The system was not able to induce any expression and issued a warning message.

As previously discussed, relation *before* was initially proposed as described in Table 2, i.e., an interval is before another if there is a third interval 'connecting' them. However, in [9], Allen presents a modified definition for *before*, taking into account (i) if two intervals satisfy the *meets* relation then they also satisfy the *before* relation and (ii) the recursive aspect that could be embedded in the description of relation *before*. The experiment #3 aimed at inducing the modified definition of *before*. As in #1, the eight ground instantiations of relation *meets* were given to the system as $\mathcal{K}$ and as PE, the one from experiment #1 enlarged with the addition of the eight ground facts satisfying relation *meets* (bold-faced in #3).

Table 5 shows two successful experiments, where #1 refers to the learning of relation *starts* and #2 to the learning of relation *finishes*. Input data to both experiments was extracted from the middle diagram

shown in Figure 2. For both experiments the number of ground instantiations of relation *meets*, given as $\mathcal{K}$, was 19.

Table 5. Learning *starts* and *finishes* using time interval.

ID	PE	|NE|	IC
#1	c,l d,c d,l e,f g,h i,j	138	starts(A,B):- meets(A,C), meets(C,D), meets(B,D), meets(E,A), meets(E,B).
#2	e,c f,l g,f g,l i,h k,j	138	finishes(A,B):- meets(C,A), meets(A,D), meets(B,D), meets(E,C), meets(E,B).

The successful results shown in Table 6, when learning *overlaps* and *during* were obtained in a similar fashion as those from the previous relations. The main difference, however, is related to the number of positive examples required. For both relations a greater number of time intervals was needed (i.e., 16), as shows diagram on the right in Figure 2, to provide enough data for establishing appropriate positive examples of both relations, particularly for the *overlaps* relation that required 15 positive examples for the induction of its correct expression. For both experiments described in Table 6 the number of ground instantiations of relation *meets*, given as $\mathcal{K}$, was 28. Learning results described in tables 4 to 6 can be considered evidence that *time inverval*, as TIR primitive concept, is a suitable assumption to both, (i) TIR axiomatization and (ii) representing the basic temporal relations in terms of the primitive relation *meets*.

Table 6. Learning *overlaps* and *during* using time inverval.

ID	PE	|NE|	IC
#1	h,m h,n h,o i,k j,i j,n j,o j,p l,k l,o l,p m,i m,p n,k n,p	241	overlaps(A,B):- meets(A,C), meets(D,B), meets(D,E), meets(E,C), meets(C,F), meets(G,D), meets(B,F), meets(G,A).
#2	c,j c,l d,l d,n d,o e,o e,p i,o m,l	247	during(A,B):- meets(A,C), meets(D,A), meets(C,E), meets(F,D), meets(B,E), meets(F,B).

In a real-world domain, considering periods of time defined by days of week, Table 7 and Table 8 describe the learning of relations *before* and *starts*, respectively; this situation has been considered before (Figure 2). For learning *before* the argument type definition used was: X: monday, tuesday, wed, thursday, monday_tuesday, tuesday_wed, wed_thursday. The positive examples and the background knowledge in Table 7 allowed FOIL to induce (under the CWA) the expression of relation *before* as the two clauses: (1) *before(A,B):- meets(A,B).* and (2) *before(A,B):- meets(A,C), before(C,B).*

For learning *starts* the argument type definition used was: X: monday, tuesday, wed, thursday, friday, saturday, sunday, tuesday_wed, tuesday_thursday, wed_thursday, thursday_friday, friday_saturday. The positive examples and the background knowledge in Table 8 allowed FOIL to induce (under the CWA) the relation *starts* as the clause: *starts(A,B):- meets(A,C), meets(C,D), meets(B,D),meets(E,A),meets(E,B).*

Table 7. Learning *before* using time inverval in a real-world domain where periods are defined by days of week and knowledge domain ($\mathcal{K}$) given by eight ground facts of relation *meets*.

PE		$\mathcal{K}$
monday,thursday	monday_tuesday,thursday	monday,tuesday
monday,wed	monday,tuesday	monday,tuesday_wed
tuesday,thursday	monday_tuesday,wed_thursday	monday_tuesday,wed
monday,tuesday_wed	tuesday,wed_thursday	monday_tuesday,wed_thursday
monday_tuesday,wed	tuesday_wed,thursday	tuesday,wed
tuesday,wed		tuesday,wed_thursday
wed,thursday		tuesday_wed,thursday
monday,wed_thursday		wed,thursday

Table 8. Learning *starts* using time inverval in a real-world domain where periods are defined by days of week and knowledge domain ($\mathcal{K}$) given by nineteen ground facts of relation *meets*.

PE	$\mathcal{K}$	
tuesday_wed,tuesday_thursday	monday,tuesday_wed	saturday,sunday
tuesday,tuesday_wed	monday,tuesday	tuesday_wed,thursday_friday
tuesday,tuesday_thursday	monday,tuesday_thursday	tuesday_wed,thursday
wed, wed_thursday	tuesday,wed	thursday_friday,saturday
thursday,thursday_friday	tuesday,wed_thursday	wed_thursday,friday
friday,friday_saturday	wed,thursday	wed_thursday,friday_saturday
	wed,thursday_friday	friday_saturday,sunday
	thursday,friday	tuesday_thursday,friday
	thursday,friday_saturday	tuesday_thursday,friday_saturday
	friday,saturday	

4.2. Learning TIR Relations Based on Time Points

As mentioned before, assuming a model consisting of a fully ordered set of points of time, TIR contemplates the possibility of representing an interval T as an ordered pair of points of time $< t_-, t_+ >$, satisfying the pre-condition $t_- < t_+$. This subsection reports the experiments with FOIL for learning Allen's basic temporal relations, where the concept of time interval is assumed to be a composite concept, based on the primitive notion of time point. Since each time interval is represented by two time points, relations between two time intervals have arity 4 – the first two arguments represent the ending points of the first interval and the last two, of the second. An interval is a valid interval if its ending points satisfy the previous stated pre-condition. For the experiments described in this subsection all time intervals fed to FOIL were valid. The condition that qualifies a point-based interval as valid, however, can be learnt by FOIL, as shown in Table 9. In the table, based on a discrete time line defined by four time points (i.e., 1,2,3,4), considering the exhaustive list of possible positive examples (i.e., six positive examples) and $\mathcal{K}$ represented by all the ground instantiations of relation *lessthan* taking into account the discrete

domain, FOIL induces, under the CWA, the appropriate expression that represents a valid point-based time interval, represented by relation *interval*.

Table 9. Learning the definition of valid point-based time interval, given by relation *interval*, whose arguments are the two time points defining it.

PE	$\mathcal{K}$	IC
interval(X,X)	*lessthan(X,X)	interval(A,B):-
1,2 2,3	1,2 2,3	lessthan(A,B).
1,3 2,4	1,3 2,4	
1,4 3,4	1,4 3,4	

In the following tables a point-based binary relation generically named *relation* is represented as *relation(A,B,C,D)*, where *A* and *B* are the ending points of the first interval and *C* and *D* are the ending points of the second interval. The same notation used in Subsection 4.1 has been adopted, i.e.: ID – identification number of the experiment, PE – positive examples of the relation to be learnt, |NE| – number of negative examples automatically generated under CWA and IC – relation expression induced by FOIL. The number of instantiations of relation *meets* that define the background knowledge fed to FOIL is represented by $|\mathcal{K}|$.

As pointed out before, FOIL requires the definition of the type of data the examples represent. For all experiments, this requirement has been acomplished by having, at the beginning of the input file to FOIL, the type definition of relation arguments, as individually shown, *per* relation, in Table 10.

Table 10. Type definition required by FOIL, used in the experiments for learning temporal relations using *point-based interval*, where |Intervals| represents the number of temporal intervals used.

Relation	Type definition	|Intervals|	$\|\mathcal{K}\|$
before(X,X,X,X)	X: 0,1,2,3,4	7	8
starts(X,X,X,X)	X: 0,1,2,3,4,5,6	10	14
finishes(X,X,X,X)	X: 0,1,2,3,4,5,6	10	14
overlaps(X,X,X,X)	X: 0,1,2,3,4,5,6	13	22
during(X,X,X,X)	X: 0,1,2,3,4,5,6	11	16

Figure 3 shows five pictorial situations which provided data for learning the five basic temporal relations, as described in Table 11, Table 12 and Table 13. It is worth reminding again that, as in Subsection 4.1, several learning experiments were initially conducted, trying to identify a minimal domain configuration that would lead FOIL to infer the appropriate meets-based relation expressions under CWA. Figure 3 pictorially represents the identified configurations, for each basic relation.

Table 11 describes two experiments for learning relation *before*. Experiment #1 describes the data used for inferring the expected (in the sense of information given in Table 2) expression of relation *before*. Experiment #2 reproduces the same situation described in Subsection 4.1 (Table 4, #3), this time, though, using a point-based interval representation. The induced concept is represented by two clauses: the first stating that four time points satisfy *before* relation if they satisfy relation *meets* and the second, defined as a conjunction of *meets* and a recursive call to *before*. An exhaustive list of instantiations of

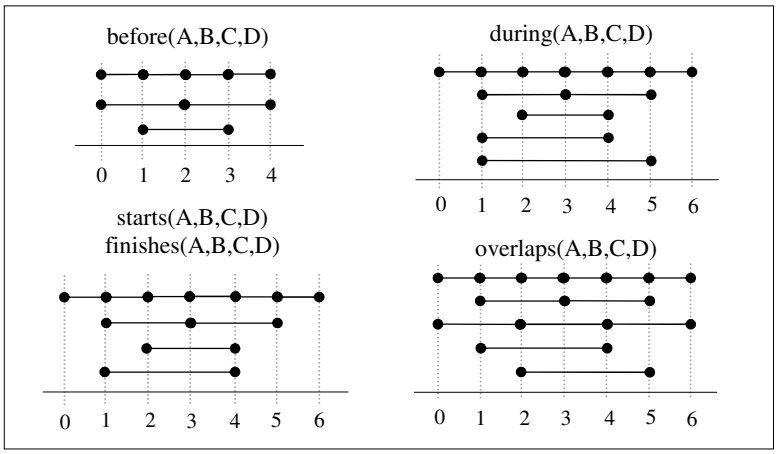

Figure 3. Pictorial examples for learning basic temporal relations in terms of relation *meets*, using point-based temporal interval.

relation *meets* have been given to FOIL as positive examples of relation *before* (shown bold-faced in #2 of Table 11).

Table 11. Learning *before* using point-based temporal interval.

ID	PE	|NE|	IC
#1	0,1,2,3 0,1,2,4 0,1,3,4 0,2,3,4 1,2,3,4	620	before(A,B,C,D):- meets(A,B,B,C), meets(B,C,C,D).
#2	0,1,2,3 0,1,2,4 0,1,3,4 0,2,3,4 1,2,3,4 **0,1,1,2 0,1,1,3 0,2,2,3 0,2,2,4 1,2,2,3 1,2,2,4 1,3,3,4 2,3,3,4**	612	before(A,B,C,D):- meets(A,B,C,D). before(A,B,C,D):- meets(A,B,B,C), before(B,C,C,D).

TIR relations, as stated in Table 2, have embedded in their meets-based representation, the 'concept' of time-continuity. When learning *overlaps*, although the four pairs of point-based intervals: 0,2,1,3; 0,2,1,4; 2,5,4,6 and 3,5,4,6 could qualify as positive examples of *overlaps*, they have not been included as PEs due to what has been called here 'lack of continuity representativeness'. Their inclusion as positive examples would provoke the side-affect of biasing the learning process towards the induction of an expression lacking continuity representativeness. Also, their inclusion as PEs would require an expansion of the domain, so to accomodate the point-based intervals used in their definition, aiming to support the representation of 'continuity' of the time line; that on its turn, would provoke again a similar situation of lack of continuity, requiring another expansion and so on.

The main idea underneath the experiments was to show that TIR basic binary temporal relations can be learned based on the primitive relation *meets*, as stated in Table 2 (Section 2), using point-based temporal intervals representation as well. The experiments also aimed at determining the smallest number of positive examples which would allow FOIL to induce the appropriate expression for each of the five relations, under the CWA.

Table 12. Learning *starts* and *finishes* using point-based temporal interval.

ID	PE			|NE|	IC
#1	1,2,1,3	1,2,1,4	1,3,1,4	2396	starts(A,B,C,D):-
	2,3,2,4	3,4,3,5			meets(E,A,C,B), meets(B,D,D,F), meets(A,D,D,F).
#2	2,3,1,3	2,4,1,4	3,4,1,4	2396	finishes(A,B,C,D):-
	3,4,2,4	4,5,3,5			meets(A,B,D,E), meets(F,C,C,A), meets(C,B,B,E).

Table 13. Learning *overlaps* and *during* using point-based temporal interval.

ID	PE			|NE|	IC
#1	1,3,2,4	1,3,2,5	2,4,3,5	2396	overlaps(A,B,C,D):-
	1,4,3,5	1,4,2,5			meets(A,B,B,D), meets(A,C,C,B),
					meets(C,D,D,E), meets(F,A,A,B).
#2	2,3,1,4	2,3,1,5	2,4,1,5	2397	during(A,B,C,D):-
	3,4,1,5				meets(A,B,B,D), meets(E,C,C,A), meets(C,D,D,F).

The expressions defining each TIR basic temporal relation are conjunctions of *meets* relations (see Table 2). A comparison between the number of *meets* relations that define each temporal relation and the number of *meets* relations in the body of FOIL induced clauses described in this subsection, is shown in Table 14. Differently from what happened with experiments described in Subsection 4.1 where the induced relation expressions coincide with their definition, this was not the case for the learning experiments using point-based intervals, described in this subsection.

Table 14 is clear evidence that, when compared with their original counterparts (Table 2), the induced relation expressions using point-based interval are conjunctions of a smaller number of *meets* relation. The binary point-based relations are represented by four arguments; the 'concept' each of them represent is not only embedded in the representation but also represented by the arguments. Except for the *before* relation, the other four have representations which slightly differ from their originally proposed counterparts. The concepts they represent, however, are still the same given by TIR original relations. This can be visualized in Figure 4, where each point-based induced expression is 'mapped' into its original expression, as presented in Table 2.

Table 14. Comparing the number of *meets* relations needed for defining temporal relations.

Relation	Definition - Table 2	Learning - Tables 11, 12 and 13
before	2	2
starts	5	3
finishes	5	3
overlaps	8	4
during	6	3

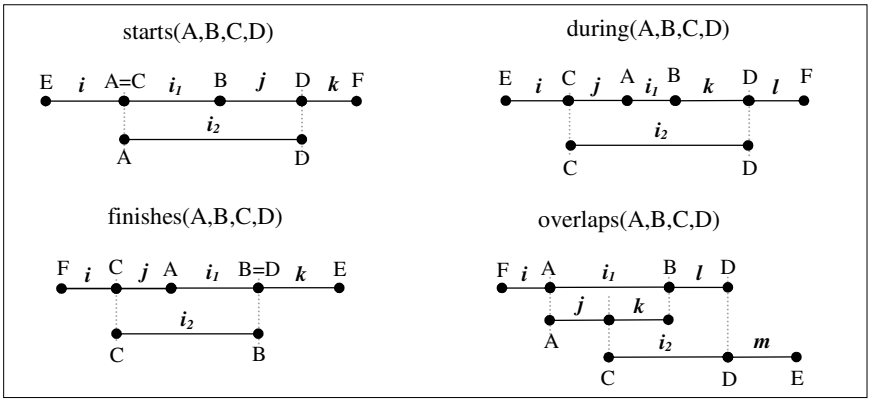

Figure 4. Pictorial representation of the induced expressions, taking into account their original representations given in Table 2.

5. Conclusions and Further Work

This paper investigates how efficiently an automatic learning system can learn formal expressions representing temporal knowledge. The formalism of choice for representing temporal knowledge was Allen's TIR. The choice was motivated mainly by TIR's simplicity and understandability; also its set of basic binary temporal relations seems to represent well some of usual temporal relations commonly employed. As far as time interval representation is concerned, it has become quite obvious, based on experiment results, that Allen's temporal formalism is much more suitable for time interval considered as primitive concept, as initially proposed, than as a composite concept defined by time-points. Both representations, though, can be used; particularly the slightly modified expressions learnt when using point-based time intervals, represent the same concept of their counterparts, using time interval.

As always happens with automatic learning, induced concepts reflect the initial knowledge available to the learning system. The learning experiments described in this paper clearly corroborate the influence of the given information (positive examples and background knowledge) on the induced temporal expressions. Although not explicitly shown (except for experiment #2 in Table 4), the influence of the given positive examples has been determinant for inducing the appropriate expression of temporal concepts.

In relation to interpretability (in the sense of human comprehension), for those familiar with Allen's TIR, the learning results described in Section 4.1 offer no extra burden to human comprehension, since they coincide with the original representation given in TIR. When point-based time intervals are used, however, the induced expressions, although representing the same relation, requires efforts into 'translating' it into a more familiar representation, such as shown in Figure 4.

One of the goals of the experiments conducted and described in this paper was also to determine the feasibility of implementing a temporal learner system, based on data collected from the Web, in a continuous way, in a never-ending learning fashion. The conducted experiments explored the use of the CWA having in mind that, by assuming CWA, the intended never-ending learning task could be simplified (considering that the other existing option would be to provide the negative examples as well). The experiences learnt from the investigation of temporal learning reported in this paper, however, are far from being portable and are hardly able to be customized to a dynamic and constantly changing

environment as the Web. Also, good results were obtained due to close supervision when defining the information given to the learning system – this type of supervision is not viable in a never-ending learning environment. Next it is intended to modify the formalism by removing predicates which reflect the idea of 'time continuity'.

Acknowledgements – The authors would like to thank the reviewers for their comments and suggestions on an early draft of the paper.

References

[1] Adlassnig, K.-P., Combi, C., Das, A. K., Keravnou, E. T., Pozzi, G.: Temporal representation and reasoning in medicine: research directions and challenges, *Artificial Intelligence in Medicine*, vol. 38, 2006, 101–113.

[2] Aiello, M., Monz, C., Todoran, L., Worring, M.: Document understanding for a broad class of documents, *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 5, no. 1, 2002, 1–16.

[3] Allen, J. F.: An interval-based representation of temporal knowledge, in: *Proc. of The 7th International Joint Conference on Artificial Intelligence (IJCAI 81)*, Vancouver: Morgan Kaufmann, 1981, 221–226.

[4] Allen, J. F.: Maintaining knowledge about temporal intervals, *Communications of the ACM*, vol. 26, no. 11, 1983, 832–843.

[5] Allen, J. F.: Towards a general theory of action and time, *Artificial Intelligence*, vol. 23, 1984, 123–154.

[6] Allen, J. F., Hayes, P. J.: A common-sense theory of time, in: *Proc. of The 9th International Joint Conference on Artificial Intelligence*, Los Angeles: Morgan Kaufmann, 1985, 528–531.

[7] Allen, J. F., Hayes, P. J.: Moments and points in an interval-based temporal logic, *Computational Intelligence*, vol. 5, no. 3, 1989, 225–238.

[8] Allen, J. F.: Time and time again: the many ways to represent time, *International Journal of Intelligent Systems*, vol. 6, no. 4, 1991, 341–355.

[9] Allen, J.F., Ferguson, G.: *Actions and events in interval temporal logic*, Technical Report, The University of Rochester, 1994, 58 pgs.

[10] Artale, A., Franconi, E.: A temporal description logic for reasoning about actions and plans, *Journal of Artificial Intelligence Research*, vol. 9, 1998, 463–506.

[11] Bellazi, R., Larizza, C., Magni, P., Bellazi, R.: Temporal data mining for the quality assessment of hemodialysis services, *Artificial Intelligence in Medicine*, vol. 34, 2005, 25–39.

[12] Bouzid, M., Combi, C., Fisher, M., Ligozat, G.: Guest editorial: temporal representation and reasoning, *Annals of Mathematics and Artificial Intelligence*, vol. 46, no. 3, 2006, 231–234.

[13] Cameron-Jones, R. M., Quinlan, J. R.: Efficient top-down induction of logic programs, *ACM SIGART Bulletin*, vol. 5, no. 1, 1994, 33–42.

[14] Carlson, A.: *Coupled semi-supervised learning*, PhD. Thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, USA, 2010.

[15] Chittaro, L., Montanari,A.: Temporal representation and reasoning in artificial intelligence: issues and approaches, *Annals of Mathematics and Artificial Intelligence*, vol. 28, 2000, 47–106.

[16] Freksa, C.: Temporal reasoning based on semi-intervals, *Artificial Intelligence*, vol. 54, nos. 1-2, 1992, 199–227.

[17] Furia, C., Mandrioli, D., Morzenti, A., Rossi, M.: Modeling time in computing: a taxonomy and a comparative survey, *ACM Computing Surveys*, vol. 42, no. 2, 2010, 1–59.

[18] Galton, A.: A critical examination of Allen's theory of action and time, *Artificial Intelligence*, vol. 42, nos. 2-3, 1990, 159–188.

[19] Gamper, J., Nejdl, W.: Abstract temporal diagnosis in medical domains, *Artificial Intelligence in Medicine*, vol. 10, no. 3, 1997, 209–234.

[20] Hayes, P. J., Allen, J. F.: Short time periods, in: *Proc. of The 10th International Joint Conference on Artificial Intelligence (IJCAI 87)*, Milan: Morgan Kaufmann, 1987, 981–983.

[21] Kietz, J-U.: Some lower bounds for the computational complexity of inductive logic programming, *Lecture Notes in Computer Science*, vol. 667, P. B. Bradzil (ed.), Springer-Verlag, 1993, 115–123.

[22] Kietz, J-U., Dzeroski, S.: Inductive logic programming and learnability, *SIGART Bulletin*, vol. 5, no. 1, 1994, 22–32.

[23] Knight, B., Ma, J.: Time representation: a taxonomy of temporal models, *Artificial Intelligence Review*, vol. 7, no. 6, 1993, 401–419.

[24] Kowalski, R., Sergot, M.: A logic-based calculus of events, *New Generation Computing*, vol. 4, no. 1, 1986, 67–95.

[25] Ma, J., Knight, B.: A general temporal theory, *The Computer Journal*, vol. 37, no. 2, 1994, 114–123.

[26] Mani, I., Pustejovsky, J., Sundheim, B.: *Introduction to the special issue on temporal information processing*, vol. 3, no. 1, 2004, 1–10.

[27] McCarthy, J., Hayes, P. J.: Some philosophical problems from the standpoint of artificial intelligence, *Machine Intelligence*, vol. 4, 1969, 463–502.

[28] McDermott, D.: A temporal logic for reasoning about processes and plans, *Cognitive Science*, vol. 6, no. 2, 1982, 101–155.

[29] Morchen, F.: Algorithms for time series knowledge mining, in: *Proc. of The 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Philadelphia:ACM Press, 2006, 668–673.

[30] Morchen, F.: A better tool than Allen's relations for expressing temporal knowledge in interval data, in: *Proc. of The 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Philadelphia: ACM Press, 2006, 25–34.

[31] Muggleton, S., Feng, C.: Efficient induction of logic programs, in: *Proc. of The First Conference on Algorithmic Learning Theory*, Tokyo, Japan.

[32] Nebel, B., Burckert, H. J.: Reasoning about temporal relations: a maximal tractable subclass of Allen's interval algebra, *Journal of the ACM*, vol. 42, no. 1, 1995, 43–66.

[33] Nicoletti, M. C.; Lisboa, F. O. S. S.; Hruschka, E. R.: Learning temporal interval relations using inductive logic programming, *Communications in Computer and Information Science*, vol. 165, 2011, 90–104.

[34] Page, C. D., Frish, A. M.: Generalisation and learnability: a study in constrained atoms, in: S. H. Muggleton (ed.), *Inductive Logic Programming*, Academic Press, 1992, 29–61.

[35] Plotkin,G. D.: *Automatic Methods of Inductive Inference*, Ph. D. thesis, Edinburgh University, 1971.

[36] Quinlan, J. R.: Induction of decision trees, *Machine Learning*, vol. 1, 1986, 81–106.

[37] Quinlan, J. R.: Learning from relational data, in: *Proc. of The 4th Australian Joint Conference on Artificial Intelligence*, World Scientific, 1990, 38–47.

[38] Quinlan, J. R.: Learning logical definitions from relations, *Machine Learning*, vol. 5, 1990, 239–266.

[39] Quinlan, J. R.; Cameron-Jones, R. M.: *FOIL - an overview*, FOIL Version 5.0, 1993.

[40] Quinlan, J. R.; Cameron-Jones, R. M.: FOIL: A midterm report, in: *Proc. of The European Conference on Machine Learning (ECML)*, Springer Verlag: Bled, Slovenia, vol. 667, 1993, 3–20.

[41] Cameron-Jones, R. M., Quinlan, J. R.: Efficient top-down induction of logic programs, *ACM SIGART Bulletin*, vol. 5, no. 1, 1994, 33–42.

[42] Robinson, J.: A machine-oriented logic based on the resolution principle, *Journal of the ACM*, vol. 12, no. 1, 1965, 23–41.

[43] Roddick, J. F., Mooney, C. H.: Linear temporal sequences and their interpretation using midpoint relationships, *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 1, 2005, 133–135.

[44] Sanampudi, S. K., Kumari, G. V.: Temporal reasoning in natural language processing: a survey, *International Journal of Computer Applications*, vol. 1, no. 4, 2010, 53–57.

[45] Schockaert, S., De Cock, M., Kerre, E. E.: Fuzzifying Allen's temporal interval relations, *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 2, 2008, 517–533.

[46] Vila, L.: A survey on temporal reasoning in artificial intelligence, *AI Communications*, vol. 7, 1994, 4–28.

[47] Wrobel, S.: Higher-order concepts in a tractable knowledge representation, in: *Proc. of The 11th German Workshop on Artificial Intelligence – GWAI-87*, Informatik-Fachberichte no. 152, Springer-Verlag, 1987, 129–138.