LOGIC, PHILOSOPHY AND COMPUTER SCIENCE

Series: STUDIES IN LOGIC, GRAMMAR AND RHETORIC 22(35)

Under the Auspices of the Polish Association
for Logic and Philosophy of Science

LOGIC, PHILOSOPHY AND COMPUTER SCIENCE

edited by
Dariusz Surowik

University of Białystok
Białystok 2010

Refereed by Edward Oczeretko, Kazimierz Trzęsicki, Robert Milewski

Series: STUDIES IN LOGIC, GRAMMAR AND RHETORIC 22(35)
http://logika.uwb.edu.pl/studies/

Edited by Halina Święczkowska
University of Białystok, Faculty of Law, Section of Semiotics

in collaboration with Kazimierz Trzęsicki
University of Białystok, Faculty of History and Sociology
Chair of Logic, Informatics and Philosophy of Science – logika@uwb.edu.pl

Guest-Editor:
Dariusz Surowik
University of Białystok

Editorial Assistants:
Katarzyna Doliwa
University of Białystok
Dariusz Surowik
University of Białystok

Editorial Advisory Board:
Jerzy Kopania, University of Białystok
Grzegorz Malinowski, University of Łódź
Witold Marciszewski (Chairman), University of Białystok
Roman Murawski, Adam Mickiewicz University, Poznań
Mieczysław Omyła, Warsaw University
Katarzyna Paprzycka, Warsaw School of Social Psychology
Jerzy Pogonowski, Adam Mickiewicz University, Poznań
Andrew Schumann, Belarusian State University (Minsk)
Jan Woleński, Jagiellonian University, Cracow
Ryszard Wójcicki, Polish Academy of Sciences, Wrocław

© Copyright by Uniwersytet w Białymstoku, Białystok 2010

Cover design: Krzysztof Tur
Type-setting: Stanisław Żukowski

ISBN 978–83–7431–273–8
ISSN 0860–150X

WYDAWNICTWO UNIWERSYTETU W BIAŁYMSTOKU
15-097 Białystok, ul. Marii Skłodowskiej-Curie 14, tel. 0857457059
http://wydawnictwo.uwb.edu.pl, e-mail: ac-dw@uwb.edu.pl

Druk i oprawa: „QUICK-DRUK” s.c., Łódź

CONTENTS

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 22 (35) 2010

Mariusz Giero
Institute of Sociology
University of Białystok

FORMALIZATION OF PROPOSITIONAL LINEAR TEMPORAL LOGIC IN THE MIZAR SYSTEM

Abstract: The paper describes formalization of some issues of propositional linear temporal logic (*PLTL*). We discuss encountered problems and applied solutions. The formalization was carried out in the Mizar system. In comparison with other systems, Mizar is famous for its large repository of computer checked mathematical knowledge and also for its user-friendly knowledge representation and proof language.

1. Introduction

Temporal logic [13, 6, 17] is a system of logic, whose language is enriched with temporal connectives. The purpose of temporal logic is reasoning about time (in philosophy), and about the behavior of systems evolving over time (in computer science). Enrichment with temporal connectives results in economy of wording and reasoning closer to intuition than if one uses classical first order logic. There are various temporal logics due to the choice of temporal operators, and what model of time is used. In computer science one usually considers propositional temporal logic with temporal operators *referring only to the future* and *linear flow of time* (isomorphic with the set of natural numbers). The main application of *PLTL* in computer science is the verification of finite-state reactive and concurrent systems (typical examples include microprocessors, operating systems, network protocols, etc.). One of the verification methods is a proof-theoretical method [18]. In this method, verification consists in *proving a theorem* expressing the desired property (e.g. freedom from deadlocks) in a formal deductive theory. The theory consists of formal language in which formulas are written, a set of axioms and inference rules. Propositions specifying the verified system are joined as premises to the theorem one is to prove. In the paper, we define such a deductive theory, with a future plan to be able to carry out verification by this method.

ISBN 978–83–7431–273–8 ISSN 0860-150X

Mizar is the name of a formal language designed by Andrzej Trybulec for writing strictly formalized mathematical definitions and proofs, but is also used as the name of a computer program which is able to check proofs written in this language [9, 15, 19, 1]. Mizar language is similar to the language used in mathematical texts. This makes it an attractive tool for mathematicians who do not have to put much effort in mastering it, gainig the opportunity to check their texts by a machine. Texts written in Mizar (called *articles*), positively verified by the program and positively reviewed by people, are added to the Mizar database of mathematical knowledge (*Mizar Mathematical Library – MML* for short). Currently, *MML* contains over 1,100 articles with more than 50,000 proved theorems, and almost 10,000 definitions of various fields, mostly from mathematics, but also computer science. It is considered to be the largest such database in the world. The list of most important theorems (available at [3]) includes, among others:

- The Fundamental Theorem of Algebra
- The Fundamental Theorem of Arithmetic
- The Fundamental Theorem of Calculus
- The Jordan Curve Theorem
- The Gödel Completeness Theorem

There is also defined Armstrong's deductive system in *MML,* used to infer all the functional dependencies on a relational database.

As an example of readability of Mizar text and its similarity to standard mathematical language let us present the proof of one of the set theory properties:

```
X /\ (Y \/ Z) c= (X /\ Y) \/ (X /\ Z)
proof
  let x be set;
  assume A1: x in X /\ (Y \/ Z);
  then x in Y \/ Z by XBOOLE_0:def 4;
  then A2: x in Y or x in Z by XBOOLE_0:def 3;
  x in X by A1, XBOOLE_0:def 4;
  then x in X /\ Y or x in X /\ Z by A2, XBOOLE_0:def 4;
  hence x in (X /\ Y) \/ (X /\ Z) by XBOOLE_0:def 3;
end;
```

The symbols `/\`, `\/`, `c=`, `in`, `or` denote in Mizar the conventional ones: $\cap$, $\cup$, $\subseteq$, $\in$, $\vee$. Expressions of the form: `by XBOOLE_0:def 4` are justifications with theorems contained in *MML* or local (labeled) propositions. `set` is the broadest type in the Mizar system. Any object extends to the type `set`.

The purpose of formalization is primarily to build a digital database of computer-checked mathematical knowledge. The advantages of such database in comparison with mathematics literature can not be overestimated. The knowledge is stored in one place. It is easier to access, search, process and, most importantly, it has a computer guarantee of correctness. It can also be used for didactic purposes [14, 5].

Formalization is often a tedious activity. Frequently, coding based on paper publication cannot be done directly. Definitions are not precise and formal, proofs (among others, of trivial facts that turn out not necessarily to be trivial) are left to the reader. It happens that the construct of a definition or a proof can not be introduced due to the limitations of a proof-checker system and the user has to invent an equivalent one. Much also depends on what is already formalized in the database of the system. If it lacks basic knowledge in a given field, one, at first, has to introduce it and after that can deal with desired formalization.

The formalization carried out by the author is based mostly on [13, Ch. 2, 3]. There are basics of propositional linear temporal logic with only future-referring connectives in these chapters. We defined syntax and semantics of language of this logic (Sec. 2). An axiomatic system of this logic was introduced (Sec. 3). We proved the soundness theorem and the deduction theorem for this system (Sec. 4, 5). Additionally, numerous minor lemmas and facts were proved. In each section, first, what was formalized is presented, and then how the formalization was carried out is described together with encountered problems and solutions applied. The Mizar code of the formalization is available in [7]. It can be accessed in the *MML* as the Mizar article with the identifier `LTLAXIO1`.

The carried out formalization extends knowledge on temporal logic already stored in the *MML* (see 2.2.3).

2. The language of *PLTL*

2.1. What was formalized

Definition 1.

Syntax of $PLTL$

$$\phi ::= \bot \,|\, pl \,|(\phi \Rightarrow \phi)|(\phi\, \mathbf{Us}\, \phi)$$
$$pl ::= p \,|\, pl'$$

$\bot$ is the *constant false, pl* is a *propositional letter* (for readability we denote propositional letters by p_1, $p_2, p_3, \ldots$ or p, q, r instead of p, p', $p'', \ldots$), $\Rightarrow$

is a classical connective called *implication*, and $\mathbf{Us}$ is a temporal connective called *strong until*. Others classical and temporal connectives are introduced as abbreviations:

Abbreviation.

$\neg\phi \overset{def}{=} (\phi \Rightarrow \bot)(negation)$

$\top \overset{def}{=} \neg\bot\ (constant\ true)$

$\phi_1 \vee \phi_2 \overset{def}{=} (\neg\phi_1 \Rightarrow \phi_2)(disjunction)$

$\phi_1 \wedge \phi_2 \overset{def}{=} \neg(\phi_1 \Rightarrow \neg\phi_2)\ (conjunction)$

$\phi_1 \Leftrightarrow \phi_2 \overset{def}{=} (\phi_1 \Rightarrow \phi_2) \wedge (\phi_2 \Rightarrow \phi_1)\ (equivalence)$

$\mathbf{X}\,\phi \overset{def}{=} (\bot\,\mathbf{Us}\,\phi)\ (next)$

$\phi_1\,\mathbf{U}\,\phi_2 \overset{def}{=} \phi_2 \vee (\phi_1 \wedge (\phi_1\,\mathbf{Us}\,\phi_2))\ (weak\ until)$

$\mathbf{G}\,\phi \overset{def}{=} \phi \wedge \neg(\top\mathbf{U}\,\neg\phi)\ (global)$

$\mathbf{F}\,\phi \overset{def}{=} \neg\mathbf{G}\,\neg\phi\ (future)$

$\phi_1\,\mathbf{R}\,\phi_2 \overset{def}{=} \neg(\neg\phi_1\,\mathbf{U}\,\neg\phi_2)\ (release)$

Definition 2.

Let PL be a set of all propositional letters. A **model** is a function $\mathcal{M} : \mathbb{N} \longrightarrow 2^{PL}$, where $\mathbb{N}$ is a set of natural numbers.

Definition 3.

The **satisfaction** of a $PLTL$ formula ϕ in model $\mathcal{M}$ at time point $n \in \mathbb{N}$ ($\mathcal{M}, n \models \phi$) is defined inductively as follows:

1. $\mathcal{M}, n \not\models \bot$, for every $n \in \mathbb{N}$
2. $\mathcal{M}, n \models pl$ iff $pl \in \mathcal{M}(n)$
3. $\mathcal{M}, n \models \phi_1 \Rightarrow \phi_2$ iff $\mathcal{M}, n \not\models \phi_1$ or $\mathcal{M}, n \models \phi_2$
4. $\mathcal{M}, n \models \phi_1\,\mathbf{Us}\,\phi_2$ iff
   there exists $i \in \mathbb{N}$, $i > 0$: $\mathcal{M}, n+i \models \phi_2$ and
   for every $j \in \mathbb{N}$, $1 \leqslant j < i$: $\mathcal{M}, n+j \models \phi_1$

According to the above definition, the semantics of temporal connectives introduced as abbreviations is as follows:

1. $\mathcal{M}, n \models \mathbf{X}\phi$ iff $\mathcal{M}, n+1 \models \phi$
2. $\mathcal{M}, n \models \mathbf{G}\phi$ iff for every $i \in \mathbb{N}$, $i \geqslant 0$: $\mathcal{M}, n+i \models \phi$
3. $\mathcal{M}, n \models \mathbf{F}\phi$ iff there exists $i \in \mathbb{N}$, $i \geqslant 0$: $\mathcal{M}, n+i \models \phi$
4. $\mathcal{M}, n \models \phi_1\,\mathbf{U}\,\phi_2$ iff
   there exists $i \in \mathbb{N}$, $i \geqslant 0$: $\mathcal{M}, n+i \models \phi_2$ and
   for every $j \in \mathbb{N}$, $j < i$: $\mathcal{M}, n+j \models \phi_1$

5. $\mathcal{M}, n \models \phi_1 \,\mathbf{R}\, \phi_2$ iff
   there exists $i \in \mathbb{N}$, $i \geqslant 0$: $\mathcal{M}, n+i \models \phi_1$ and
   for every $j \in \mathbb{N}$, $j \leq i$: $\mathcal{M}, n+j \models \phi_2$
   or
   for every $i \in \mathbb{N}$: $\mathcal{M}, n+i \models \phi_2$

Definition 4.
A formula ϕ is **valid** in model $\mathcal{M}$ (denoted: $\mathcal{M} \models \phi$) iff for every $n \in \mathbb{N}$: $\mathcal{M}, n \models \phi$

2.2 The formalization

2.2.1. Syntax

The formulas of *PLTL* are defined as finite sequences of natural numbers with the use of Polish notation (prefix notation). $\perp$ is the sequence $\langle 0 \rangle$, propositional letters are sequences of the form $\langle n \rangle$, where $n \geqslant 3$, implications are sequences whose the first element is 1, and *strong until* formulas are sequences whose the first element is 2 [7, see synonyms].

Definition 5.
The set of *PLTL* formulas is the set D such that:

1. $\langle 0 \rangle \in D$
2. if $n \geqslant 3$, then $\langle n \rangle \in D$
3. if $\phi_1, \phi_2 \in D$, then $\langle 1 \rangle^\frown \phi_1{}^\frown \psi_2 \in D$
4. if $\phi_1, \phi_2 \in D$, then $\langle 2 \rangle^\frown \phi_1{}^\frown \psi_2 \in D$
5. for every set D' satisfying 1–4 holds $D \subseteq D'$,

where $\frown$ is a concatenation of sequences. The set D is therefore the smallest set satisfying 1–4.

Example 6.
Let p, q, r be, respectively, sequences $\langle 3 \rangle$, $\langle 4 \rangle$, $\langle 5 \rangle$. The formula $(p \Rightarrow q)\,\mathbf{Us}\, r$ is represented by the sequence $\langle 2, 1, 3, 4, 5 \rangle$. The formula $p\,\mathbf{U}\,q$, after unfolding abbreviations: $(q \Rightarrow \perp) \Rightarrow ((p \Rightarrow ((p\,\mathbf{Us}\,q) \Rightarrow \perp)) \Rightarrow \perp)$, is represented by the sequence $\langle 1, 1, 4, 0, 1, 3, 1, 2, 3, 4, 0, 0 \rangle$.

This method of defining the set of formulas of a language (i. e. as finite sequences of natural numbers) has been used, among others, in [8] for the language of classical propositional logic without negations (*PPL*).

Definition 7.

Syntax of *PPL*

$\phi ::= \top | \, pl \, | (\phi \Rightarrow \phi) | (\phi \wedge \phi),$

$pl ::= p \, | \, pl'.$

$\top$ is the sequence $\langle 0 \rangle$, propositional letters are sequences of the form $\langle n \rangle$, where $n \geqslant 3$, implications are sequences whose first element is 1, and conjunctions are sequences whose first element is 2. In terms of **syntax** the language of *PLTL* is no different from the language of *PPL* ($\perp$ corresponds to $\top$, **Us** corresponds to the binary connective $\wedge$, and the other two symbols *pl* and $\Rightarrow$ are the same). Thus, the definition of *PLTL* was not introduced from the beginning, repeating the definition of *PPL* but there was used one of the features of Mizar language allowing to define synonyms. The language of *PLTL* was defined as a synonym for *PPL.* Thanks to this, the definition took only a few lines of code [7, see synonyms] and it also allowed to make use of several theorems proved in [16], among others, the principal of structural induction for *PPL* which licenses the definition of function with domain *PPL* by structural recursion. Proving the analogous principle for *PLTL* would take about several hundred lines of code.

Abbreviations of other connectives were introduced as *functors* with the `equals` construct [9, p. 18]. The Mizar code for the connective **U** is as follows:

```
definition let p, q be Element of LTLB_WFF ;
  func p 'U' q -> Element of LTLB_WFF equals :: LTLAXIO1:def 8
  q 'or' (p '&&' (p 'Us' q));
end;
```

where `LTLB_WFF` denotes the set of *PLTL* formulas, `'or'` denotes $\vee$, and `'&&'` denotes $\wedge$.

2.2.2. Semantics

It was not possible to formalize the concept of satisfaction directly as it is placed conventionally in literature, i.e., as a recursive predicate (def. 3), due to the fact that Mizar does not allow for this kind of definitions. However, the Mizar system allows defining recursive functions, and with the use of the recursive function $SAT_{\mathcal{M}}$ the satisfaction was defined [7, def. 11]:

Definition 8.

$\mathcal{M}, n \models \phi$ iff $SAT_{\mathcal{M}}(n, \phi) = 1,$

where $SAT_{\mathcal{M}}$ is defined as follows:

Definition 9.

Let $\mathcal{M}$ be a model. $SAT_{\mathcal{M}} : \mathbb{N} \times PLTL \to \{0, 1\}$ is a function satisfying the following conditions:

1. $SAT_{\mathcal{M}}(n, \bot) = 0$, for every $n \in \mathbb{N}$
2. $SAT_{\mathcal{M}}(n, pl) = 1$ iff $pl \in \mathcal{M}(n)$
3. $SAT_{\mathcal{M}}(n, \phi_1 \Rightarrow \phi_2) = 1$ iff $imp(SAT_{\mathcal{M}}(n, \phi_1), SAT_{\mathcal{M}}(n, \phi_2)) = 1$
4. $SAT_{\mathcal{M}}(n, \phi_1 \mathbf{Us}\, \phi_2) = 1$ iff
   there exists $i \in \mathbb{N}$, $i > 0$: $SAT_{\mathcal{M}}(n + i, \phi_2) = 1$ and
   for every $j \in \mathbb{N}$, $1 \leqslant j < i$: $SAT_{\mathcal{M}}(n + j, \phi_1) = 1$,

where *imp* is the Boolean function of implication defined in [2].

The notion of validity in model is then defined directly as the definition 8 says. The Mizar code of the definition of validity is as follows:

```
definition
  let M be LTLModel;
  let p be Element of LTLB_WFF;
  pred M |= p means :: LTLAXIO1:def 12
    for n being Element of NAT holds (SAT M).[n,p] = 1;
end;
```

The function $SAT_{\mathcal{M}}$ was defined as a *functor* which required proving of correctness conditions, i. e., showing that such an object exists (*existence* condition) and is uniquely determined (*uniqueness* condition) [9, p. 17]. There is no general theorem which licenses definition of function by structural recursion. Each case requires a separate proof. In this paper, we made use of the fact that *PLTL* is syntactically synonym of *PPL* and the theorem about the existence of a recursive function over the set of *PPL* formulas [16] was applied.

There were proved theorems [7, Th. 5–13] about the correctness of semantics of connectives introduced as abbreviations (def. 3). The Mizar code for temporal operator **G** is as follows:

```
theorem :: LTLAXIO1:10
for A being Element of LTLB_WFF
for n being Element of NAT
for M being LTLModel holds
((SAT M).[n,('G' A)] = 1 iff
 for i being Element of NAT holds (SAT M).[(n+i),A]=1
)
```

2.2.3. Related Work

There are three Mizar articles concerning temporal logic in *MML*: `MODELC_1` (on computational tree logic) [10], `MODELC_2` (on propositional linear time temporal logic) [11] and `MODELC_3` (on programs verification with the use of Büchi automaton) [12]. In [11] there was defined the following syntax of *PLTL*:

$$\begin{aligned}&\phi ::= pl \,|\neg\phi|\, (\phi \wedge \phi) \,|\, (\phi \vee \phi) \,|\, \mathbf{X}\phi \,|\, (\phi\, \mathbf{U}\, \phi) \,|\, (\phi\, \mathbf{R}\, \phi)\\ &pl ::= p \,|\, pl'\end{aligned}$$

Adding to the language connectives, which can be introduced as abbreviations results in extension of proofs. For example, if one wants to use the principal of structural induction, instead of two cases (two connectives) one has to consider six (for each connective). So, the author of this paper decided to redefine the language of *PLTL* introducing the optimal number of connectives. In [11] formulas of *PLTL* are also formalized as finite sequences of natural numbers with the use of Polish notation. However, the set of these formulas, introduced in [11, def. 9], can be formalized in a shorter and more readable way using the Mizar concept of *attributes* [9, p. 15]. That was done in [8] for *PPL* and we based our definition (def. 5) for *PLTL* on the solution.

Semantics of *PLTL* is defined in [11] by abstract algebra with operations corresponding to connectives. The definition is complex and far from the conventional one known from literature. So, we decided to introduce a new definition (def. 9) which seems to be as close as possible to the conventional one.

3. Axiomatization

3.1. What was formalized

3.1.1 Axioms

The set of axioms consists of:

(A1) all tautologies of classical propositional logic of the *PLTL* language

and all the formulas of the form:

(A2) $\neg\mathbf{X}\phi \Leftrightarrow \mathbf{X}\neg\phi$

(A3) $\mathbf{X}(\phi_1 \Rightarrow \phi_2) \Rightarrow (\mathbf{X}\phi_1 \Rightarrow \mathbf{X}\phi_2)$

(A4) $\mathbf{G}\phi \Leftrightarrow \phi \wedge \mathbf{XG}\phi$

(A5) $\phi_1 \mathbf{Us}\, \phi_2 \Leftrightarrow \mathbf{X}\phi_2 \vee \mathbf{X}(\phi_1 \wedge (\phi_1 \mathbf{Us}\, \phi_2))$

(A6) $\phi_1 \mathbf{Us}\, \phi_2 \Rightarrow \mathbf{XF}\phi_2$

A formula is of the form of (A1) if it results from a tautology of classical propositional logic by consistently replacing the propositional letters of the tautology by formulas of *PLTL.*

Example 10.

The formula $\mathbf{X}p \vee \neg\mathbf{X}p$ is of the form of (A1). It results from the tautology $p \vee \neg p$ by replacing p by $\mathbf{X}p$.

Axioms of the form of (A2) assert linearity of time, (A4), (A5) are the fixpoint characterizations of operators **G** and **Us**, respectively, and axioms of the form of (A6) indicate that the strong version of *until* is used.

3.1.2. Inference rules

There are three inference rules:

(MP): $\phi_1, \phi_1 \Rightarrow \phi_2 \;/ \phi_2$

(NEX): $\phi \;/\; \mathbf{X}\phi$

(IND): $\phi_1 \Rightarrow \phi_2, \phi_1 \Rightarrow \mathbf{X}\phi_1 \;/\; \phi_1 \Rightarrow \mathbf{G}\phi_2$

3.2. The formalization

3.2.1. Axioms

Formalization of the axioms of the form (A1) directly from [13] would require, among others, a definition of replacing operation and a definition of a subformula. We have defined these axioms in another equivalent way, seemingly easier and shorter in formalization. We have used a modified zero-one method for verification whether a formula is a classical propositional logic tautology. The modification consists in the fact that we valuate all the formulas (in particular case also propositional letters) being an argument of an implication but not being an implication themselves. In other words, we valuate $\perp$, propositional letters and *until* formulas, only if they are arguments of an implication (the only classical connective of *PLTL*). The exception is the $\perp$ formula which is always valuated by 0. Then, we calculate the value of the verified formula in accordance with the Boolean function of implication. If, at any such valuation, the value is 1, the formula is of the form (A1), otherwise it is not. If a verified formula is not an implication, we valuate the whole formula. So this type of formula is not of the form (A1), because its value can be 0.

Formalization was carried out in two steps. First, we defined the function V_f for calculating the value of the formula:

Definition 11.

Let $f : PLTL \to \{0,1\}$ be a function and ϕ be a formula of $PLTL$. The value of the formula ϕ at valuation f calculates the function $V_f : PLTL \to \{0,1\}$ such that:

$$V_f(\phi) = \begin{cases} 0, & \text{if } \phi = \perp \\ f(\phi), & \text{if } \phi \in pl \text{ or } \phi = \phi_1 \mathbf{Us}\, \phi_2 \\ imp(V_f(\phi_1), V_f(\phi_2)), & \text{if } \phi = \phi_1 \Rightarrow \phi_2 \end{cases}$$

where *imp* is the Boolean function of implication defined in [2], and then we defined the formulas of the form of (A1) as follows:

Definition 12.

A formula ϕ is of the form of (A1) iff for every function $f : PLTL \to \{0,1\}$ holds $V_f(\phi) = 1$.

The function f valuates any formula, but V_f uses only the values that f assigns to propositional letters and *until* formulas which are not arguments of an implication. If we do not need all values of f, then we could restrict the domain of f to the set of propositional letters and *until* formulas, but it is not necessary. It is always better to have for an object as broad type as possible. Some difficulty in understanding this definition can cause the fact that f can assign 1 to $\perp$, while V_f always assigns 0. However, there is no inconsistency, because this particular value of f is not used by V_f. The restriction of a function f would be worth to consider in case of implementation of the method, it could affect how long the program would run.

Example 13.

The formula $\mathbf{X}p \vee \neg\mathbf{X}p$ is of the form of (A1). We evaluate the function V_f for the formula:

$$\begin{aligned} V_f(\mathbf{X}p \vee \neg\mathbf{X}p) &= V_f(((\perp \mathbf{Us}\, p) \Rightarrow \perp) \Rightarrow ((\perp \mathbf{Us}\, p) \Rightarrow \perp)) \\ &= imp(imp(f(\perp \mathbf{Us}\, p), 0), imp(f(\perp \mathbf{Us}\, p), 0) \end{aligned}$$

and we have:

if $f(\perp \mathbf{Us}\, p) = 0$,
then $V_f(Xp \vee \neg Xp) = imp(imp(0,0), imp(0,0)) = imp(1,1) = 1$,

and also

if $f(\perp \mathbf{Us}\, p) = 1$,
then $V_f(Xp \vee \neg Xp) = imp(imp(1,0), imp(1,0)) = imp(0,0) = 1$.

The function V_f is recursive. As in the definition of the function $SAT_{\mathcal{M}}$, we had to show that such a function exists and we also used the principal of structural induction proved in [16]. The Mizar code of the definition of V_f (denoted in Mizar as: `VAL f`) is as follows:

```
definition
  let f be Function of LTLB_WFF,BOOLEAN;
  func VAL f -> Function of LTLB_WFF,BOOLEAN means
    :: LTLAXIO1:def 15
    for A, B being Element of LTLB_WFF
    for n being Element of NAT holds
    (it.TFALSUM = 0 & it.(prop n)=f.(prop n) &
    it.(A=>B)=(it.A)=>(it.B) &
    it.(A 'Us' B)=f.(A 'Us' B));
  correctness;
end;
```

The set of axioms is defined as the smallest subset AKS of the set of $PLTL$ formulas, i.e., the set satisfying the following conditions:

1. If ϕ is of the form of A1, then $\phi \in AKS$
2. If ϕ is of the form of A2–A6, then $\phi \in AKS$
3. for every set AKS' satisfying 1–2 holds $AKS \subseteq AKS'$

3.2.2. Inference rules

The rules were defined as relations on the set of $PLTL$. The definition of the MP-rule is as follows:

$$(\phi_1, \phi_2, \phi_3) \in MP \text{ iff } \phi_2 = \phi_1 \Rightarrow \phi_3$$

The Mizar code of the definition is as follows:

```
definition
  let p,q,r be Element of LTLB_WFF ;
  pred p,q MP_rule r means :: LTLAXIO1:def 19
  q = p => r;
end;
```

4. The Soundness Theorem

Theorem.

For every set of formulas $\mathcal{F}$ and a formula ϕ holds: if $\mathcal{F} \vdash \phi$, then $\mathcal{F} \models \phi$.

The Mizar code of the theorem is as follows:

```
theorem :: LTLAXIO1:41
  for A being Element of LTLB_WFF
  for F being Subset of LTLB_WFF st F |- A
  holds F |= A;
```

Semantic consequence, $\mathcal{F} \vDash \phi$, introduces the following definition:

```
definition
  let F be Subset of LTLB_WFF;
  let p be Element of LTLB_WFF ;
  pred F |= p means :: LTLAXIO1:def 14
    for M being LTLModel st M |= F holds M |= p;
end;
```

A set of formulas $\mathcal{F}$ is valid in model $\mathcal{M}$ (denoted in Mizar: `M |= F`) iff every formula being element of F is valid (as stated in def. 4) in model $\mathcal{M}$.

Syntactic consequence, $\mathcal{F} \vdash \phi$ (a derivation of a formula ϕ from a set of formulas $\mathcal{F}$) is defined conventionally, i.e. as a finite sequence of formulas such that:

- elements of the sequence are: axioms or elements of the set $\mathcal{F}$ or formulas derived from the earlier element(s) by inferences rules
- the last element of the sequence is the formula ϕ.

The proof of the theorem is based on [13]. From assumption, we have that there exists a finite sequence of formulas f (as described above) such that the last element of the sequence is ϕ. Then, we prove that for every element of the sequence f holds $\mathcal{F} \models f_i$, where $1 \leqslant i \leqslant |f|$, applying the following induction rule [4, sch. 4]:

$$\forall i \in \mathbb{N}\ P(i) \text{ iff } \forall i \in \mathbb{N}\ (\forall k \in \mathbb{N}\ (k < i \Rightarrow P(k)) \Rightarrow P(i)), \quad \text{(indrule)}$$

where predicate $P(i)$ is defined as $P(i) \overset{def}{=} \mathcal{F} \models f_i$

Since $P(i)$ holds for every element of the sequence f, it also holds for the last element and the thesis is proved. The proof required considering all the cases of what f_i can be. For each case, semantic consequence was showed [7, Th. 24–27, 37]:

- any axiom is semantic consequence of $\mathcal{F}$
- any element of $\mathcal{F}$ is semantic consequence of $\mathcal{F}$
- for each of the rules of inference: if the premises is semantic consequence of $\mathcal{F}$, then so is the conclusion.

5. The Deduction Theorem

Theorem.

For every set of formulas $\mathcal{F}$ and formulas ϕ_1, ϕ_2 holds: if $\mathcal{F} \cup \{\phi_1\} \vdash \phi_2$, then $\mathcal{F} \vdash \mathbf{G}\phi_1 \Rightarrow \phi_2$.

The Mizar code of the theorem is as follows:

```
theorem :: LTLAXIO1:57
  for p,q being Element of LTLB_WFF
  for F being Subset of LTLB_WFF st F \/ {p}|- q
  holds F|-('G' p) => q;
```

The proof of the theorem [13] is analogous to the previous one. From assumption, we have that there exists a finite sequence of formulas f such that the last element of the sequence is ϕ_2. Then, we prove that for every element of the sequence f holds $\mathcal{F} \vdash \mathbf{G}\phi_1 \Rightarrow f_i$, where $1 \leqslant i \leqslant |f|$, applying induction rule (*indrule*). We define $P(i)$ as follows:

$$P(i) \ \overset{def}{=} \ \mathcal{F} \vdash \mathbf{G}\phi_1 \Rightarrow f_i.$$

Since $P(i)$ holds for every element of the sequence f, it also holds for the last element and the thesis is proved.

Sketch of the proof:

We consider all the cases of what f_i can be:

1. The formula $f_i \Rightarrow (\mathbf{G}\phi_1 \Rightarrow f_i)$ is an axiom of the (A1) form, so $\mathcal{F} \vdash f_i \Rightarrow (\mathbf{G}\phi_1 \Rightarrow f_i)$. If f_i is an axiom or $f_i \in \mathcal{F}$, then $\mathcal{F} \vdash f_i$. Then, we infer $P(i)$ applying (MP) rule.
2. The formula $\mathbf{G}\phi_1 \Leftrightarrow \phi_1 \wedge \mathbf{XG}\phi_1$ is an axiom of the (A4) form, so $\mathcal{F} \vdash \mathbf{G}\phi_1 \Leftrightarrow \phi_1 \wedge \mathbf{XG}\phi_1$. If $f_i = \phi_1$, then we infer $P(i)$ applying (MP) rule and the classical logic tautology (particular case of (A1)).
3. If f_i is derived from a inference rule, e.g., (*NEX*) rule, then there exists $j < i$: $f_i = \mathbf{X}f_j$. $P(j)$ holds under the assumption of the induction. Then, we infer $\mathcal{F} \vdash \mathbf{G}\phi_1 \Rightarrow \mathbf{X}f_j$ applying successively:
   - the NEX rule: $\mathcal{F} \vdash \mathbf{X}(\mathbf{G}\phi_1 \Rightarrow f_j)$
   - axiom of the (A3) form and the MP rule: $\mathcal{F} \vdash \mathbf{XG}\phi_1 \Rightarrow \mathbf{X}f_j$
   - axiom of the (A4) form, the classical tautology and the MP rule: $\mathcal{F} \vdash \mathbf{G}\phi_1 \Rightarrow \mathbf{XG}\phi_1$
   - axiom of the (A1) form (transitivity of implication) and the MP rule
4. If f_i is derived from the *(MP)* or (*IND*) rule, we proceed by analogy to p. 3.

The proof required showing numerous minor facts (sometimes trivial), among others:

- every axiom is syntactic a consequence of any set of formulas [7, Th. 42]
- for each of the rules of inference: if the premises is a syntactic consequence of a set of formulas, then so is the conclusion [7, Th. 43–45]

- if $\phi_1 \Rightarrow \phi_2$ is a classical propositional logic tautology and ϕ_1 is a syntactic consequence of a set of formulas, then so is ϕ_2. For example, if $\mathcal{F} \vdash p \Rightarrow q \wedge r$, then $\mathcal{F} \vdash p \Rightarrow q$. This property and others of this kind were derived from the MP-rule and the formulas of the form of (A1) [7, Th. 46–52]

The Deduction Theorem of classical propositional logic does not hold generally in *PLTL*.

6. Future Work

To finish formalization of the deductive system of *PLTL,* the completeness theorem must be proved: if $\mathcal{F} \models \phi$, then $\mathcal{F} \vdash \phi$. Formalization of deductive systems for other temporal logics (*CTL – Computational Tree Logic, CTL* –* a combination of *PLTL* and *CTL*) would be a valuable extension of the *MML* database. Another valuable extension would be Mizar articles on specification of hardware and software. Then, having a deductive system and being able to formalize specification, Mizar could be used as a tool for hardware and software verification by proof-theoretical method, i.e., by proving the theorem expressing the desired property (e.g. mutual exclusion) in the deductive system. The correctness of the proof would be computer checked (by Mizar).

REFERENCES

[1] Mizar Home Page, http://mizar.org.

[2] On the Arithmetic of Boolean Values. Available at http://mizar.uwb.edu.pl/version/current/html/xboolean.html.

[3] Bancerek G., The MML Query Home Page. Available at http://mmlquery.mizar.org/mmlquery/fillin.php?filledfilename=mml-facts.decoded.mqt&argument=number+1.

[4] Bancerek G., The Fundamental Properties of Natural Numbers. *Formalized Mathematics*, 1 (1): 40–46, 1990.

[5] Borak E. and Zalewska A., Mizar Course in Logic and Set Theory. In *Proceedings of Calculemus '07 / MKM '07 Proceedings of the 14th symposium on Towards Mechanized Mathematical*, pages 191–204, Berlin Heidelberg , 2007, Springer-Verlag.

[6] Gabbay D. M., Finger M. and Reynolds M., *Temporal Logic: Mathematical Foundations and Computational Aspects*, volume 1, Oxford University Press, New York, 1994.

[7] Giero M., The Axiomatization of Propositional Linear Time Temporal Logic, Available at http://mizar.uwb.edu.pl/version/current/html/ltlaxio1.html.

[8] Grabowski A., Hilbert Positive Propositional Calculus, *Formalized Mathematics*, 8 (1): 69–72, 1999.

[9] Grabowski A., Korniowicz A. and Naumowicz A., Mizar in a Nutshell, In Assperti A., Harrison J. and Munoz C., editors, *Journal of Formalized Reasoning*, volume 3 of *User Interface I*, pages 153–245, 2010.

[10] Ishida K., Model Checking. Part I, *Formalized Mathematics*, 14 (4): 171–186, 2006.

[11] Ishida K., Model Checking. Part II, *Formalized Mathematics*, 16 (3): 231–245, 2008.

[12] Ishida K. and Shidama Y., Model Checking. Part III, *Formalized Mathematics*, 16 (4): 339–353, 2008.

[13] Kröger F. and Merz S., *Temporal Logic and State Systems*, Springer-Verlag, Berlin Heidelberg, 2008.

[14] Naumowicz A., Teaching How to Write a Proof. In *Proceedings of ETAPS 2008 satellite workshop Formal Methods in Computer Science Education (FORMED2008)*, pages 91–100, Budapest, 2008.

[15] Trybulec A., Some Features of the Mizar Language, In *Procedings of ESPRIT Workshop*, Torino, 1993.

[16] Trybulec A., Defining by Structural Induction in the Positive Propositional Language, *Formalized Mathematics*, 8 (1): 133–137, 1999.

[17] Trzęsicki K., *Logika temporalna. Wybrane zagadnienia*, Wydawnictwo UwB, Białystok 2008.

[18] Trzęsicki K., Temporal Logic Model Checkers as Applied in Computer Science, *Studies in Logic, Grammar and Rhetoric*, 17 (30): 13–125, 2009.

[19] Wiedijk F., Writing a Mizar article in nine easy steps, Available at http://www.cs.ru.nl/˜freek/mizar/mizman.pdf.

Mariusz Giero
Institute of Sociology
University of Białystok
giero@uwb.edu.pl

Dariusz Surowik
University of Białystok
College of Computer Science and Business Administration in Łomża

TEMPORAL-EPISTEMIC LOGIC

Abstract: The paper aims at providing temporal epistemic logic TEL with sound and complete axiomatization. This logic combines temporal and epistemic operators. Time is represented as isomorphic to the set of natural numbers, whereas knowledge is modeled as an $S5$-like modality.

Keywords: Temporal logic, Epistemic logic, Modal logic

Time and knowledge are strongly related. Temporal reasoning and knowledge representation is getting more attention in recent years. To formalize considerations on knowledge changing in time we use some combined logic. The natural way to construct this kind of logic is to insert epistemic logic in a temporal framework. We add two sets of specific operators to an existing propositional system. One set of specific operators is the set of temporal operators, which we use to describe temporal interactions. Because of the basic language in which the agent can describe his knowledge is propositional epistemic logic, then the second set of specific operators is the set of epistemic operators. We construct a multi-modal system combining tense and knowledge operators. Reasoning can be seen as an activity of an agent taking place in time. It is a stepwise process. The agent starts with a set of initial facts to which it applies some rules to arrive at a new state, in which he has more knowledge. In this new state, the agent may apply rules to arrive at a next state. We assume that semantic time is linear, discrete and has a starting point (time is isomorphic to the set of natural numbers). Moreover, we assume that the agent may perform positive and negative introspection, then knowledge is modeled as an S5-like modality. We construct sound and complete system of temporal epistemic logic with *Since* and *Until* operators. To prove the completeness of our system we use the method described in [6].

ISBN 978–83–7431–273–8 ISSN 0860-150X

Temporal-epistemic logic *TEL*

The knowledge of our agent can be modeled by modalities of $S5$ epistemic logic, then our epistemic language is the language of $S5$ logic.

Definition 1 *(Epistemic language)*
Let $Prop$ be a countably set of propositional atoms. The language $\mathcal{L}_{S5}$ is the smallest set closed under:

1. if $p \in Prop$, then $p \in \mathcal{L}_{S5}$,
2. if $\varphi, \psi \in \mathcal{L}_{S5}$, then $\neg\varphi, K\varphi, (\varphi \wedge \psi) \in \mathcal{L}_{S5}$.

K is knowledge operator and a formula $K\varphi$ is interpreted as follows: *The agent knows that* φ. We introduce the following abbreviations:

- $\varphi \vee \psi \equiv \neg(\neg\varphi \wedge \neg\psi)$,
- $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$,
- $M\varphi \equiv \neg K \neg\varphi$,
- $\top \equiv p \vee \neg p$,
- $\bot \equiv \neg\top$,

where: $\vee$ and $\rightarrow$ are classical disjunction and implication, respectively. M is a possibility operator. A formula $M\varphi$ is interpreted as follows: φ *is possible with respect to a knowledge of an agent.* $\top$ and $\bot$ are abbreviations for *constant true* and *constant false* respectively.

We make our temporal epistemic logic by adding a temporal dimension to $S5$ epistemic logic, then a temporal epistemic language we define in the following way:

Definition 2 *(Temporal epistemic language $\mathfrak{L}_{TEL}$)*
The temporal epistemic language $\mathfrak{L}_{TEL}$ is the smallest set closed under:

- If $\varphi \in \mathcal{L}_{S5}$, then $\varphi \in \mathfrak{L}_{TEL}$
- If $\varphi, \psi \in \mathfrak{L}_{TEL}$, then $\neg\varphi, (\varphi \wedge \psi), U(\varphi, \psi), S(\varphi, \psi) \in \mathfrak{L}_{TEL}$

U and S are temporal operators. A formula $U(\varphi, \psi)$ we read: ψ *holds until* φ *does*, and a formula $S(\varphi, \psi)$ we read: ψ *holds since* φ *does.* Again the abbreviations for $\vee, \rightarrow, \top$ and $\bot$, we introduce, as well as:

- $F\varphi \equiv U(\varphi, \top)$,
- $P\varphi \equiv S(\varphi, \top)$,
- $G\varphi \equiv \neg(U(\neg\varphi, \top))$,
- $H\varphi \equiv \neg(S(\neg\varphi, \top))$,
- $\bigcirc\varphi \equiv U(\varphi, \bot)$,
- $\Box\varphi \equiv H\varphi \wedge \varphi \wedge G\varphi$.

The interpretation of introduced temporal operators is as follows:

- $F\varphi$ – sometimes in the future φ,
- $P\varphi$ – sometimes in the past φ,
- $G\varphi$ – always in the future φ,
- $H\varphi$ – always in the past φ,
- $\bigcirc\varphi$ – at the next moment of time φ,
- $\Box\varphi$ – always φ.

Semantics of *TEL*

Definition 3 *(TEL-model)*

TEL-model $\mathfrak{M} = \langle W, \mathbb{N}, \{A_t : t \in \mathbb{N}\}, R, V\rangle$
where:

- W is nonempty set,
- $\mathbb{N}$ is the set of natural numbers,
- $\{A_t : t \in \mathbb{N}\}$ is a collection of accessibility relations,
- $R(\subseteq \mathbb{N} \times \mathbb{N})$ is "earlier-later" relation,
- $V : Prop \to 2^{\mathbb{N}\times W}$.

W is the set of worlds. We assume, that semantic time in our system is isomorphic to the set of natural numbers, then $\mathbb{N}$ is the set of moments of time. Each moment of time is assigned to a model of epistemic $S5$ logic, hence to each moment of time we assign to accessibility relation (one relation for each moment of time). Thus for each $t \in \mathbb{N}$, $A_t \subseteq W \times W$.

V is function mapping to each propositional letter p subset $V(p)$ of cartesian product $\mathbb{N} \times W$. $V(p)$ is the set consist of pairs (t, w) such that p is true in the world w at the moment t.

Definition 4 *(The satisfiability of a formula)*

The satisfiability of a formula $\varphi \in \mathfrak{L}_{TEL}$ in a model $\mathfrak{M}$, at a moment of time $t \in \mathbb{N}$, in a world $w \in W$, denoted by $\mathfrak{M} \models \varphi[t, w]$, is defined inductively as follows:

1. $\mathfrak{M} \models \varphi[t, w] \quad \equiv (t, w) \in V(\varphi)$, if $\varphi \in Prop$,
2. $\mathfrak{M} \models \neg\varphi[t, w] \quad \equiv$ it is not the case that $\mathfrak{M} \models \varphi[t, w]$,
3. $\mathfrak{M} \models (\varphi \wedge \psi)[t, w] \quad \equiv \mathfrak{M} \models \varphi[t, w]$ and $\mathfrak{M} \models \psi[t, w]$,
4. $\mathfrak{M} \models U(\varphi, \psi)[t, w] \quad \equiv \exists t'\ tRt'$ such that $\mathfrak{M} \models \varphi[t', w]$ and $\forall u \in T$ such that $(tRu$ and $uRt')$ holds $\mathfrak{M} \models \psi[u, w]$,

5. $\mathfrak{M} \models S(\varphi, \psi)[t, w] \quad \equiv \exists t'\ t'Rt$ such that $\mathfrak{M} \models \varphi[t', w]$ and $\forall u \in T$ such that $(t'Ru$ and $uRt)$ holds $\mathfrak{M} \models \psi[u, w]$,
6. $\mathfrak{M} \models K\varphi[t, w] \quad \equiv \forall w' \in W$ such that wA_tw' holds $\mathfrak{M} \models \varphi[t, w']$.

Definition 5 *(The truth of formula in a model)*
Let $\mathfrak{M} = \langle \mathbb{N}, W, \{A_t : t \in T\}, R, V \rangle$.
If for all $t \in \mathbb{N}$ and for all $w \in W$, $\mathfrak{M} \models \varphi[t, w]$, then the formula φ is true in a model $\mathfrak{M}$, and we write $\mathfrak{M} \models \varphi$.

Definition 6 *(The truth of a formula)*
If φ is true in all models, then φ is true and we write $\models \varphi$.

Axiomatization of *TEL*

TEL system is axiomatizable. To create axiom system for TEL we combine axioms for modal logic $S5$ with axioms for temporal logic of time isomorphic to the set of natural numbers. Axioms for the logic $S5$ is well known. Axioms for temporal logic for natural numbers were given by Burgess in the paper [2].

Definition 7 *(Axioms for TEL)*
Axioms system for temporal-epistemic logic TEL is as follows:

1. All instances of propositional tautologies
2. $G(\varphi \to \psi) \to (U(\varphi, \xi) \to U(\psi, \xi))$,
3. $G(\varphi \to \psi) \to (U(\xi, \varphi) \to U(\xi, \psi))$,
4. $(\varphi \wedge U(\psi, \xi)) \to U(\psi \wedge S(\varphi, \xi), \xi)$,
5. $(U(\varphi, \psi) \wedge \neg U(\varphi, \xi)) \to U(\psi \wedge \neg\xi, \psi)$,
6. $U(\varphi, \psi) \to U(\varphi, \psi \wedge U(\varphi, \psi))$,
7. $U(\psi \wedge U(\varphi, \psi), \varphi) \to U(\varphi, \psi)$,
8. $(U(\varphi, \psi) \wedge U(\xi, \phi)) \to ((U(\varphi \wedge \xi, \psi \wedge \phi) \vee U(\varphi \wedge \phi, \psi \wedge \phi) \vee$
$U(\psi \wedge \xi, \psi \wedge \phi))$,
9. the mirror images of axioms 2-8[1],
10. $U(\top, \bot) \wedge S(\top, \bot)$,
11. $H\bot \vee PH\bot$,

[1] The *mirror image of* φ is obtained by simultaneously replacing S by U and U by S, everywhere in φ.

12. $F\top$,
13. $F\varphi \to U(\varphi, \neg\varphi)$.

Rules:

1. $\dfrac{\varphi, \varphi \to \psi}{\varphi}$ (*Modus Ponens*)
2. $\dfrac{\varphi}{G\varphi} \qquad \dfrac{\varphi}{H\varphi}$ (*Temporal generalization*)
3. $\dfrac{\vdash_{S5} \varphi}{\vdash_{TEL} \varphi}$, for every formula $\varphi \in \mathfrak{L}_{S5}$ (*Preserve*)

Let us remark, that the axioms of (1) and (2) in some sense correspond to axioms for modal logic K. Axiom (3) relates the mirror image connectives. Axioms (6) and (7) express transitivity. Axiom (8) express linearity. Axioms (11), (12) and (13) relates respectively: having a first point, right-seriality, Prior-style Dedekind completeness.

We construct our temporal-epistemic logic according to the method of addition of a temporal dimension to the logic system described in the paper [6]. In our case we add a temporal dimension to the modal epistemic logic $S5$. We temporalize $S5$ epistemic logic.

TEL is sound and complete. The soundness of TEL we obtain by theorem 2.2 of [6] using soundness of $S5$ and soundness of the axiom system for temporal logic of time isomorphic to the set of natural numbers.

Theorem 1 *(Soundness TEL)*
The axiom system of TEL is sound.

The logic $S5$ is complete. The Burgess's system given in [2] is complete over a class of time isomorphic to the set of natural numbers. Then by theorem 2.3 of [6] we obtain completeness of TEL. Therefore, we have the following theorem:

Theorem 2 *(Completeness TEL)*
The axiom system of TEL is complete.

And finally, by theorem 3.1 of [6], decidability of $S5$ logic [8] and decidability of the temporal logic of time isomorphic to the set of natural numbers [9] we have:

Theorem 3 *(Decidability of TEL)*
The TEL logic is decidable.

R E F E R E N C E S

[1] Burgess J. P., *Axioms for Tense Logic I. "Since" and "Until"*, Notre Dame Journal of Formal Logic, 23, pp. 367–374, 1982.

[2] Burgess J. P., *Logics of Time and Comutation*, CSLI Lecture Notes, No. 7, Stanford University Press, Palo Alto, 1992.

[3] Calardo E. and Rybakov V., *An axiomatisation for the multimodal logic of knowledge and linear time LTK*, Logic Journal of IGPL, 2007, vol. 15, no. 3, pp. 239–254.

[4] Engelfriet J., Treur J., *A temporal model theory for default logic*, Proceedings of 2nd European Conference on Symbolic and Quantitaive Approaches to Reasoning an Uncertainty, ECSQARU '93, edited by M. Clarke, R. Kruse and S. Moral, Lecture Notes in Computer Science, vol. 747, Springer-Verlag, Berlin, 1993.

[5] Joeri Engelfriet, *Minimal temporal epistemic logic*, Notre Dame Journal of Formal Logic, 1996, vol. 37, no. 2, pp. 233–259.

[6] Finger M., Gabbay Dov M., *Adding a temporal dimension to a logic system*, Journal of Logic, Language and Information, vol. 1, pp. 203–233. 1992.

[7] Fitch F., *A Logical Analysis of Some Value Concepts*, The Journal of Symbolic Logic, 28, 1963, pp. 135–142.

[8] Meyer J. J. Ch. and van der Hoek W., *Epistemic Logic for Computer Science*, vol. 41, Cambidge University Press, Cambridge, 1995.

[9] Sistla A. P. and Clarke E. M., *The complexity of propositional linear temporal logics*, Journal of ACM, vol. 32 (1985), pp. 733–749.

Dariusz Surowik
Chair of Logic, Informatics and Philosophy of Science
University of Białystok
surowik@uwb.edu.pl

Marcin Koszowy
University of Białystok

PRAGMATIC LOGIC AND THE STUDY OF ARGUMENTATION[1]

Abstract: The main achievements of the Lvov–Warsaw School (LWS) are associated by argumentation theorists mostly with the developments of mathematical logic. However, in the LWS there was carried on also research which may be particularly inspiring for the study of argumentation: systematic investigation of applying language and methods of logic in order to develop knowledge and skills which constitute the so-called *logical culture*. The discipline which aimed at developing these skills was called *pragmatic logic*; this is also the title of Kazimierz Ajdukiewicz's book of 1965. The aim of this paper is to show that the pragmatic approach to reasoning and argumentation constituted the core concern of the LWS as early as in the first half of the 20th century. This aim is realized by discussing the subject-matter, goals and methods of pragmatic logic. I argue that (1) some crucial assumptions of pragmatic logic harmonize with those accepted in argumentation theory, (2) pragmatic logic is a legitimate tool in the study of argumentation.

Keywords: pragmatic logic, logical culture, Lvov–Warsaw School (LWS), Kazimierz Ajdukiewicz, argumentation theory

1. The main question

Logical studies in Poland are mainly associated with the Lvov–Warsaw School (LWS), labeled also *the Polish school* in analytical philosophy (Lapointe, Woleński, Marion & Miskiewicz 2009; Jadacki 2009).[2] Among other

[1] The earlier version of this paper was presented at the Seventh Conference of the International Society for the Study of Argumentation (ISSA) in Amsterdam (June 29–July 2, 2010). It is to be published by Sic Sat in the ISSA Conference Proceedings. I am grateful to Prof. Ralph H. Johnson for discussion which was inspiring for raising the main question of this paper. I would also like to thank Prof. Agnieszka Lekka-Kowalik for her helpful comments.

[2] The LWS was established by Kazimierz Twardowski at the end of the 19th century in Lvov (Woleński 1989, Ch. 1, part 2). It should be noted that Polish analytical philosophy is a broader enterprise than the LWS, since there were prominent analytic philosophers, such as Leon Chwistek or Roman Ingarden, who did not belong to the school (see Jadacki 2009, p. 7). However, the analytic approach to language and methods of science constituted the key feature of the research carried out in the school.

achievements of the school, the developments of mathematical logic (see Kneale & Kneale 1962; McCall 1967; Coniglione, Poli & Woleński 1993) became world-wide famous thanks to such thinkers as Jan Łukasiewicz, Stanisław Leśniewski, Alfred Tarski, Bolesław Sobociński, Andrzej Mostowski, Adolf Lindenbaum, Stanisław Jaśkowski and many others (see e.g. Woleński 1995, p. 369–378).

In 'the golden age of Polish logic', which lasted for two decades (1918–1939), 'formal logic became a kind of international visiting card of the School as early as in the 1930s – thanks to a great German thinker, Scholz' (Jadacki 2009, p. 91).[3] Due to this fact, some views on the study of reasoning and argumentation in the LWS were associated exclusively with a formal-logical (deductivist) perspective, according to which a good argument is the one which is deductively valid.[4] Having as a point of departure a famous controversy over the applicability of formal logic (or FDL – formal deductive logic – see Johnson & Blair 1987; Johnson 1996; Johnson 2009) in analyzing and evaluating everyday arguments, the LWS would be commonly associated with deductivism.[5]

However, this formal-logical interpretation of the studies of reasoning and argumentation carried on in the LWS does not do full justice to its subject-matter, research goals and methods of inquiry. There are two reasons supporting this claim:

(1) Although logic became the most important research field in the LWS, its representatives were active in all subdisciplines of philosophy (Woleński 2009). The broad interest in philosophy constitutes one of the reasons for searching applications of logic in formulating and solving philosophical problems.

(2) Some of the representatives of the LWS developed a pragmatic approach to reasoning and argumentation. Concurrently with the developments in formal logic, research was carried out which – although much less known – turns out to be particularly inspiring for the study of argumentation: systematic investigation consisting in applying language and methods of logic in order to develop skills which constitute 'logical culture'. Two ba-

[3] Heinrich Scholz, who is claimed to be the first modern historian of logic (Woleński 1995, p. 363) called Warsaw one of the capitals of mathematical logic (Scholz 1930).

[4] Such views on formal logic are indicated by Johnson (2009, p. 17).

[5] Deductivism is the view concerning the criteria which allow us to distinguish good and bad reasoning. The main thesis of deductivism states that good reasoning in logic is minimally a matter of deductively valid inference (Jacquette 2009, p. 189). The logical tradition of the LWS accepts deductivism, however it deals not only with reasoning, but also with broader 'logical' norms of defining, questioning or ordering. For the detailed characteristic of deductivism see Jacquette 2007, Jacquette 2009 and Marciszewski 2009.

sic skills that the logical culture focuses on are: describing the world in a precise language and correct reasoning. My paper concentrates on the second point.

The discipline which aimed at describing these skills and showing how to develop them was called "Pragmatic Logic"; this is also the English title of Kazimierz Ajdukiewicz's 1965 book *Logika pragmatyczna* (see Ajdukiewicz 1974). The program of pragmatic logic may be briefly characterized as applying general rules of scientific investigation in everyday communication. This inquiry focused on the question whether the tools of logic can be used to educate people to (1) think more clearly and consistently, (2) express their thoughts precisely and systematically, (3) make proper inferences and justify their claims (see Ajdukiewicz 1957, p. 3). It should be added that this pragmatic approach to logic was something more fundamental than just one of many ideas of the school: it constituted the *raison d' être* of the didactic program of the LWS. Thus, the pragmatic approach to reasoning and argumentation had a strong institutional dimension: teaching how to think logically was one of the main goals of the school. The joint effort of propagating the developments of logic and exposing the didactic power of logic as a tool of broadening the skills of thinking logically may be illustrated by the passage from the status of the Polish Logical Association, founded on the initiative of Jan Łukasiewicz and Alfred Tarski in April 22nd, 1936.[6] The aim of the association was 'to practice and propagate logic and methodology of science, their history, didactics and applications' (see *The History of the Polish Society for Logic and Philosophy of Science*).

The inspiration for exposing this research field in the LWS comes from numerous publications on the origins of the informal logic movement and the pragma-dialectical theory of argumentation. In their writings informal logicians and pragma-dialecticians explained the phenomenon of revitalizing argumentation theory in the 1970s (e.g. Johnson & Blair 1980; Woods, Johnson, Gabbay & Ohlbach 2002; van Eemeren & Grootendorst 2004; Blair 2009; Johnson 2009; van Eemeren 2009). They indicated a pragmatic need to evaluate arguments in the context of everyday communication as one of the main causes of this phenomenon. Thus, at the beginning of the modern study of arguments in the early 1970s we observe the 'marriage of theory and practice' in the study of logic (Kahane 1971, p. vii; see Johnson 2009,

[6] The first President of the Association was Jan Łukasiewicz. The other members of the first Executive Board were Adolf Lindenbaum, Andrzej Mostowski, Bolesław Sobociński and Alfred Tarski. The constitution of the Association was adopted in 1938 (see *The history of the Polish Society for Logic and Philosophy of Science*).

p. 19). In the case of the LWS this ‘marriage’ was realized by treating formal and pragmatic logic as two interrelated, and not competing, wings of inquiry.

From what has been said above, some similarities are noticeable between the approaches of the LWS and contemporary argumentation theory (including informal logic and pragma-dialectics). My paper aims at making those similarities more explicit, so I raise the question: what relation obtains between logical studies carried on in the LWS and the recent study of argumentation? The answer is given in three steps. In section 2 I present some elements of the conceptual framework of the LWS, which are relevant for exploring connections between the school and argumentation theory. Among those elements there are concepts of: (a) logic, (b) logical fallacy, (c) argument, and (d) knowledge-gaining procedures. These concepts are helpful for introducing the conception of (e) logical culture. In section 3 I discuss some crucial elements of the program of pragmatic logic, which was aimed at elaborating a theoretical background for developing knowledge and skills of logical culture. Among those elements there are: (a) the subject-matter of pragmatic logic and (b) its main goals. Section 4 explores some perspectives for the rapprochement of pragmatic logic with argumentation theory. In the paper I refer to the works of the representatives of the LWS, as well as to the tradition of the school that is continued to this day.

2. The conceptual framework of the LWS

2.1. Logic

Due to its achievements in formal logic the LWS is usually associated with the view on logic as a formal theory of sentences (propositions) and relationships between them. This understanding of ‘logic’ (so-called ‘narrow conception of logic’) is dissociated from the ‘broad conception of logic’ that embraces also semiotics and methodology of science (see e.g. Ajdukiewicz 1974, p. 2–4). Both conceptions of logic are employed in the tradition of the LWS what is illustrated by the fact that in it ‘logical skills’ encompass not only formal-logical skills, but also skills which can be described as using tools elaborated in semiotics, e.g. universal tools for analyzing and evaluating utterances, and in the methodology of science, e.g. tools for developing and evaluating definitions, classifications, and questions occurring in scientific inquiry (see the Appendix A in Johnson 2009, p. 38–39). An interesting example of the broader account of logic can be found in Tarski (1995, p. xi). ‘Logic’ refers here to the discipline ‘which analyses the meaning of the concepts common to all the sciences, and establishes the general laws governing the concepts'. So, if such a notion of logic is introduced, its consequence relies on treating semiotics (the discipline within which the analysis of concepts is one of the crucial procedures) and the methodology of science (the one dealing with principles of scientific inquiry) as fundamental parts of logic.[7]

Other members of the LWS gave substantial reasons for treating the methodology of science as an element of logic in the broad sense. Jan Woleński makes this point explicit by focusing on the philosophy of science as a discipline that uses tools of logic in exploring the structure of scientific theories:

> The philosophy of science was a favourite field of the LWS. Since science is the most rational human activity, it was important to explain its rationality and unity. Since most philosophers of the LWS rejected naturalism in the humanities and social sciences, the way through the unity of language (as in the case of the Vienna Circle) was excluded. The answer was simple: science qua science is rational and is unified by its logical structure and by definite logical tools used in scientific justifications. Thus, the analysis of the inferential machinery of science is the most fundamental task of philosophers of science (Woleński 2009).

Treating philosophy and methodology of science as part of logic is not that obvious for other research traditions because of the fact that methodology of science is seen as associated with philosophy rather than with logic. The broad conception of logic employed by the LWS includes semiotics and the methodology of science within logic, not within philosophy (Przełęcki 1971), which is one of the reasons why this treatment of logic is unique. Another distinctive feature of the LWS is the analytical character of philosophical studies – the very reason for introducing the broad conception of logic. For semiotics and the methodology of science are treated in the LWS as disciplines developing universal tools used not only in scientific inquiry, but also in everyday argumentative discourse where analyzing meanings of terms (the skill of applying semiotics) and justifying claims (the skill of applying the methodology of science) are also of use.

[7] Examples of such a broad understanding of the term 'logic' may be found in the works of Antoine Arnauld and Pierre Nicole (*Port Royal Logic*), John Stuart Mill (*The System of Logic. Ratiocinative and Inductive*) and Charles Sanders Peirce (*Collected Papers*) (see the Appendix A in Johnson 2009, p. 39).

2.2. Logical fallacy

One of the consequences of employing this conception of logic is the LWS understanding of logical fallacies as violations of norms of logic broadly understood. These norms of logic in a broad sense are: (1) rules for deductive inference (formal logic), (2) rules for inductive inference (inductive logic), (3) rules for language use as elaborated in semiotics (syntax, semantics and pragmatics), and (4) methodological rules for the scientific inquiry. If these are the 'logical' norms, then consequently there are at least three general types of logical fallacies, i.e. (1) the fallacies of reasoning (also called the fallacies in the strict sense; see Kamiński 1962), (2) fallacies of language use ('semiotic fallacies'), and (3) fallacies of applying methodological rules governing such procedures as defining, questioning or classifying objects ('methodological fallacies').

There are some difficulties with such a broad conception of fallacy. Two major objections against it are:

(a) This conception is too broad because it covers fallacies that are not violations of any logical norms strictly understood. For instance, it would be very hard to point to any logical norm, strictly understood, which would be violated in the case of improper measurement.

(b) The types of fallacies discerned from the viewpoint of the broad conception of logic overlap. For example, the fallacy *post hoc ergo propter hoc* may be classified both as the fallacy of reasoning and as a methodological fallacy. The *fallacy of four terms* may be classified both as a fallacy of reasoning and a semiotic fallacy, because of the fact that it is caused by the ambiguity of terms, and the ambiguity is classified as a semiotic fallacy.

Despite these and other objections, this conception was useful at least in determining a general scope of logicians' interests in identifying fallacies. For example, *affirming the consequent* may be classified as a fallacy of reasoning, amphibology as a semiotic fallacy and vicious circle in defining as a methodological fallacy. This conception of fallacy was briefly presented to show that the conception of logical fallacy accepted by the majority of researchers of the LWS was much broader than that elaborated exclusively from the perspective of formal deductive logic.

2.3. Argument

Another element of the conceptual framework of the LWS is the concept of argument. Since most representatives of the LWS dealt basically with reasoning (e.g. elaborating very detailed classifications of reasoning), the conception of argument is related to the conception of reasoning. For instance, Witold Marciszewski (1991, p. 45) elaborates the definition of argument by associating it with a kind of reasoning performed when the reasoner has an intention of influencing the audience:

> A reasoning is said to be an *argument* if its author, when making use of logical laws and factual knowledge, also takes advantage of what he knows or presumes about his audience's possible reactions.

This definition is treated by Marciszewski as a point of departure for seeking theoretical foundations of argumentation not only in formal logic, but also in philosophy:

> Therefore the foundations of the art of argument are to be sought not only in logic but also in some views concerning minds and mind-body relations including philosophical opinions in this matter.

These general remarks point to the need of analyzing argumentation not only from the formal-logical perspective, but also with bearing in mind the broader context of reasoning performed in any argumentative discourse. One of the ideas that may be used in analyzing arguments in a broader context is the conception of knowledge-gaining procedures. The procedures are treated in the LWS as components of argumentation.

2.4. Knowledge-gaining procedures

From the perspective of the broad conception of logic elaborated in the LWS, arguments may be studied by analyzing and evaluating the main knowledge-gaining procedures (or 'knowledge-creative procedures'; see Jadacki 2009, pp. 98–100) and their results. According to Jadacki (2009, p. 99), in the Polish analytical philosophy the following knowledge-gaining procedures were examined in detail:

(1) Verbalizing, defining, and interpreting;
(2) Observation (the procedure consisting of experience and measurement);
(3) Inference:
  (a) Deduction (proof and testing);
  (b) Induction (statistic inference, 'historical' inference, inference by analogy, prognostics and explanation);
(4) Formulating problems;
(5) Partition, classification, ordering.

When we take *argumentation as a process*, it may be studied as a general procedure consisting of activities as those listed above. When one is dealing with *argumentation as a product*, the results of these procedures are to be analyzed and evaluated. The major research interests in the LWS focused on the following results:

Ad. (1) Concepts and definitions (as the results of verbalizing, defining, and interpreting);
Ad. (2) Observational sentences;
Ad. (3) Arguments understood as constellations of premises and conclusions:
- (a) Deductive inference schemes;
- (b) Inductive inference schemes;

Ad. (4) Questions (as results of the procedure of formulating problems);
Ad. (5) Typologies and classifications (as results of the procedure of ordering).

As Jadacki emphasizes, the procedure which was carefully investigated in the LWS, was inference.[8] So, one of the most interesting results of the knowledge-gaining procedures are arguments understood as constellations of premises and conclusions.

2.5. Logical culture

The conception of logical culture joins two components: (1) advances in the logical studies (i.e. research in logic) are claimed to be applicable in (2) teaching critical thinking skills. According to Tadeusz Czeżowski (2000, p. 68):

> Logical culture, just as any social, artistic, literary or other culture, is a characteristic of someone who possesses logical knowledge and competence in logical thinking and expressing one's thoughts.

Thus, the term 'logical culture' refers both to the knowledge of logic (as applied in using language and reasoning) and to the skill of performing commonsense and scientific reasoning (Koszowy 2004, p. 126–128). Logic broadly understood elaborates tools helpful in sharpening the skills of the logical culture. The general areas of its application are illustrated by Figure 1.

We may here observe that some skills characteristic of the person who possesses logical culture are also substantial for the two normative models in the study of argumentation: (a) an ideal of a critical thinker in the tradition of teaching informal logic in North America, (b) the ideal of a reasonable discussant in a pragma-dalectical theory of argumentation.

[8] This is why classifying various types of inference was one of the crucial tasks for the representatives of the LWS (see Woleński 1989).

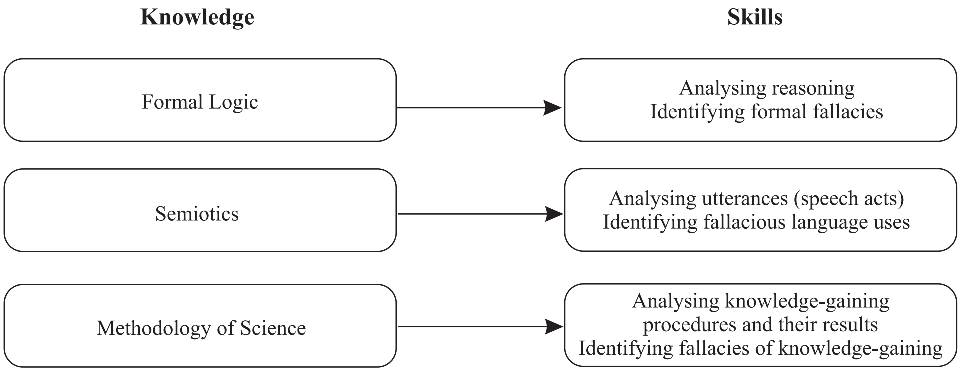

Knowledge
Skills
Formal Logic
Analysing reasoning
Identifying formal fallacies
Semiotics
Analysing utterances (speech acts)
Identifying fallacious language uses
Methodology of Science
Analysing knowledge-gaining procedures and their results
Identifying fallacies of knowledge-gaining

Figure 1. Knowledge and skills of logic (broadly understood)

3. The program of pragmatic logic

The concept of logical culture as presented in the previous section is here a point of departure for introducing Ajdukiewicz's program of pragmatic logic. The term 'logical culture' denotes both knowledge of logic and skills of applying this knowledge in science and everyday conversations, whereas the term 'pragmatic logic' refers to a discipline aimed at describing these skills and showing how to develop them.

The program of pragmatic logic is based on the idea that general (logical and methodological) rules of scientific investigation should be applied in everyday communication. Pragmatic logic is a discipline aimed at applying logic (in a broad sense) in teaching and in everyday language use. So, two basic goals of pragmatic logic are: extending knowledge of logic and improving skills of applying it.

3.1. Subject-matter of pragmatic logic

Pragmatic logic consists of the analyses concerning:

(1) Word use: (a) understanding of expressions and their meaning, (b) statements and their parts, (c) objective counterparts of expressions (extension and intension of terms), (d) ambiguity of expressions and defects of meaning (ambiguity, vagueness, incomplete formulations) and (e) definitions (e.g. the distinction between nominal and real definition, definitions by abstraction and inductive definitions, stipulating and reporting definitions, definitions by postulates and pseudo-definitions by postulates, errors in defining).

(2) Questioning: (a) the structure of interrogative sentences, (b) decision questions and complementation questions, (c) assumptions of questions and

suggestive questions, (d) improper answers, (e) thoughts expressed by an interrogative sentence and (f) didactic questions.
(3) Reasoning and inference: (a) formal logic and the consequence relation (logical consequence, the relationship between the truth of the reason and the truth of the consequence, enthymematic consequence), (b) inference and conditions of its correctness, (c) subjectively certain inference (the conclusiveness of subjectively certain inference in the light of the knowledge of the person involved), (d) subjectively uncertain inference (the conclusiveness of subjectively uncertain inference, logical probability versus mathematical probability, statistical probability, reductive inference, induction by enumeration, inference by analogy, induction by elimination).
(4) Methodological types of sciences: (a) deductive sciences, (b) inductive sciences, (c) inductive sciences and scientific laws, (d) statistical reasoning.

Since inference is one of the key topics of inquiry, in order to show that the program of pragmatic logic has a similar subject-matter to the contemporary study of argumentation, I shall discuss, as an example, Ajdukiewicz's account of the 'subjectively uncertain inference'.

According to Ajdukiewicz (1974, p. 120), a subjectively uncertain inference is the one in which we accept the conclusion with lesser certainty than the premises. It results from the fact that in spite of the premises being true the conclusion may turn out to be false. The instances of this type of inference are such that the strength of categorically accepted premises leads to a non-categorical acceptance of the conclusion. This is illustrated by the following example:

> The fact that in the past water would always come out when the tap is turned on, makes valid – we think – an almost, though not quite, certain expectation that this time, too, water would come out when the tap is turned on. But our previous experience would not make full certainty valid (p. 120).

If we are to be entitled to accept the conclusion with less than full certainty, it suffices if the connection between them is weaker than the relation of consequence is. Ajdukiewicz deals with this kind of reasoning in terms of the probability of conclusion:

> Such a weaker connection is described by the statement that the premisses make the conclusion probable. It is said that a statement B makes a statement A probable in a degree p in the sense that the validity of a fully certain acceptance of B makes the acceptance of A valid if and only if the degree of certainty with which A is accepted does not exceed p (pp. 120–121).

So, 'a statement B makes a statement A probable in a degree p, if the logical probability of A relative to B is *p'*:

$$\mathrm{P}_1(A/B) = p.$$

Furthermore, Ajdukiewicz distinguishes the psychological probability of a statement (i.e. the degree of certainty with which we actually accept that statement) from the logical probability of a statement (that degree of certainty with which we are entitled to accept it). The logical probability is related to the amount of information one possesses at a given stage, because 'the degree of certainty with which we are entitled to accept the statement depends on the information we have'. This claim is in accord with the 'context-dependent' treatment of arguments: argument analysis and evaluation done both in informal logic and in pragma-dialectics depends on the context in which arguments occur. Ajdukiewicz is aware of the fact that evaluating the logical probability of a given statement (P) depends on the actual knowledge of the subject who believes P. The following example confirms this interpretation:

> If we know about the playing card which is lying on the table with its back up merely that it is one of the cards which make the pack used in auction bridge, then we are entitled to expect with less certainty that the said card is the ace of spades than if we knew that it is one of the black cards in that pack (p. 121).

This example gives Ajdukiewicz reasons not to speak about the logical probability of a statement 'pure and simple', but exclusively about the logical probability of that statement relative to a certain amount of information. Ajdukiewicz points to the fact that this relation between the logical probability and the amount of information we possess in a given context is clearly manifested in the following definition of logical probability:

> The logical probability of the statement A relative to a statement B is the highest degree of the certainty of acceptance of the statement A to which we are entitled by a fully certain and valid acceptance of the statement B (ibid.).

This definition is helpful in giving the answer to the question: when is an uncertain inference conclusive in the light of the body of knowledge K? Ajdukiewicz's answer is given in terms of the degree of certainty of the acceptance of the conclusion:

> Such inference is conclusive in the light of K if the degree of certainty with which the conclusion is accepted on the strength of a fully certain acceptance of the premises does not exceed the logical probability of the conclusion relative to the premises and the body of knowledge K (ibid.).

This piece of Ajdukiewicz's account of the subjectively uncertain inference shows that pragmatic logic deals with defeasible reasoning by looking for objective (here 'logical') criteria of evaluating defeasible reasoning. It clearly shows the tendency in pragmatic logic to analyze and evaluate not only deductively valid arguments, but also defeasible ones, as it is done in the contemporary theory of argumentation.[9]

3.2. The goal of pragmatic logic

The goal of pragmatic logic may be extracted from Ajdukiewicz's view on logic treated as a foundation of teaching. This part of Ajdukiewicz's analyses shows how important pedagogical concerns are for the program of pragmatic logic. It also explains why logic is called 'pragmatic'.

For Ajdukiewicz 'the task of the school is not only to convey to the pupils information in various fields, but also to develop in them the ability of correctly carrying out cognitive operations' (Ajdukiewicz 1974, p. 1). This excerpt clearly explains why analysis and evaluation of knowledge-gaining procedures and their results is the main goal of pragmatic logic. If teaching students how to reasonably carry out major cognitive procedures (aimed at achieving knowledge) is one of the main purposes of teaching, then pragmatic logic, understood as a discipline aimed at realizing this goal, has as its theoretical foundation the description of the basic principles of knowledge-gaining procedures.

Ajdukiewicz's crucial thesis is that logic consisting of formal logic, semiotics and the methodology of science constitutes one of the indispensable foundations of teaching. Logical semiotics (the logic of language) 'prepares the set of concepts and the terminology which are indispensable for informing about all kinds of infringements, and indicates the ways of preventing them' (Ajdukiewicz 1974, p. 3). The methodology of science provides 'the knowledge of terminology and precise methodological concepts, and also the knowledge of elementary methodological theorems, which lay down the conditions of correctness of the principal types of cognitive operations, must be included in the logical foundations of teaching' (p. 3). Ajdukiewicz gives an example of a science teacher, who informs students about the law of gravitation and its substantiation by explaining how Newton arrived at the formulation of the law:

[9] In the paper I do not discuss whether defeasible inference is a separate type of inference, as distinct from inductive inference. For the brief overview of the literature on this topic see e.g. Johnson 2009, p. 32.

> When doing so he will perhaps begin by telling pupils that the said law was born in Newton's mind as a *hypothesis*, from which he succeeded to *deduce* the law which states how the Moon revolves round the Earth and how the planets revolve round the Sun, the law which agrees with observations with the *margin of error*. That agreement between the *consequences* of the said hypothesis with empirical data is its *confirmation*, which Newton thought to be sufficient to accept that hypothesis as a *general law* (p. 2).

Thus, according to Ajdukiewicz, the role of the methodology of science in the foundations of teaching is revealed by the fact that crucial terms such as 'hypothesis', 'deduction' or 'verification of hypothesis' are in fact methodological and this is why they are useful in the process of achieving knowledge.

However, pragmatic logic is to be applied not only to scientific research or at school, but also to everyday speech communication. As Ajdukiewicz clearly states, pragmatic logic is not the opposite of formal logic, but both formal and pragmatic logic complement each other. Moreover, pragmatic logic is useful for the teacher, who aims – among other things – at training students to make statements that are relevant, unambiguous and precise, which is 'one of the principal tasks of school education' (Ajdukiewicz 1974, p. 3).

4. Pragmatic logic and argumentation theory: towards bridging the gap

The overview of the concepts of logic, logical fallacy, argumentation, logical culture, pragmatic logic, subjectively uncertain inference and the logical foundations of teaching gives support for the claim that in the LWS and in argumentation theory there are similar tendencies of crucial importance. One of the issues is that the two disciplines share in fact the same subject-matter. To show this in detail, however, would require further inquiry.

Future research should also answer the question of how the main ideas of pragmatic logic may be of use in the analysis, evaluation and presentation of natural language arguments. Research on such applicability of pragmatic logic may focus on the analysis of those components of the program of pragmatic logic which also constitute the subject-matter of argumentation theory. Some similarities may be treated as a point of departure for further systematic exploration of the connection between pragmatic logic and argumentation theory. Figure 2 sketches future lines of inquiry by showing the relation between three research topics in pragmatic logic and in argumentation theory:

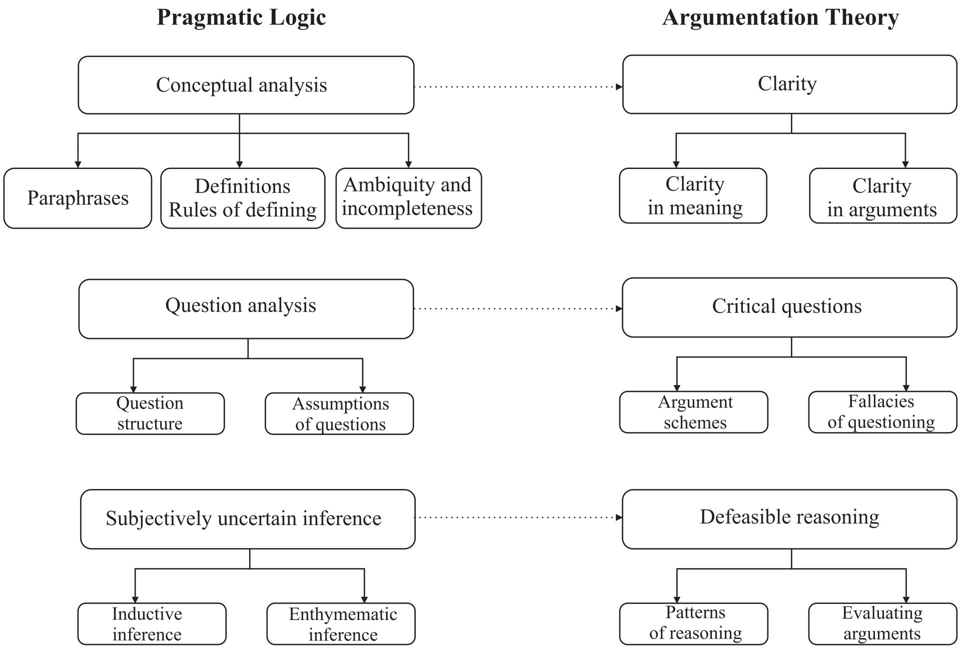


Figure 2. Some connections between pragmatic logic and argumentation theory

Moreover, some fundamental assumptions of pragmatic logic harmonize with methodological foundations (i.e. the subject-matter, goals and methods) of informal logic and pragma-dialectics. The main assumptions of this kind are: (1) the normative concern for reasoning and argumentation and (2) the claim that the power of the study of reasoning and argumentation manifests itself in improving critical thinking skills. As it was shown above, the representatives of the LWS were fully aware of the pragmatic need of studying everyday reasoning. And the ideas of Ajdukiewicz were aimed to be systematically applied to teaching and educational processes. The title given by Ajdukiewicz to one of his papers (Ajdukiewicz 1965: *What can school do to improve the logical culture of students?*) clearly illustrates this approach to teaching logic. In order to stress the pragmatic dimension of this project, it should be mentioned that Ajdukiewicz together with other thinkers of the LWS applied the program in their work as academic teachers. In the *Preface* of his *Introduction to Logic and to the Methodology of Deductive Sciences* (1995) Tarski states:

> I shall be very happy if this book contributes to the wider diffusion of logical knowledge. These favorable conditions can, of course, be easily overbalanced by other and more powerful factors. It is obvious that the future of logic as well as of all theoretical science, depends essentially upon normalizing the political and social relations of mankind, and thus upon a factor which is beyond the control of professional scholars. I have no illusions that the development of logical thought, in particular, will have a very essential effect upon the process of the normalization of human relationships; but I do believe that the wider diffusion of the knowledge of logic may contribute positively to the acceleration of this process. For, on the one hand, by making the meaning of concepts precise and uniform in its own field, and by stressing the necessity of such a precision and uniformization in any other domain, logic leads to the possibility of better understanding between those who have the will to do so. And, on the other hand, by perfecting and sharpening the tools of thought, it makes man more critical – and thus makes less likely their being misled by all the pseudo-reasonings to which they are in various parts of the world incessantly exposed today (Tarski 1995, p. xiii).

The program of pragmatic logic shows that the idea of the necessity of choosing formal and informal analyses of arguments is a false dilemma. For instead of competing with each other, formal logic and pragmatic logic are both legitimate instruments of research and teaching.

R E F E R E N C E S

Ajdukiewicz, K. (1957). *Zarys Logiki* (*An Outline of Logic*). Warsaw: PZWS – Państwowe Zakłady Wydawnictw Szkolnych.

Ajdukiewicz, K. (1965). Co może szkoła zrobić dla podniesienia kultury logicznej uczniów? (What school can do to improve the logical culture of students?). In K. Ajdukiewicz, *Język i poznanie*, t. II (*Language and Cognition*, vol. II) (pp. 322–331), Warsaw: PWN – Polish Scientific Publishers.

Ajdukiewicz, K. (1974). *Pragmatic Logic.* (O. Wojtasiewicz, Trans.). Dordrecht/Boston/Warsaw: D. Reidel Publishing Company & PWN – Polish Scientific Publishers. (Original work published 1965). [English translation of *Logika pragmatyczna*].

Blair, J. A. (2009). Informal logic and logic. *Studies in Logic, Grammar and Rhetoric*, *16* (29), 47–67.

Coniglione, F., Poli, R. & Woleński, J., (Eds.) (1993), *Polish Scientific Philosophy. The Lvov–Warsaw School*, Amsterdam: Rodopi.

Czeżowski, T. (2000). On logical culture. In T. Czeżowski, *Knowledge, Science, and Values. A Program for Scientific Philosophy* (pp. 68–75), Amsterdam/Atlanta: Rodopi.

Eemeren, F. H. van (2009). Strategic manoeuvring between rhetorical effectiveness and dialectical reasonableness. *Studies in Logic, Grammar and Rhetoric*, *16* (29), 69–91.

Eemeren, F. H. van & Grootendorst, R. (2004), *A Systematic Theory of Argumentation: The Pragma-Dialectical Approach*, Cambridge: Cambridge University Press.

Jacquette, D. (2007). Deductivism and the informal fallacies. *Argumentation*, *21*, 335–347.

Jacquette, D. (2009). Deductivism in formal and informal logic. *Studies in Logic, Grammar and Rhetoric*, *16* (29), 189–216.

Jadacki, J. (2009). *Polish Analytical Philosophy. Studies on Its Heritage*, Warsaw: Wydawnictwo Naukowe Semper.

Johnson, R. H. (1996). *The Rise of Informal Logic.* Newport News: Vale Press.

Johnson, R. H. (2009). Some reflections on the Informal Logic Initiative, *Studies in Logic, Grammar and Rhetoric*, *16* (29), 17–46.

Johnson, R. H. & Blair, J. A. (1980). The recent development of informal logic. In J. A. Blair & R. H. Johnson (Eds.), *Informal Logic, The First International Symposium* (pp. 3–28), Inverness, CA: Edgepress.

Johnson, R. H. & Blair, J. A. (1987). The current state of Informal Logic. *Informal Logic*, *9*, 147–151.

Kahane, H. (1971). *Logic and Contemporary Rhetoric.* Belmont: Wadsworth.

Kamiński, S. (1962). Systematyzacja typowych błędów logicznych (Classification of the typical logical fallacies), *Roczniki Filozoficzne*, *10*, 5–39.

Kneale, W. & Kneale, M. (1962). *The Development of Logic*, Oxford: The Clarendon Press.

Koszowy, M. (2004). Methodological ideas of the Lvov–Warsaw School as a possible foundation for a fallacy theory. In T. Suzuki, Y. Yano & T. Kato (Eds.), *Proceedings of the 2nd Tokyo Conference on Argumentation* (pp. 125–130), Tokyo: Japan Debate Association.

Koszowy, M. (2007). A methodological approach to argument evaluation. In F. H. van Eemeren, J. A. Blair, C. A. Willard & B. Garssen (Eds.), *Proceedings of the Sixth Conference of the International Society for the Study of Argumentation* (pp. 803–807), Amsterdam: Sic Sat – International Center for the Study of Argumentation.

Lapointe, S., Woleński, J., Marion, M. & Miskiewicz, W. (Eds.) (2009). *The Golden Age of Polish Philosophy. Kazimierz Twardowski's Philosophical Legacy.* Dordrecht/Heidelberg/London/New York: Springer.

Marciszewski, W. (1991). Foundations of the art of argument. *Logic Group Bulletin*, *1*, Warsaw Scientific Society, 45–49.

Marciszewski, W. (2009). On the power and glory of deductivism. *Studies in Logic, Grammar and Rhetoric*, *16* (29), 353–355.

McCall, S. (1967). *Polish Logic, 1920–1939.* Oxford: Clarendon Press.

Przełęcki, M. (1971). O twórczości Janiny Kotarbińskiej (On Janina Kotarbińska's Output). *Przegląd Filozoficzny*, *5* (72).

Scholz, H. (1930). *Abriss der Geschichte der Logik.* Berlin: Junker and Dünnhaupt.

Tarski, A. (1995). *Introduction to Logic and to the Methodology of Deductive Sciences.* New York: Dover Publications.

The history of the Polish Society for Logic and Philosophy of Science. Retrieved December 17, 2009, from http://www.logic.org.pl

Woleński, J. (1989). *Logic and Philosophy in the Lvov–Warsaw School*, Dordrecht/Boston/Lancaster: D. Reidel Publishing Company.

Woleński J. (1995). Mathematical logic in Poland 1900–1939: people, circles, institutions, ideas, *Modern Logic*, *5*, 363–405.

Woleński, J. (2009). Lvov–Warsaw School. In *Stanford Encyclopedia of Philosophy.* Retrieved June 20, 2010, from http://plato.stanford.edu/entries/lvov-warsaw/

Woods, J., Johnson, R. H., Gabbay, D. M. & Ohlbach, H.-J. (2002). Logic and the Practical Turn. In D. M. Gabbay, R. H. Johnson, H.-J. Ohlbach & J. Woods (Eds.), *Handbook of the Logic of Argument and Inference* (pp. 1–39). Amsterdam: North-Holland.

Marcin Koszowy
Chair of Logic, Informatics and Philosophy of Science
University of Białystok
koszowy@uwb.edu.pl

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 22 (35) 2010

Kazimierz Trzęsicki
University in Białystok, Chair of Logic, Informatics and Philosophy of Science
Stanislaw Staszic College of Public Administration in Bialystok

PHILOSOPHY OF MATHEMATICS AND COMPUTER SCIENCE

Abstract: It is well known fact that the foundation of modern computer science were laid by logicians. Logic is at the heart of computing. The development of contemporary logic and the problems of the foundations of mathematics were in close mutual interaction. We may ask why the concepts and theories developed out of philosophical motives before computers were even invented, prove so useful in the practice of computing. Three main programmes together with the constructivist approach are discussed and the impact on computer science is considered.

Thought has been the father of every advance since time began. 'I didn't think' has cost the world millions of dollars.

Thomas J. Watson, President of IBM

Introduction

The introduction of computers to our daily lives started in the 1950s. At the beginning of the XXI century the information society passed the threshold of unimaginable future. Advances of computer science (CS for short) are amongst the most decisive factors. Three paradigms of CS can be distinguished (Eden 2007):

1. CS is a branch of mathematics,
2. CS is an engineering discipline,
3. CS is a natural (empirical) science.

Re:1. Programs are conceived as a certain kind of mathematical objects and these objects should be investigated according to paradigm of mathematics, i.e. by means of deductive reasoning. *A priori* certain knowledge is the goal of CS. This **rationalist paradigm** is common among theoretical computer scientists.

Re:2. Engineers deal with real objects. They apply theoretical knowledge to develop technology. In the case of CS, real are programs conceived as data. *A posteriori* knowledge about their reliability (practical knowledge) is achieved by empirical testing. This experiential knowledge is only probable. This **technocratic paradigm** is typical for software engineers.

Re:3. The most promising and the most desirable discipline of CS, the AI (**A**rtificial **I**ntelligence) is based on the method of natural sciences. In AI the **scientific paradigm** is prevalent. Programs are conceived as types of mental processes. *A priori* and *a posteriori* knowledge about them is achieved by testifying hypotheses. Induction as well as deduction is applied.

In the following CS will be understood according to the rationalist paradigm, i.e. in the 1st meaning.

CS is essential to almost every human activity, especially science, arts and humanities, even theology. Nevertheless, in the case of mathematics, the expectation that CS will revolutionize it, does not come true. Proof has been considered a fundamental part of the science of mathematical practice since ancient times. Its important status was stressed in Euclidean geometry. The great successes and spectacular achievements of CS are mainly in the field of efficiency, but not in the most important activity of mathematicians: proving new theorems. The situation is well commented by Paul Halmos' statement `http://scidiv.bellevuecollege.edu/math/Halmos.html`:

> Efficiency is meaningless. Understanding is what counts.

and

> I like words more than numbers, and I always did – conceptual more than computational.

However

> I have the world's most intelligent typewriter . . . I spend at least six hours a day on my Macintosh.

For him:

> The computer is important, but not to mathematics.

There are different views of mathematics as a science, its subject and methods. The creation of the set theory by Cantor and the discovery of paradoxes started the discussion of the paradigm of mathematics, i.e. the search for the foundations of mathematics. With the discovery of paradoxes it became evident that mathematics did not live up to the standards of certainty and rigor with which mathematics was over-credited.

The period that started in 1879 with Frege's *Begriffsschrift* and ended in 1931 with Gödel's *Über formal unentscheidbare Satze der Principia Mathematica und verwandter Systeme I* saw the development of three major foundational programmes:

- the logicism of Frege, Russell and Whitehead,
- the intuitionism of Brouwer, and
- Hilbert's formalist and proof-theoretic programme.[1]

Each programme addresses the issues that came to the fore at that time, either attempting to resolve them or claiming that mathematics is not entitled to its status of our most trusted knowledge. To this period, classical period,

- the programme of constructivists mathematics

as a direct legacy of Brouwer's intuitionism can be linked.

Each of the programmes aimed to give a satisfactory firm foundation for mathematics. For logicism:

- mathematics is a branch of logic.

Formalists argued that:

- mathematics is the science of formal systems.

Intuitionists maintained that

- mathematics is about mental constructions.

After Gödel's *Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I* (1931) it was clear that the efforts would not achieve the goal. Though the mathematicians on the whole threw up their hands in frustration and turned away from the philosophy of mathematics, it does not mean that the investigations were fruitless.

We ask about the impact of the programmes on the contemporary CS. We will show that ideas originating in the philosophy of mathematics participated in the rise of CS and have proved very helpful in its development. Philosophy is often thought of as a theoretical activity, which is of little or no practical importance. The influence of philosophy of mathematics on the development of CS demonstrates how this opinion is profoundly mistaken. Philosophical ideas and some kind of philosophical orientation are necessary for many quite practical activities, at least in CS. In the opinion of Gillies (2002):

[1] See (Lindstrom, Palmgren, Segerberg & Stoltenberg-Hansen 2007).

The Wall Street crash of 1929 ushered in the depression of the 1930's. One could say that the Gödel crash of (1931) initiated a period of depression in the philosophy of mathematics. The three main schools all appeared to have failed. Not one had carried out its promise of providing a satisfactory foundation for mathematics. Yet fate was preparing an odd turn of events. In the post-War period the ideas of these philosophical programmes turned out, surprisingly, to be of the greatest possible use in the new and rapidly expanding field of computer science.

1. Logicism and Computer Science

In any case your discovery is very remarkable and will perhaps result in a great advance in logic, unwelcome as it may seem at first glance.

Gottlob Frege
From a letter to Bertrand Russell

1.1. The main ideas of logicism

Logicism tries to found the mathematics on basic logical principles. Its main tenet is that mathematical objects are in fact logical constructions. It was started by Frege (1879, 1884, 1893–1903). To accomplish his goal,[2] Frege defined number in terms of purely logical notions. The existing Aristotelian logic was not adequate for his purposes. So he devised a new kind of formal logic which he published in his *Begriffsschrift* (1879) (literally: concept writing). This is essentially the same as the formal logic taught today.[3] Frege then went on to set up a formal system, and tried to show that the whole of arithmetic could be logically deduced within this system using his definition of number. When Frege was nearing completion of the second volume of his 18 years long work, when he was about to see the success of the project, disaster struck. Bertrand Russell, a young logician, wrote him a letter dated June 16, 1902, in which he pointed out on the contradiction (Russell's paradox) derivable from Frege's basic axioms of logic. Frege's system appeared inconsistent. In his replay of June 22, 1902, Frege wrote (1967, pp. 127–8):

> Your discovery of the contradiction caused me the greatest surprise and, I would almost say, consternation, since it has shaken the basis on which I intended to build arithmetic. It seems, then, . . . that my Rule V . . . is false. . . . I must reflect further on the matter. It is all the more serious since, with the

[2] His aim was not to show that the whole of mathematics was reducible to logic, but only that arithmetic was reducible to logic.

[3] Frege used a curious two dimensional notation, which has been abandoned in favor of the more usual one dimensional manner of writing.

> loss of my Rule V, not only the foundations of my arithmetic, but also the sole possible foundations of arithmetic seem to vanish.

Russell himself and Whitehead aimed to reformulated logicism to avoid inconsistencies. They tried to found mathematics on a basis of *ramified theory of types*[4]. This attempt was not entirely successful: they accomplished the creation of a system in which, in principle, most of the mathematics known then could be formalized, but at the cost of blurring the logical simplicity of Frege's primary ideas. When the three huge volumes of *Principia Mathematica* (1910–1913) were published, it looked as if the logicist programme had been brought to a successful conclusion. However, once again, this apparent success proved short-lived. In (1931) Kurt Gödel showed that, if *Principia Mathematica* were consistent, then there was a true arithmetical statement which could not be proved within the system. Thus reducing arithmetic to the logical system of *Principia Mathematica* or any other similar logicist system is impossible, i.e. any logical system is inherently incomplete.

1.2. How logicism has affected CS?

The logicist's approach to mathematics dwindled,[5] yet many of the ideas of logicism have been essential to CS. Though, in CS logic is no longer viewed as a foundation, but as a tool.[6]

The research of Frege and Russell was inspired by philosophical considerations, and they were both either not influenced at all, or to a negligible extent only, by considerations having anything to do with computing. Nevertheless, their work proved useful in CS later on.

Russell's theory of types was conceived as a basis of his logicist programme, in particular the set theory. Today mathematicians prefer to use ZF (and ZFC), the axiomatic set theory developed by Zermelo, Fraenkel and others.

> Indeed type theory is not taught at all in most mathematics departments. The situation is quite different in computer science departments where courses on type theory are a standard part of the syllabus. This is because the theory of types is now a standard tool of computer science. (Gillies 2002)

[4] A simpler version of logicism, that avoids the reduction principle and still allows the reconstruction of all classical mathematics, was later presented by Frank Plumpton Ramsey (1931). The simple theory of types was developed by Leon Chwistek (1921, 1948). For more see (Linsky 2009).

[5] The idea of logicism to build mathematics on logical base is continued in projects of computer-aided proving, e.g., the computer system `Mizar` `http://mizar.org/`.

[6] The relationships between CS and logic could be described in biological term as symbiotic ones.

Two ideas of logicist programme are of special interest to CS:
- universality of the language of logic, and
- theory of types.

Language of logic is a language of computer

First of all, let us state that any programming language is a formal language. After Gilles we may repeat (2002):

> In fact logic has provided the syntactic core for ordinary programming languages. At an even more fundamental level, the *Begriffsschrift* is the first example of a fully formalized language, and so, in a sense, the precursor of all programming languages.

The kind of language needed to communicate with computer was discussed by Turing (2004, p. 392):

> I expect that digital computing machines will eventually stimulate a considerable interest in symbolic logic and mathematical philosophy. The language in which one communicates with these machines . . . forms a sort of symbolic logic. The machine interprets whatever it is told in a quite definite manner without any sense of humour or sense of proportion. Unless in communicating with it one says exactly what one means, trouble is bound to result. Actually one could communicate with these machines in any language provided it was an exact language, i.e. in principle one should be able to communicate in any symbolic logic, provided that the machine were given instruction tables which would enable it to interpret that logical system. This should mean that there will be much more practical scope for logical systems than there has been in the past.

The difference between the ordinary and formal languages is explained by Frege in the *Begriffsschrift* (Berka & Kreiser 1971, p. 49):

> Das Verhältnis meiner Begriffsschrift zu der Sprache des Lebens glaube ich am deutlichsten machen zu können, wenn ich es mit dem des Mikroskops zum Auge vergleiche. Das Letztere hat durch den Umfang seiner Anwendbarkeit, durch die Beweglichkeit, mit der es sich den verschiedensten Umständen anzuschmiegen weiss, eine grosse Ueberlegenheit vor dem Mikroskop. Als optischer Apparat betrachtet, zeigt es freilich viele Unvollkommenheiten, die nur in Folge seiner innigen Verbindung mit dem geistigen Leben gewöhnlich unbeachtet bleiben. Sobald aber wissenschaftliche Zwecke grosse Anforderungen an die Schärfe der Unterscheidung stellen, zeigt sich das Auge als ungenügend. Das Mikroskop hingegen ist gerade solchen Zwecken auf das vollkommenste angepasst, aber eben dadurch für alle anderen unbrauchbar.
> I believe that I can best make the relation of my ideography to ordinary language clear if I compare it to that which the microscope has to the eye. Because

> of the range of its possible uses and the versatility with which it can adapt to the most diverse circumstances, the eye is far superior to the microscope. Considered as an optical instrument, to be sure, it exhibits many imperfections, which ordinarily remain unnoticed only on account of its intimate connection with our mental life. But, as soon as scientific goals demand greater sharpness of resolution, the eye proves to be insufficient. The microscope, on the other hand, is perfectly suited to precisely such goals, but that is just why it is useless for all others. (1879, p. 6)

Similarly the language of formal logic is suited to the scientific goal of communicating with computers, since this task demands great precision of expression. It is less suited, however, to the task of communicating with other human beings.

It is remarked by Gillies that (2002):

> the idea that different languages are suited to different purposes is already to be found in a reputed saying of the multi-lingual emperor Charles V. He is supposed to have said that he found French the most suitable language for talking to men, Italian for women, Spanish for God, and German for horses. If he had lived today, he could have added that the language of formal logic was the most suitable for talking to computers.

The builders' language-game (Wittgenstein 1953, 2), in which a builder and his assistant use exactly four terms (block, pillar, slab, beam), resemblance our communication with computers when we use the formal language of logic. The rules of language (grammar) are analogous to the rules of games; meaning something in a language is thus analogous to making a move in a game. Thus we arrive at the conclusion that computers do understand the meaning of the symbols they process.

Impact of type theory on CS

In CS, a type system defines how a programming language classifies values and expressions into types, how it can manipulate those types and how they interact. Some form of typing is incorporated into most programming languages. The most obvious application of type theory is in constructing type checking algorithms in the semantic analysis phase of compilers for programming languages. Thus three major areas of using of type theory in CS may be pointed out:

- typed programming languages,
- type-driven program analysis and optimization,
- type-aided security mechanisms

Type theory proponents have the following saying, which proclaims that the design of type systems is the very essence of programming language design:

> Design the type system correctly, and the language will design itself.[7]

Davis recapitulates the situation as follows (1988, p. 322):[8]

> Although the role of a hierarchy of types has remained important in the foundations of set theory, strong typing has not. It has turned out that one can function quite well with variables that range over sets of whatever type. So, Russell's ultimate contribution was to programming languages!

The term "type theory" is used to refer to the formal study of type systems for programming languages, although some limit it to the study of more abstract formalizms such as typed lambda-calculi.

Lambda-calculus

The lambda-calculus emerged in Church's famous paper (1936) showing the existence of an "undecidable problem". The Turing machine was invented later, though independently from the lambda-calculus.[9] Alonzo Church's lambda-calculus and Steven Kleene's recursive functions were arguably more elegant, but it was the mechanical action of Turing's machines that most agreed intuitively about how people calculate:

> These (Turing's) machines are *humans* who calculate. (Wittgenstein 1980, 1096)

It also makes the Turing machines a natural object for studying even more powerful models of computation. Church (1937, pp. 42–43) reviewed Turing's paper comparing the Turing machine to other concepts:[10]

---

[7] See eg. `www.javaworld.com/javaworld/jw-07-2007/jw-07-awscripting1.html?page=3`.

[8] Cf. (Gillies 2002).

[9] As regards the tricky question of priority, Church wrote:

> In an appendix, the author [i.e. AMT] sketches a proof of the equivalence of 'computability' in his sense and 'effective calculability; in the sense of the present author [i.e. Church's definition using the lambda-calculus.] The author's result concerning the existence of uncomputable sequences was also anticipated, in terms of effective calculability, in the cited paper [i.e. Church's paper]. His work was, however, done independently . . . .

[10] Church omitted Post's concept of binormality. For Post (1965, pp. 408, 419) himself it was only a working hypotheses:

> As a matter of fact, there is involved here the equivalence of three different notions: computability by a Turing machine, general recursiveness in the sense of Herbrand-Gödel-Kleene, and the λ-definability in the sense of Kleene and the present reviewer. Of these, the first has the advantage of making the identification with effectiveness in the ordinary (not explicitly defined) sense evident immediately – i.e., without the necessity of proving preliminary theorems. The second and third have the advantage of suitability for embodiment in a system of symbolic logic.

Lambda-calculus was applied to answer Hilbert's *Entscheidungsproblem*, but its idea originated in Russell's type theory. It was conceived by Church (1932, 1933) as part of a general theory of functions and logic, intended as a foundation for mathematics to develop the logicist position of Russell and Whitehead (1910–1913). Church (1940) introduced functions as primitive objects and the lambda-calculus notation was used in his elegant formulation of the simple type theory (Church 1940).

We may point to at least two ideas of *CS* that the lambda-calculus influenced: the design of the LISP programming language and functional programming languages in general.[11] There are two different families of systems of lambda-calculus: one elaborated by Curry (1934) and another by Church (1940). They correspond to two paradigms in programming (Barendregt 1992, p. 119). In the case originated in Curry's paper (1934) program is written without typing at all, i.e. is implicitly typed. ML (MetaLanguage) is such a language, (Milner 1984). Explicite typing corresponds to the version originated in Church's paper (1932, 1933). In this case, the program is written together with its type. ALGOL 68 and PASCAL are examples of such a language.

> [...] for full generality a complete analysis would have to be made of all the possible ways in which the human mind could set up finite processes for generating sequences. [...] we have to do with a certain activity of the human mind as situated in the universe. As activity, this logico-mathematical process has certain temporal properties; as situated in the universe it has certain spatial properties.

[11] Let us mention a curiosity concerning the lambda-calculus. Under the name *The Knights of the Lambda Calculus* is hidden a semi-mythical organization of wizardly LISP and Scheme hackers. LISP and Scheme are based on lambda-calculus, hence the phrase "lambda-calculus" is used in the name of the group. The word "knights" refers to the Knights Templar. It mostly only exists as a hacker culture in-joke. See `http://www.catb.org/~esr/jargon/html/K/Knights-of-the-Lambda-Calculus.html`. We may mention the Church encoding, too. In mathematics, Church encoding is a means of embedding data and operators into the lambda-calculus. The method is named for Alonzo Church, who first encoded data in the lambda-calculus this way.

LISP

LISP (for **LIS**t **P**rocessor) is a family of programming languages with a distinctive, fully parenthesized syntax. LISP was invented by John McCarthy in 1958:

> The system was designed to facilitate experiments with a proposed system called the Advice Taker, whereby a machine could be instructed to handle declarative as well as imperative sentences and could exhibit "common sense" in carrying out its instructions. (McCarthy 1960, p. 184)

LISP is the second-oldest high-level programming language in widespread use today; only FORTRAN is older. Today, the most widely known general-purpose LISP dialects are COMMON LISP, Scheme, and their derivative DYLAN. LISP quickly became the favored programming language for AI research. LISP introduced many features now found in functional languages, though LISP is technically a multi-paradigm language. It pioneered many ideas in computer science, including tree data structures, automatic storage management, dynamic typing, and the self-hosting compiler.

Functional programming

It is proved that all recursive functions can be represented in the lambda-calculus (Kleene & Rosser 1935). Moreover, exactly the functions computable by a Turing machine can be represented in the lambda-calculus (Turing 1937). Computable functions as expressions in the lambda-calculus give rise to the so-called functional programming (Thompson 1991, Barendregt 1992, p. 118). Lambda-calculus forms the basis of almost all functional programming languages. These languages can be viewed as elaborations of the lambda-calculus. Functional programming is a programming paradigm that treats computation as the evaluation of mathematical functions and avoids state and mutable data. It emphasizes the application of functions, in contrast to the imperative programming style, which emphasizes changes in state.

"Haskell", Curry's first name is used to dub a programming language that is rooted in his work (1934, 1958) and his intellectual descendants (de Bruijn 1968, Howard 1980) observations that:

> a proof is a program; the formula it proves is a type for the program.[12]

In Haskell, "a function is a first-class citizen" (Burstall 2000) of the pro-

[12] The Curry–Howard correspondence is the direct relationship between computer programs and proofs in constructive mathematics. Some researchers tend to use the term Curry–Howard–de Bruijn correspondence in place of Curry–Howard correspondence.

gramming language. Haskell is an advanced purely functional programming language, with non-strict semantics and strong static typing.[13]

Javascript, one of the most widely employed languages today, incorporates functional programming capabilities.

Unlambda

The combinatory logic was founded by Schönfikel's article *Über die Bausteine der mathematischen Logik* (1924). It is more abstract than lambda-calculus and preceded it in invention. Both, combinatory logic and lambda-calculus, are theoretically equivalent (Curry, Feys & Craig 1958). A lambda expressions can be easily transformed into combinator expressions, and combinator reduction is much simpler than lambda reduction. Combinatory logic has been used to model some non-strict functional programming languages and hardware. Combinatory logic is used in some esoteric languages including Unlambda.[14] A functional language based on the combinatory logic is a language without variables or lambda expressions. Unlambda is of some theoretical interest.

Lambda calculus and combinatory logic are now studied as idealized programming languages.

2. Formalizm and Computer Science

Aus dem Paradies, das Cantor uns geschaffen hat, soll uns niemand vertreiben können.
No one will drive us from the paradise which Cantor created for us.

David Hilbert
Über das Unendliche

2.1. The main postulates of formalizm

The formalist philosophy of mathematics known as Hilbert's Program was developed by David Hilbert in order to "dispose of the foundational questions in mathematics once and for all". The concept of formal system was taken from the logicists. But for Hilbert each branch of mathematics could be based on different formal system,[15] which should be:

[13] For more see `http://www.haskell.org/`.

[14] For the official Unlambda distribution, including several (other) implementations and documentation of the language, see the Unlambda Homepage by David Madore: `http://www.madore.org/~david/programs/unlambda/`.

[15] Hilbert also realized that axiomatic investigations required a well worked-out logical formalizm. He, with the assistance of Bernays and Behmann, made significant new contributions to formal logic.

1. consistent, i.e. free from contradictories;
2. complete, i.e. each theorem (true statement) has to be provable in it;
3. decidable, i.e. in case of any proposition the answer about its truth or about its falsity should be available with the help of "finitary" methods.[16]

If reading or writing a formalized text is concerned, it matters little whether this or that meaning is attached to the formulas. The only important point is the correct observance of the rules of syntax.[17] Hilbert told (in a railway station restaurant):

> Man muß jederzeit an Stelle von 'Punkte, Geraden, Ebenen' 'Tische, Stühle, Bierseidel' sagen können.
> It must be possible to replace the words 'point, line, plane' with 'table, chair, beer mug'.

Properties of formalized mathematics should be proven in the metalanguage via finitistic methods ('metamathematica').[18] The only part of mathematics to have meaningful content is finitary mathematics, that is, the analysis of concrete discrete objects and their decidable properties. The consistency coincides with satisfiability, that is, the existence of a mathematical universe in which all the derivable statements are true (Hilbert 1926, p. 370):

> So in recent times we come upon statements like this: even if we could introduce a notion safely (that is, without generating contradictions) and if this were demonstrated, we would still not have established that we are justified in introducing the notion. Is this not precisely the same objection as the one formerly made against complex numbers, when it was said that one could not, to be sure, obtain a contradiction by means of them, but their introduction was nevertheless not justified, for, after all, imaginary magnitudes do not exist? No, if justifying a procedure means anything more than proving its consistency, it can only mean determining whether the procedure is successful in fulfilling its purpose.

Kurt Gödel (1930) demonstrated that it is enough to prove the consistency of a theory, to be sure that it is meaningful (that is, it has a model). Since

[16] The problem of "decidability of every mathematical question" traces back to Hilbert's (1900) address.

[17] For Hilbert mathematics is the science of formal systems, consisting of a well-described syntax and a derivation criterium. But mathematics itself is no formal system; it only studies formal systems.

[18] In Weyl's opinion (1967, p. 483): Hilbert "...succeeded in saving classical mathematics by a radical reinterpretation of its meaning without reducing its inventory, namely by ...transforming it in principle from a system of intuitive results into a game with formulas that proceeds according to fixed rules."

finitary mathematics is the only solid ground to start from, the proof of consistency must be carried out by finitary means. Gödel (1931) discovered that a proof of consistency for a theory that – roughly speaking – contains arithmetic, always requires the use of a more powerful, and thus less reliable theory. Infinitary mathematics cannot be proved consistent by finitary means. Gödel's result showed that there can be no absolute consistency proof of all of mathematics; hence work in proof theory after Gödel concentrated on relative results. Starting with the work of Gerhard Gentzen in the 1930s, the Relativized Hilbert Programs[19] have been central to the development of proof theory.

2.2. Contribution of formalizm to *CS*

Turing machine

One of the important problems that was raised by Hilbert's Program was the question if there was a mechanical procedure for separating mathematical truths from mathematical falsehoods. It is the so-called *Entscheidungsproblem*.[20] David Hilbert's formulation of the *Entscheidungsproblem*, i.e. classical problem of decidability, produced a plethora of ideas that – in particular – gave rise to *CS* and is still abundant in opening new horizons to computer science, mathematics and philosophy.

In the 20th century, the idea of computation, the basic idea of *CS*, was developed in connection with the problem of decidability. The Church–Turing thesis is a statement that characterizes the nature of computation. The thesis (conjecture) claims that an intuitive notion of calculus is adequate to the notion of the Turing machine[21] (or equivalent notions such as: recursive functions, the lambda-calculus by A. Church, the canonical systems by E. Post, the normal algorithms by A. A. Markov, the Minsky machines, the Kolmogorov algorithms, etc.). The Church–Turing thesis cannot be formally proven. Despite this fact, it now has near-universal acceptance.

Alan Turing's work on the *Entscheidungsproblem* followed Kurt Gödel's work on the incompleteness theorems, and the notion of general purpose computers that came from this work, was of fundamental importance to the designers of the computer machinery in the 1940s. The Turing machine is

[19] For more, see (Murawski 1999, 4.4. Relativized Hilbert's Program vs. Reverse Mathematics).

[20] For more about *Entscheidungsproblem* and its implications see (Trzęsicki 2006).

[21] By Turing the machine was named a logical computing machine, or "*a*-machine" (automatic machine). It has subsequently become known as the "Turing Machine". For the first time this term was used by Church (1937).

a theoretical device that computes all the functions that are computable in any reasonable sense. Conversely, it is also believed that if a computation cannot be performed by the Turing machine, then it cannot be computed at all. In other words, the thesis states that all effective computational models are equivalent to, or weaker than, the Turing machine (Shoenfield 1991, p. 26). It has long been assumed that the Turing machine is able to do all that computers equipped with recursive algorithms do and all that can be done by the Turing machine may be executed by such computers (if they have enough time[22]). Thus the Turing-Church thesis states that any computation solvable by a precisely stated set of instruction (an algorithm) can be run on the Turing machine or a digital process computer.[23]

The idea of computability executed by the Turing machine is very fertile and has originated many ideas such as von Neumann's classical computer. In Turing's paper, there appeared very fruitful notions of "input-output", "memory", "kompiler/interpreter", "finite-state machine", "coded program", and "algorithm". Turing's definition of computability remains a classic paper in the elucidation of an abstract concept into a new paradigm.

In recent years, the number of people who maintain that the Turing machine cannot capture the entire spectrum of applications of computers has been growing. There are important constraints on the ability of the Turing machine. Generally speaking, there are problems related to tractability, commensurability and computability. Moreover, some of these limitations do not concern physical restrictions. There are well-stated problems that are not computable by the Turing machine. Thus the question arises if there are possible devices which can compute more than the Turing machine.

The theory of computation became an independent academic discipline and was separated from mathematics. It is one of the disciplines of CS. Computability theory is closely related to recursion theory. Recursion theory is not restricted to consider models of computation that are reducible to the Turing model.

For formalists mathematical reasoning as captured in a formal system means manipulating strings of symbols. Manipulating symbols is what computers do. Thus the formalists approach encourages projects of systematic encoding in computer-readable format of mathematical knowledge so as to facilitate automated proof checking of mathematical proofs and the use

[22] The notion of the "speed of computation" makes little sense in the classical understanding of mathematics. Its importance has been recognized with the advent of modern computers and their applications, especially such that need to be accomplished in real time.

[23] For more about the Turing-Church Thesis, its history and implication, see (2009).

of interactive theorem proving in the development of mathematical theories and computer software.[24] Moreover, if human knowledge could be expressed in a formal language, it would be possible – as some researchers predicted in the 1950s and 1960s – an artificial intelligence, a machine that "thinks"[25]. Today AI is a flourishing branch of *CS*.

Logic, mathematics, and *CS* are strongly related to each other. They are the only genuine formal sciences in the sense that they can be carried out purely formally. Programs are of course nothing more than rather large and very complicated formal objects.

The consequences of Hilbert's program, unexpected by himself, are summarized by Chaitin (2004):

> As I said, formal systems did not succeed for reasoning, but they succeeded wonderfully for computation. So Hilbert is the most incredible success in the world, but as technology, not as epistemology. [...] Hilbert's idea of going to the limit, of complete formalization, which was for epistemological reasons, this was a philosophical controversy about the foundations of mathematics – are there foundations? And in a way this project failed, as I've explained, because of the work of Gödel and Turing. But here we are with these complete formalizations which are computer programming languages, they're everywhere!

3. Intuitionism and Computer Science

> The question where mathematical exactness does exist, is answered differently by the two sides; the intuitionist says: in the human intellect, the formalist says: on paper.
>
> L. E. J. Brouwer (Brouwer 1913, p. 56)

3.1. Fundamentals of intuitionism

Intuitionism originates with the Dutch mathematician L. E. J. Brouwer.[26] The main ideas of his philosophy were formulated in his doctoral dissertation *Over de Grondslagen der Wiskunde* (1907). Brouwer's philosophy of mathematics is embedded in a general philosophy, the essentials of which are found already in (Brouwer 1905). Intuitionism as a philosophy of

[24] Because of their close connection with computer science, this idea is also advocated by some mathematical intuitionists and constructivists.

[25] "Think" is the motto of IBM.

[26] For more about history of intuitionism see (van Atten, Boldini, Bourdeau & Heinzmann 2008).

mathematics is based on the rejection of the Platonic notion of the existence of mathematical objects. For Brouwer (1975) mathematics is not formal; the objects of mathematics are mental constructions in the mind of the (ideal) mathematician. Only the thought constructions of the (idealized) mathematician are exact. Mathematics is a stand-alone activity concerning mental constructions according to self-evident rules, independent of experience in some outside reality, and mathematics is in principle also independent of language:

> The first act of intuitionism completely separates mathematics from mathematical language, in particular from the phenomena of language which are described by theoretical logic, and recognizes that intuitionist mathematics is an essentially languageless activity of the mind. (Brouwer 1952, pp. 141–2)

Numbers and functions are mental constructions and mathematical theorems express the capacity of the human mind to perform certain operations on these constructions. Brouwer accepted only one basic notion: time,[27] mathematically seen as the real line. For set theorists the real numbers are set theoretical construction, e.g. Dedekind cuts, but for Brouwer the reals are there without the need of any further qualification. Objects of mathematics have meaning only inside the human mind. Mathematics is the study of mental mathematical constructions realized by a creative subject.

Mathematics does not depend on logic; on the contrary, logic is part of mathematics. Mathematics precedes logic: the logic we use in mathematics grows from mathematical practice, and is not something *a priori* given before mathematical activity can be undertaken. For intuitionists the truth of a proposition must be constructive: it must rest on proof (a certain kind of mental construction). This led to the rejection of some principles of the classical logic. In (1908) Brouwer explicitly noted that intuitionism required a different logic. The principle of the *excluded middle*[28] and the principle of *double negation elimination* were rejected. A question has an answer when we find it; if we are unable to construct a proof or a confutation of a proposition, we cannot assume that:

[27] For some protagonists of the role of insight in mathematics intuition of space was the source of mathematical concepts, e.g. Poincaré rejected purely logical or linguistic descriptions as the only source for mathematics and stressed the role of geometric insight.

[28] Propositions restricted to a finite collections are still regarded as being either true or false, as they are in classical mathematics, but this bivalence does not extend to propositions which refer to infinite collections. L. E. J. Brouwer viewed the law of the excluded middle as abstracted from finite experience, and then applied to the infinite without justification. To him the law of the excluded middle was tantamount to assuming that every mathematical problem has a solution.

- it must be either that a proposition is true or that a proposition is false;
- the truth of doubled negation of a proposition is not the sufficient reason of the truth of that proposition.[29]

The principle of impredicativity[30] that allows the construction of a new object of a certain class by referring to the whole class as a completed whole is not accepted. It means that the principle of the classical first-order logic that general proposition is a sufficient reason of a particular (existential) proposition: $\forall x P(x) \rightarrow \exists x P(x)$ is rejected, too. The intuitionistic interpretation (of truth) of propositions gives rise to a non-classical logic.

In (1930), Brouwer's most famous pupil, Arend Heyting, published the first Hilbert style formalization of the logic used by intuitionists which so clearly characterized the logic that it has become universally known as the axioms for intuitionistic logic (1930, 1956). This formal system captured the informal intuitionistic comprehension of the connectives and quantifiers. The standard informal interpretation of logical operators in intuitionistic logic is the so-called proof-interpretation or the Brouwer–Heyting–Kolmogorov interpretation, (BHK for short).[31]

Intuitionism as the only of the three major programs was not affected by Gödel's incompleteness theorems. Nevertheless, the intuitionistic mathematics turned out to be more involved and intricate than standard mathematics, and, as a result, it was rejected by most mathematicians as just too complicated to be acceptable.

Traditionally, some mathematicians have been suspicious, if not antagonistic, towards intuitionism. David Hilbert forcefully expressed his attitude. In *Die Grundlagen der Mathematik* he wrote (1928)[32]:

> I am astonished that a mathematician should doubt that the principle of excluded middle is strictly valid as a mode of inference. I am even more astonished that, as it seems, a whole community of mathematicians who do the same has so constituted itself. I am most astonished by the fact that even in mathematical circles the power of suggestion of a single man, however full of

[29] The converse holds: the truth of a proposition is the sufficient reason of the truth of its double negation.

[30] See (1906, 1916, 1925) and Weyl (1918).

[31] The meaning of a connective is described by explaining what is to be regarded as a proof of a compound statement assuming one knows what counts as proof of each of its arguments. Such an interpretation is implicit in Brouwer's writings (e.g. (1908, 1924), and has been made explicit by Heyting (1934) for predicate logic, and by A. N. Kolmogorov (1932) for propositional logic.

[32] See `www.marxists.org/reference/subject/philosophy/works/ge/hilbert.htm`

temperament and inventiveness, is capable of having the most improbable and eccentric effects.[33]

Many mathematicians probably shared Bourbaki's view (Bourbaki 1991, p. 38):

> The intuitionistic school, of which the memory is no doubt destined to remain only as an historical curiosity, would at least have been of service by having forced its adversaries, that is to say definitely the immense majority of mathematicians, to make their position precise and to take more clearly notice of the reasons (the ones of a logical kind, the others of a sentimental kind) for their confidence in mathematics.

Intuitionism, in Brouwer's form, doesn't lend itself to computer implementation, since Brouwer rejected the idea that mathematical reasoning is a formal activity. He even refused to use logical symbolism, preferring, whenever possible, to express himself in natural language. The human mind, he contended, is a *creative subject*, that can perform constructions that elude any formal process. He even contested that mathematics is based on logic. Thus, intuitionism is radically opposed to both formalizm and logicism. Nevertheless, intuitionism gave rise to critical review of

- the notion of an (existence) proof,
- the notion of a computable function.

According to Troelstra (1999)

> Ideas developed in the context of intuitionism turned out to have a relevance which transcends the original setting.

In the contemporary philosophy of mathematics with the posthumous publication of Wittgenstein's *Remarks on the Foundations of Mathematics* in 1956, Brouwer's ideas again sparked a lively interest among philosophers.

With the publication, in 1967, of Errett Bishop's *Foundations of Constructive Analysis* perception of Brouwer's legacy changed dramatically (Bridges & Reeves 1997). The intuitionist philosophy of mathematics gave rise to different streams of constructivism and due to that fact intuitionism has an impact on *CS*. Constructivism has many connections to *CS* and has found numerous applications in *CS* (Troelstra & van Dalen 1988, Beeson 1985).

[33] Brouwer struggled, perhaps too aggressively, to overcome sometimes harsh polemic against him and the antipathy of Hilbert and his followers to intuitionistic mathematics. See (van Stigt 1990) for more details of the history of that period. See also `http://en.wikipedia.org/wiki/Brouwer%E2%80%93Hilbert_controversy`

3.2. Constructivism in computer science

In mathematics everything is algorithm and nothing is meaning; even when it doesn't look like that because we seem to be using words to talk about mathematical things. Even these words are used to construct an algorithm.

Ludwig Wittgenstein (1974, p. 468)

When Brouwer proposed its new foundation of mathematics, the modern computer had not appeared yet. But soon later, the first cornerstones of theoretical computer science began to be laid by Turing (1936–37), Church (1932, 1941) and Kleene (1936, 1952). The convergence of some of the basic ideas of intuitionism and computer science became soon evident. The starting point can be identified in the work of Arend Heyting (1956), Brouwer's student.

Constructivism

Constructivism[34] is often identified with intuitionism, although intuitionism is not only a constructivist program.

Constructivism emerges in the final quarter of the 19th century, and may be regarded as a reaction to the rapidly increasing use of highly abstract concepts and methods of proof in mathematics. In the course of time, the focus of activity in constructive mathematics has shifted from Brouwerian intuitionism to Markov's constructivism, then to Bishop's constructivism. Together with the growth of interest in CS, recursive mathematics attracted more researches. The subject is still flourishing.[35]

Characteristic for the constructivist trend is the insistence that mathematical objects are to be constructed (mental constructions) or computed. Leopold Kronecker and Henri Poincaré may be described as the first conscious constructivists. For Kronecker, only the natural numbers were 'God-given'; all other mathematical objects ought to be explained in terms of natural numbers (at least in algebra).[36] From the philosophical angle,

[34] Constructivism is also the label given to a set of theories about learning which fall somewhere between cognitive and humanistic views. It is a theory, which claims that students construct knowledge rather than merely receive and store knowledge transmitted by the teacher. In CS is practised by, e.g. Ben-Ari (2001). In the case of mathematics, and – by analogy – CS could be based on social constructivist philosophy of mathematics:

In fact, whether one wishes it or not, all mathematical pedagogy, even if scarcely coherent, rests on a philosophy of mathematics. (Thom 1973, p. 204)

[35] About history of constructivism till ca. 1965, see (Troelstra 1991).

[36] The thought of Kronecker is summarized in his famous saying: Die ganzen Zahlen hat der Liebe Gott gemacht, alles andere ist Menschenwerk (God created the integers, all else is the work of man).

constructivism asserts that it is necessary to find (or "construct") a mathematical object to prove that it exists. The proof of the existence of a mathematical object is tied to the possibility of its construction. Even though most mathematicians do not accept the constructivist's thesis, that only mathematics performed on the basis of constructive methods is sound, constructive methods are increasingly of interest on non-ideological grounds. Although investigations in the area of constructive logic and mathematics do not belong to the main stream of research, the concepts and techniques of constructive systems play a significant role in theoretical computer science and artificial intelligence, e.g. the notion of 'formulas-as-types'[37] and Martin-Löf-style type theories. The most celebrated contemporary mathematical constructivism, at least before 1967, was Brouwerian (Dutch or neo-) intuitionism.

Because the meaning "to construct" is not clearly defined, that has led to many forms of constructivism. In particular – and this is of interest to us – to a refinement of intuitionistic mathematics based on the reinterpretation of Brouwer's *mental constructions* as *computer programs*. From constructive proofs one can, at least in principle, extract algorithms that compute the elements and simulate the constructions whose existence is established in the proof.

Almost all constructivists accept the same logic, namely that axiomitized by Heyting. Intuitionistic connectives and quantifiers have computational interpretations. A surprising fact is that the rejection of the principles of excluded middle and impredicativity had very important consequences for computability. It is also possible to reconcile intuitionism with its competitors: the core ideas of formalizm, logicism and intuitionism can be synthesized in constructive foundation of mathematics.

Recursive realizability

Stephen Cole Kleene[38] with his notion of *recursive realizability* (Kleene 1945, Kleene 1952) thoroughly investigated the nature of intuitionism. Recursive realizability establishes a connection between the notion of computable (recursive) function and intuitionistic logic. Every proposition is

[37] The expression "formulas-as-types" is widely widely used but its precise meaning is not sharply delimited.

[38] In 1990 he was awarded the National Medal of Science.

To precision, to the clear definition of the notion of constructable and recursive functions, and to the application of these notions to intuitionism, in computer science, and in logic generally. `http://www.nap.edu/html/biomems/skleene.pdf`

For more about history of realizability, see (Oosten 2000).

interpreted as a natural number, by using the encoding of pairs of natural numbers and of recursive functions as natural numbers. Proofs become themselves mathematical objects (natural numbers) about which we can reason. Such objects as natural numbers can be implemented on a computer. The system Automath (**auto**mating **math**ematics) devised by Nicolaas Govert de Bruijn in and after 1967 is the first such an implementation (de Bruijn 1970, de Bruijn 1980).[39] This project can be seen as the predecessor of type theoretical proof assistants such as the well known Nuprl and Coq.

Recursive constructive mathematics

In the former Soviet Union, A. A. Markov,[40] who initiated the approach around 1950, and his collaborators developed what was essentially recursive mathematics using intuitionistic logic. The school maintained that mathematics (Aberth 1980, p. 2):

> may be informally described as an analysis wherein a computation algorithm is required for every entity employed. The functions, the sequences, even the numbers of computable analysis are defined by means of algorithms. In this way definition and evaluation are made inseparable.

New constructivism

New constructivism (or Bishop constructivism) originated with Errett Albert Bishop's influential *Foundations of Constructive Analysis* (1967. In *A Constructivist Manifesto* (1967, Chapter 1, p. 2) we read:

> Mathematics belongs to man, not to God. We are not interested in properties of the positive integers that have no descriptive meaning for finite man. When a man proves a positive integer to exist, he should show how to find it. If God has mathematics of his own that needs to be done, let him do it himself.

Bishop insisted that new constructive mathematics capture the 'numerical meaning' of mathematical claims.

The numerical meaning was to be cashed out in predicting the outcomes of performable computations on the integers, computations that come from realizing constructive theorems as computer programs. Bishop encouraged

[39] For more, see `http://www.win.tue.nl/automath/`, `http://www.cs.ru.nl/~freek/aut/`. For an overview of systems implementing "mathematics in the computer" see `http://www.cs.ru.nl/~freek/digimath/bycategory.html#tacticprover`.

[40] An excellent reference for the work of the Markov School is (Kushner 1985). See also `http://logic.pdmi.ras.ru/Markov/Markov.html`.

a comparison between formalizations of new constructivism and high-level specification and programming languages, ones whose proofs could be compiled into implementable code. This general idea of 'programming constructive mathematics' has since been adopted by a number of computer scientists.

Martin-Löf's Constructive Type Theory

Intuitionistic type theory, or constructive type theory, or Martin-Löf type theory is a logical system and a set theory based on the principles of mathematical constructivism. It was developed by Per Martin-Löf in the early 1970s, as a constructive foundation for mathematics. A similar approach has been developed by Constable (1971) and Beeson (1983).

This theory is based on extended[41] Curry–Howard isomorphism between propositions and types to include disjunction, existential quantification, and induction: a proposition is represented as a type: namely, the type of proofs of the proposition (Martin-Löf 1998, Martin-Löf 1982*a*, Martin-Löf 1984*b*).[42] The types of Martin-Löf type theory play a similar role as sets in set theory but functions definable in type theory are always computable.

The possibility of using constructive mathematics as a basis for programming was suggested in (Bishop 1970) by Bishop. Per Martin-Löf was the first who explicitly and directly used intuitionistic logic in connection with computer science[43]. Intuitionistic logic in the form of Martin-Löf's theory of types (1982*a*) is well suited as a theory for program construction since it is possible to express both specifications and programs within the same formalizm. It provides a complete theory of the process of program specification, construction, and verification, i.e. theory of programs correctness.

Furthermore, the proof rules can be used to derive a correct program from a specification as well as to verify that a given program has a certain property. Type theory is more than a programming language and it should not be compared with programming languages, but with formalized programming logics.[44]

[41] By introducing dependent types, that is types which contain values.

[42] Some of the basic ideas were already present in (Scott 1970), who was inspired by de Bruijn's work on Automath

[43] His first paper on the subject *Constructive Mathematics and Computer Programming* (1982*a*), later reprinted as (Martin-Löf 1984*a*, Martin-Löf 1985), was presented at the 6th International Congress for Logic, Methodology and Philosophy of Science in Hannover in August 1979. This paper was preceded by the first expositions of Martin-Löf 's ideas in (Martin-Löf 1982*b*) and in some lecture notes, made by Sambin during a course in 1980, published as (Martin-Löf 1984).

[44] NordstromPeterssonSmith1990

For Martin-Löf (1982*a*):

> the whole conceptual apparatus of programming mirrors that of modern mathematics (set theory, that is, not geometry) and yet is supposed to be different from it. How come? The reason for this curious situation is, I think, that the mathematical notions have gradually received an interpretation, the interpretation which we refer to as classical, which makes them unusable for programming. Fortunately, I do not need to enter the philosophical debate as to whether the classical interpretation of the primitive logical and mathematical notions ... is sufficiently clear, because this much at least is clear, that if a function is defined as a binary relation satisfying the usual existence and unicity conditions, whereby classical reasoning is allowed in the existence proof ... then a function cannot be the same thing as a computer program ... Now it is the contention of the intuitionists... that the basic mathematical notions, above all the notion of function, ought to be interpreted in such a way that the cleavage between mathematics, classical mathematics, that is, and programming that we are witnessing at present disappears. In the case of the mathematical notions of function and set, it is not so much a question of providing them with new meanings as of restoring old ones ...

One major advantage of Martin-Löf's formal approach to constructive mathematics is that it greatly facilitates the extraction of programs from proofs. This has led to serious work on the implementation of constructive mathematics (Martin-Löf 1982*a*, Constable, Allen, Allen, Bromley, Cleaveland, Cremer, Harper, Howe, Knoblock, Mendler, Panangaden, Smith, Sasaki & Smith 1986, Hayashi & Nakano 1988).

A number of popular computer-based proof systems are based on Martin-Löf type theory, for example NuPRL,[45] LEGO,[46] Coq,[47] Agda[48] and Twelf.[49]

The link between constructive mathematics and programming holds great promise for the future implementation and development of abstract mathematics on the computer. Martin-Löf 's work made it evident that: *computer science is equivalent to completely formalized mathematics that uses only intuitionistic logic.*

---

[45] See `http://www.cs.cornell.edu/info/projects/nuprl/book/doc.html`

[46] `http://www.dcs.ed.ac.uk/home/lego/`

[47] `http://coq.inria.fr/`

[48] See `http://www.cs.chalmers.se/~catarina/agda/`

[49] `http://www.cs.cmu.edu/~twelf/`

Constructive Reverse Mathematics

Reverse mathematics originates with Friedman's paper *Some systems of second order arithmetic and their use* (1975). The research programme is aiming to classify mathematical theorems according to their equivalence to one of a small number of set-theoretic principles. A programme of reverse mathematics based on intuitionistic logic was initiated independently by Veldman (2005) and Ishihara (2005, 2006).

Social constructivism

Although it seems that social constructivism has no direct impact on CS, to name all the contemporary kinds of constructivism, we will point out some its theses.

The most significant bases of the social constructivist theory were laid down by Vygotsky (1962). For social constructivists all knowledge, including mathematics is actually socially constructed in support of particular values and understandings. Mathematics as a social construction, a cultural product, is fallible corrigible and changing like any other kind of human cognition. Paul Ernest works (Ernest 1991, Ernest State University of New York Press) are significant for social constructivism in mathematics (Piotrowska 2008).

Generally it is not controversial that the origins of mathematics are social or cultural (Bishop 1991, Wilder 1953, Wilder 1968, Wilder 1981). The new wave in the philosophy of mathematics with such representatives as Lakatos (1976, 1978), Davis and Hersh (1980), Kitcher (1983), Tymoczko (1986) and Wittgenstein (1978) advocates the thesis that the justification of mathematical knowledge rests on its quasi-empirical basis. This view is causing controversy.

According to Lakatos (1978), the quest for certainty in mathematics leads inevitably to an infinite regression. Any mathematical system depends on a set of assumptions, and there is no way of escaping them. We cannot establish the certainty of mathematics without assumptions. Only from an assumed basis do the theorems of mathematics follow. Therefore theorems of mathematics are conditional and not absolute certain.

For social constructivist the truths of mathematics is a question of social agreement. According to Wittgenstein (1978) mathematical certainty rests on socially accepted rules of discourse embedded in 'forms of life'.

The social constructivism view of mathematics is completely opposed to the absolutists belief that mathematical truths are universal, independent of humankind (mathematics is discovered, not invented), and culture- and value-free.

Conclusions

Gödel blew away the dream of establishing firm foundation of mathematics. Nevertheless "no good tree bears bad fruit": the efforts of the creators of the programmes in the foundation of mathematics resulted in much newer and more beautiful mathematics, and – what was the subject of our considerations – have had great impact on the origin and development of computer science. Once again it is true that: Extinguished philosophies lie about the cradle of every science as the strangled snakes beside that of Hercules.[50]

Today we know that we cannot expect to find a firm foundation for our science. Such a foundation is not necessary for technical research at all. It does not mean that no philosophy is needed. Conversely, "thought has been the father of every advance since time began". Now in computer science we may see an increasing influence of ideas from the philosophy and methodology of science and humanities. Such ideas as those connected with probability, induction, and causality have opened new perspectives in computer science.

R E F E R E N C E S

Aberth, O. (1980), *Computable Analysis*, McGraw-Hill, New York. An attractive technical introduction to analysis in the style of Russian constructivism.

Barendregt, H. (1992), Lambda calculi with types, *in* T. S. E. M. S. Abramsky (Editor), Dov M. Gabbay (Editor), ed., 'Handbook of logic in Computer Science', Vol. 2, Oxford University Press, Oxford, pp. 117–309.

Beeson, M. J. (1983), Proving programs and programming proofs, *in* 'International Congress on Logic, Methodology and Philosophy of Science. Salzburg, Austria', North-Holland, Amsterdam.

Beeson, M. J. (1985), *Foundation of Constructive Mathematics*, Springer-Verlag.

Berka, K. & Kreiser, L., eds (1971), Akademie-Verlag, Berlin. Zweite durchgesehene Auflage, 1973.

Berka, K. & Kreiser, L., eds (1973), *Logik-Texte: kommentierte Auswahl zur Geschichte der modernen Logik*, 2 edn, Akademie Verlag, Berlin.

Bishop, A. J. (1991), *Mathematical Enculturation*, Kluwer, Dordrecht.

Bishop, E. (1967), *Foundations of Constructive Analysis*, McGraw-Hill, New York. The bible of new constructivism. A technical work with a lucid nontechnical introduction.

[50] Adopted from T. H. Huxley.

Bishop, E. (1970), Mathematics as a numerical language, *in* J. M. A. Kino & R. Vesley, eds, 'Intuitionism and Proof Theory', North-Holland, Amsterdam, pp. 53–71.

Bourbaki, N. (1991), *Elements of the History of Mathematics*, Springer-Verlag, Heidelberg. translated from the French by John Meldrum.

Bridges, D. & Reeves, S. (1997), 'Constructive mathematics, in theory and programming practice'.

Brouwer, L. E. J. (1907), Over de Grondslagen der Wiskunde, Doctoral thesis, University of Amsterdam, Amsterdam. Reprinted with additional material (D. van Dalen, ed.) by Matematisch Centrum, Amsterdam, 1981.

Brouwer, L. E. J. (1908), 'Over de onbetrouwbaarheid der logische principes', *Tijdschrift voor Wijsbegeerte* **2**, 152–158. English translation: *On the unreliability of the principles of logic* (Heyting & Freudenthal 1975, 107–111).

Brouwer, L. E. J. (1913), 'Intuitionism and formalism', *Bull. Amer. Math. Soc.* **20**, 81–96. Inaugural address at the University of Amsterdam, read October 14, 1912. Translated by Arnold Dresden. Reprinted in Bulletin (New Series) of the American Mathematical Society, Volume 37, Number 1, Pages 55–564. Tłumaczenie angielskie w: P. Benacerraf, H. Putnam, *Philosophy of Mathematics. Selected Readings,* (ed.) Cambridge University Press, 1987.

Brouwer, L. E. J. (1924), Beweis dass jede volle Funktion gleichmässig stetig ist, *in* 'Koninklijke Nederlandse Akademie van Wetenschappen. Proceedings of the Section of Sciences', Vol. 27, pp. 189–193. Also (Heyting & Freudenthal 1975, 274–280).

Brouwer, L. E. J. (1952), 'Historical background, principles and methods of intuitionism', *South African Journal of Science* .

Browder, F. E., ed. (1976), *Mathematical Developments Arising from Hilbert Problems, Proceedings of Symposia in Pure Mathematics*, Vol. 28, American Mathematical Society, Providence. two parts.

Burstall, R. (2000), 'Christopher strachey–understanding programming languages', *Higher-Order and Symbolic Computation* **13**(52).

Chaitin, G. J. (2004), 'Leibniz, randomness & the halting probability'.
`http://www.cs.auckland.ac.nz/CDMTCS/chaitin/turing.html`.

Church, A. (1932), 'A set of postulates for the foundation of logic', *Annals of Mathematics* **33**, 346–66.

Church, A. (1933), 'A set of postulates for the foundation of logic (second paper)', *Annals of Mathematics* **34**, 839–864.

Church, A. (1936), 'An unsolvable problem of elementary number theory', *American Journal of Mathematics* **58**, 345–363. Presented to the American Mathematical Society, April 19, 1935; abstract in B. Am. Math. S., vol(41), May 1935. Reprinted in (Davis 1965, s. 89–107).

Church, A. (1937), 'Reviews of (Turing 1936–37) and (Post 1936)', *Journal of Symbolic Logic* **2**(1), 42–43.

Church, A. (1940), 'A formulation of the simple theory of types', *Journal of Symbolic Logic* **5**, 56–68.

Church, A. (1941), *The Calculi of Lambda-Conversion*, Princeton University Press, Princeton.

Chwistek, L. (1921), 'Antynomje logiki formalnej', *Przegląd Filozoficzny* **24**, 164–171.

Chwistek, L. (1948), *The Limits of Science: Outline of Logic and of the Methodology of the Exact Sciences*, Routledge and Kegan Paul, London.

Constable, R. L. (1971), Constructive mathematics and automatic programs writers, *in* 'Proceedings of IFIP Congress', Ljubljana, pp. 229–233.

Constable, R. L., Allen, S. F., Allen, S. F., Bromley, H. M., Cleaveland, W. R., Cremer, J. F., Harper, R. W., Howe, D. J., Knoblock, T. B., Mendler, N. P., Panangaden, P., Smith, S. F., Sasaki, J. T. & Smith, S. F. (1986), 'Implementing mathematics with the NuPRL proof development system'.

Curry, H. (1934), Functionality in combinatory logic, *in* 'Proceedings of the National Academy of Sciences', Vol. 20, pp. 584–590.

Curry, H. B., Feys, R. & Craig, W. (1958), *Combinatory Logic*, Vol. 1 of *Studies in logic and the foundations of mathematics*, North-Holland Publishing Company, Amsterdam. with 2 sections by William Craig.

Davis, M. (1965), *The Undecidable: Basic Papers on Undecidable Propositions, Unsolvable Problems, and Computable Functions*, Raven Press, Hewlett, N.Y.

Davis, M. (1988), Influences of mathematical logic on computer science, *in* R. Herken, ed., 'The Universal Turing Machine. A Half-Century Survey', Oxford University Press, pp. 315–26.

Davis, M., ed. (1994), *Solvability, Provability, Definability: The Collected Works of Emil L. Post*, Birkhuser, Boston.

Davis, P. & Hersh, R. (1980), *The Mathematical Experience*, Penguin, Harmondsworth.

de Bruijn, N. (1968), Automath, a language for mathematics, Technical Report TH-report 68-WSK-05, Department of Mathematics, Eindhoven University of Technology. Reprinted in revised form, with two pages commentary, in: (Siekmann 1983).

de Bruijn, N. G. (1970), The mathematical language AUTOMATH, its usage and some of its extensions, *in* 'Symposium on automatic demonstration', Vol. 125 of *Lecture Notes in Mathematics*, p. 29.

de Bruijn, N. G. (1980), A survey of the AUTOMATH project, *in* Seldin & Hindley (1980).

Eden, A. H. (2007), 'Three paradigms of computer science', *Minds and Machines* **17**(2), 135–167. Special issue on the Philosophy of Computer Science.

Ernest, P. (1991), *The Philosophy of Mathematics Education*, Routledge, London.

Ernest, P. (State University of New York Press), *Social Constructivism as a Philosophy of Mathematics*, 1998, Albany, New York.

Ewald, W., ed. (1996), *From Kant to Hilbert: A Source Book in the Foundation of Logic*, Vol. 1–2, Clarendon Press, Oxford.

Frege, G. (1879), *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*, Halle. Introduction dated: 18 XII 1878. Reprinted in (Frege 1993). Shortened version in (Berka & Kreiser 1971). English translation in (van Heijenoort 1967, pp. 1–82).

Frege, G. (1884), *Die Grundlagen der Arithmetik. Eine logisch-mathematische Untersuchung ber den Begriff der Zahl*, W. Koebner, Breslau. Unveränderter Neudruck: M. & H. Marcus, Breslau, 1934. English translation: (Frege 1968), (Frege 1980).

Frege, G. (1893–1903), *Grundgesetze der Arithmetik, Begriffsschriftlich abgeleitet*, Vol. 1–2, Jena. Reprint: Darmstadt 1961; English translation (Furth 1964).

Frege, G. (1964), *The basic laws of arithmetic*, University of California Press. Translated by Montgomery Furth.

Frege, G. (1967), Letter to Russell, *in* van Heijenoort 1967, pp. 127–8. 2nd ed., 1971.

Frege, G. (1968), *The Foundations of Arithmetic: A Logico-Mathematical Enquiry into the Concept of Number*, Blackwell. English translation by J. L. Austin.

Frege, G. (1980), *The Foundations of Arithmetic: A logico-mathematical enquiry into the concept of number*, Northwestern University Press, Evanston, Illinois. (Frege 1884), trans. by J. L. Austin.

Frege, G. (1993), *Begriffsschrift und andere Aufsätze*, Hildesheim.

Friedman, H. (1975), Some systems of second order arithmetic and their use, *in* 'Proceedings of the 17th International Congress of Mathematicians', Vancouver, BC.

Gödel, K. (1931), 'Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I', *Monatshefte für Mathematik und Physik* **38**, 173–198. Received: 17. XI. 1930. Translated in: Solomon Feferman (ed.), "Kurt Gdel: Collected Works", Oxford University Press, volume 1 (1986) 144–195. Includes the German text and parallel English translation. English translations in (Davis 1965, 5–38), (van Heijenoort 1967, 595–616). Online version (translation by B. Meltzer, 1962):

http://www.ddc.net/ygg/etext/godel/ PDF version (translation by M. Hirzel): http://nago.cs.colorado.edu/~hirzel/papers/canon00-goedel.pdf.

Gillies, D. A. (2002), Logicism and the development of computer science, *in* A. C. Kakas & F. Sadri, eds, 'Computational Logic: Logic Programming and Beyond, Part II', Springer, pp. 588–604. A Paper for Bob Kowalski's 60th Birthday.

Gödel, K. (1930), 'Die Vollständigkeit der Axiome des logischen Funktionskalküls', *Monatshefte für Mathematik und Physik* **37**, 349–360. (Gödels Dissertation, 1929) also with English transl. in *Collected Works*, S. Feferman et al., eds., vol. 1, Oxford University Press, Oxford, 1986, pp. 60–101; English translation in (van Heijenoort 1967, pp. 582–591).

Hayashi, S. & Nakano, H. (1988), *PX: A Computational Logic*, MIT Press, Cambridge MA.

Heyting, A. (1930), 'Die formalen Regeln der intuitionistischen Logik', *S.-Ber. Preuss. Phys.-Math. Kl. II* pp. 42–56.

Heyting, A. (1934), *Mathematische Grundlagenforschung. Intuitionismus. Beweistheorie*, Springer Verlag, Berlin.

Heyting, A. (1956), *Intuitionism. An Introduction*, Amsterdam. Wyd. 2: 1966.

Heyting, A. & Freudenthal, H., eds (1975), *Collected Works of Luitzen Egbertus Jean Brouwer*, North-Holland, Amsterdam.

Hilbert, D. (1899), *Grundlagen der Geometrie*, Teubner-Verlag, Leipzig. 8th amended ed. by P. Bernays with supplements, Stuttgart, 1956-1987; 14th ed., with contributions by H. Kiechle et al., M. Toepell, ed., Stuttgart, 1999; the first English translation *The Foundations of Geometry* by E. J. Townsend, Open Court, Chicago, 1902; new translation by L. Unger, Open Court, La Salle, 1972. [The first German edition of the *Grundlagen* was a contribution to the Festschrift zur Enthllung des Gauss-Weber Denkmals in Gttingen, pp. 1–92; later editions were published separately].

Hilbert, D. (1900), 'Mathematische Probleme', *Nachrichten von der Kniglichen Gesellschaft der Wissenschaften zu Gttingen, Math.-Phys. Klasse* (3), 253–297. Lecture held at the Paris ICM 1900. Revised version in (Hilbert 1901*b*) and (Hilbert 1932–1935), pp. 290–329; English translation in (Hilbert 1901*a*); also in (Browder 1976), pp. 1–34; (Yandell 2002), pp. 324–389; (?), pp. 240–282; selected parts in (Ewald 1996), pp. 1096–1105; (Reid 1970), chap. 9; French translations by M. L. Laugel (with corrections and additions) in *Compte Rendu Deuxième Congrès International des Mathématiciens tenu à Paris du 6 au 12 août* 1900, E. Duporcq, ed., Gauthier-Villars, Paris, 1902, pp. 58–114; (fragments) L'Enseignement Mathématique 2 (1900) 349–354 and *Revue Générale des Sciences Pures et Appliquées* 12 (1901) 168–174;

annotated Russian edition, *Problemy Gilberta*, P. S. Alexandrov, ed., Nauka, Moscow, 1969; its German edition *Die Hilbertschen Probleme*, Akademische Verlagsgesellschaft, Leipzig, 1976, 2nd ed. 1979, reprint 1998 (both the Russian and German editions give the revised version of (Hilbert 1932–1935)., David Joyce, Clark University, produced a list of Hilbert's problems and a web version of Hilbert's 1900 address in March 1997 – `http://aleph0.clarku.edu/~djoyce/hilbert/problems.html`.

Hilbert, D. (1901*a*), 'Mathematical problems', *Bulletin of the American Mathematical Society* **8**, 437–479. Lecture delivered at the International Congress of Mathematicians at Paris in 1900, translated by Mary Winton Newson. Reprinted in the *Bulletin* 37 (2000) 407–436;.

Hilbert, D. (1901*b*), 'Mathematische Probleme', *Archiv der Mathematik und Physik* **1**, 44–63; 213–237.

Hilbert, D. (1926), 'Über das Unendliche', *Mathematische Annalen* **95**, 161–190. Lecture given in Mnster, 4 June 1925. Also in (Hilbert 1899), 7th ed., pp. 262–288, shortened version in *Jahresber. Deutsch. Math. Verein.* 36 (1927) 201-215; English translation (by S. Bauer-Mengelberg) in (van Heijenoort 1967), pp. 367–392; 2nd ed., 1971; French translation (by A. Weil), *Sur l'infinie*, *Acta Math.* 48 (1926) 91–122; Polish translation (by R. Murawski) in (Murawski 1986, pp. 288–307).

Hilbert, D. (1928), 'Die Grundlagen der Mathematik', *Abhandlungen aus dem Seminar der Hamburgischen Universität* **6**, 65–85. followed by Diskussionsbemerkungen zu dem zweiten Hilbertschen Vortrag by H. Weyl, pp. 86–88, and Zusatz zu Hilberts Vortrag by P. Bernays, pp. 89–95; shortened version in (Hilbert 1998), 7th ed., pp. 289–312; English translation (by S. Bauer-Mengelberg and D. Follesdal) in (van Heijenoort 1967), pp. 464–479. On-line publication `www.marxists.org/reference/subject/philosophy/works/ge/hilbert.htm`.

Hilbert, D. (1932–1935), *Gesammelte Abhandlungen*, Vol. 3, 2nd ed., 1970 edn, Springer-Verlag, Berlin.

Hilbert, D. (1998), *The Theory of Algebraic Number Fields*, Springer Verlag, Berlin.

Howard, W. A. (1980), The formulae-as-types notion of construction, *in* Seldin & Hindley (1980), pp. 479–490. original paper manuscript from 1969.

Ishihara, H. (2005), Constructive reverse mathematics: Compactness properties, *in* L. Crosilla & P. Schuster, eds, 'From Sets and Types to Analysis and Topology: Towards Practicable Foundations for Constructive Mathematics', The Clarendon Press, Oxford.

Ishihara, H. (2006), 'Reverse mathematics in Bishop's constructive mathematics', *Phil. Scientiae, Cahier Special* **6**, 43–59.

Kitcher, P. (1983), *The Nature of Mathematical Knowledge*, Oxford University Press, Oxford.

Kleene, S. C. (1936), 'General recursive functions of natural numbers', *Mathematische Annalen* **112**(1), 727–742.

Kleene, S. C. (1945), 'On the interpretation of intuitionistic number theory', *Journal of Symbolic Logic* **10**, 109–124.

Kleene, S. C. (1952), *Introduction To Metamathematics*, North-Holland Publishing Co, Amsterdam and P. Noordhoff N. V., Groningen.

Kleene, S. C. & Rosser, J. B. (1935), 'The inconsistency of certain formal logics', *Annals of Mathematics* **36**, 630–36.

Kolmogorov, A. N. (1932), 'Zur deutung der intuitionistischen logik', *Mathematische Zeitschrift* **35**, 58–65. Also (Berka & Kreiser 1973, pp. 178–185).

Kushner, B. A. (1985), 'Lectures on constructive mathematical analysis', *Amer. Math. Soc. Providence RI.*

Lakatos, I. (1976), *Proofs and Refutations*, Cambridge University Press, Cambridge.

Lakatos, I. (1978), *Philosophical Papers*, Cambridge University Press, Cambridge.

Lindström, S., Palmgren, E., Segerberg, K. & Stoltenberg-Hansen, V., eds (2007), *Logicism, Intuitionism, and Formalism What Has Become of Them?*, Vol. 341 of *Synthese Library*, Springer.

Linsky, B. (2009), Chwistek's theory of constructive types, *in* J. M. M. M. W. Lapointe, S.; Wolenski, ed., 'The Golden Age of Polish Philosophy. Kazimierz Twardowski's Philosophical Legacy', Vol. 16 of *Logic, Epistemology and the Unity of Science, Part IV*, Springer, chapter 14, pp. 203–219.

Martin-Löf (1984*a*), 'Constructive mathematics and computer programming', *Philosophical Transactions of the Royal Society of London. Series A, Mathematics and Physical Sciences* **A 312**, 501–518.

Martin-Löf, P. (1982*a*), Constructive mathematics and computer programming, *in* H. P. K.-P. P. L. J. Cohen, J. Łoś, ed., 'Proceedings of the Sixth International Congress for Logic, Methodology and Philosophy of Science in Hannover in August 1979', Vol. 6, North Holland, pp. 153–175. Reprinted in (Martin-Löf 1984*a*, Martin-Löf 1985).

Martin-Löf, P. (1982*b*), An intuitionistic theory of types: predicative part, *in* H. Rose & J. Shepherdson, eds, 'Logic Colloquium 1973', North Holland, pp. 73–118.

Martin-Löf, P. (1984*b*), *Intuitionistic Type Theory*, Bibliopolis, Naples. Notes by Giovanni Sambin of a series of lectures given in Padua, June 1980.

Martin-Löf, P. (1985), Constructive mathematics and computer programming, *in* C. Hoare & J. Shepherdson, eds, 'Mathematical Logic and Programming Languages', Prentice-Hall International, Englewood Cliffs, N. Y.

Martin-Löf, P. (1998), An intuitionistic theory of types, *in* G. Sambin & J. Smith, eds, 'Twenty-Five Years of Constructive Type Theory', Vol. 36 of *Oxford*

Logic Guides, Oxford Science Publications, pp. 127–172. Proceedings of a Congress Held in Venice, October 1995.

McCarthy, J. (1960), 'Recursive functions of symbolic expressions and their computation by machine', *Communications of the ACM* **3**(4), 184–195. Part I, Part II was never published.

Milner, R. (1984), A proposal for standard *ml*, *in* 'Proceedings of the ACM Symposium on LISP and Functional Programming', Austin, pp. 184–197.

Mordechai, B.-A. (2001), 'Constructivism in computer science education', *Journal of Computers in Mathematics and Science Teaching* **20**(1), 45–73. This article is an extended version of a paper that was presented at the *Twenty-Ninth SIGCSE Technical Symposium on Computer Science Education*, Atlanta, GA, 1998.

Murawski, R. (1999), *Recursive Functions and Metamathematics*, Kluwer Academic Publisher, Dordrecht.

Murawski, R., ed. (1986), *Filozofia matematyki. Antologia tekstów klasycznych*, ii wyd. 1994 edn, Poznań. Wybór i opracowanie R. Murawski.

Olszewski, A. (2009), *Teza Churcha. Kontekst historyczno-filozoficzny*, Universitas.

Oosten, J. V. (2000), Realizability: A historical essay, Technical report.

Piotrowska, E. (2008), *Społeczny konstruktywizm a matematyka*, Wydawnictwo Naukowe UAM, Poznań.

Poincaré, H. (1906), *La Science et l'Hypothèse*, Flammarion, Paris.

Poincaré, H. (1916), *Science et Méthode*, Flammarion, Paris.

Poincaré, H. (1925), *La Valeur de La Science*, Flammarion, Paris.

Post, E. (1936), 'Finite combinatory processes – Formulation I', *Journal of Symbolic Logic* **1**, 103–105. received Oct. 7, 1936. Presented to the American Mathematical Society Jan. 1937. Abstract in Bull. of Am. Math. Soc., Nov. 1936. Reprinted in (Davis 1994), pp. 289–291.

Post, E. (1965), Absolutely unsolvable problems and relatively undecidable propositions – account of an anticipation, *in* M. Davies, ed., 'The Undecidable: Basic Papers on Undecidable Propositions, Unsolvable Problems, and Computable Functions', Raven Press, Hewlett, N.Y., pp. 340–433.

Ramsey, F. P. (1931), *The Foundations of Mathematics and other logical essays*, Routledge & Kegan.

Reid, C. (1970), *Hilbert*, Springer-Verlag, New York. reprinted Copernicus, New York, 1996.

Russell, B. & Whitehead, A. N. (1910–1913), *Principia Mathematica*, Vol. 1–3, Cambridge. t. I – 1910, t. II – 1912, t. III – 1913. 2nd edition, with a new introduction and appendices, 1925–1927.

Schönfinkel, M. (1924), 'Über die bausteine der mathematischen logik', *Mathematische Annalen* **92**, 305–316. Translated by Stefan Bauer-Mengelberg as *On the building blocks of mathematical logic* in (van Heijenoort 1967, pp. 355–66).

Scott, D. (1970), Constructive validity, *in* 'Symposium on Automatic Demonstration', Vol. 125 of *Lecture Notes in Mathematics*, Springer-Verlag, pp. 237–275.

Seldin, J. P. & Hindley, J. R., eds (1980), *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, Academic Press.

Shoenfield, J. R. (1991), *Recursion Theory*, Vol. 1 of *Lecture Notes in Logic*, Springer-Verlag, Heidelberg, New York.

Siekmann, J. H., ed. (1983), *Automation of Reasoning: Classical Papers on Computational Logic, 1967–1970*, Vol. 2, Springer Verlag.

Thom, R. (1973), Modern mathematics: does it exist?, *in* A. G. Howson, ed., 'Developments in Mathematical Education', Cambridge University Press, Cambridge, pp. 195–209.

Thompson, S. (1991), *Type Theory and Functional Programming*, Addison-Wesley.

Troelstra, A. S. (1991), A history of constructivism in the 20th century, ITLI Prepublication Series ML-91-05, University of Amsterdam, `http://staff.science.uva.nl//~anne/hhhist.pdf`.

Troelstra, A. S. (1999), 'From constructivism to computer science', *Theor. Comput. Sci.* **211**(1-2), 233–252.

Troelstra, A. S. & van Dalen, D. (1988), *Constructivism in Mathematics*, North Holland.

Trzęsicki, K. (2006), 'From the idea of decidability to the number Ω', *Studies in Grammar, Logic and Rethoric* **9**(22), 73–142.

Turing, A. (2004), *The Essential Turing: Seminal Writings in Computing, Logic, Philosophy, Artificial Intelligence, and Artifical Life Plus The Secrets of Enigma*, Oxford University Press, Oxford.

Turing, A. M. (1936–37), 'On computable numbers, with an application to the Entscheidungsproblem', *Proceedings of the London Mathematical Society* **42** (Series 2), 230–265. Received May 25, 1936; Appendix added August 28; read November 12, 1936; corrections Ibid. vol. 43(1937), pp. 544–546. Turing's paper appeared in Part 2 of vol. 42 which was issued in December 1936 (Reprint in: (Turing 1965); 151–154). Online version: `http://www.abelard.org/turpap2/tp2-ie.asp`.

Turing, A. M. (1937), 'Computability and λ-definability', *Journal of Symbolic Logic* **2**, 153–163.

Turing, A. M. (1965), On computable numbers, with an application to the Entscheidungsproblem, pp. 116–151.

Tymoczko, T., ed. (1986), *New Directions in the Philosophy of Mathematics*, Birkhauser, Boston.

van Atten, M., Boldini, P., Bourdeau, M. & Heinzmann, G., eds (2008), *One Hundred Years of Intuitionism (1907–2007)*, Birkhauser Verlag AG, Basel-Boston-Berlin.

van Heijenoort, J., ed. (1967), *From Frege to Gödel. A Source Book in Mathematical Logic 1879-1931*, 1 edn, Harvard University Press, Cambridge Mass. 2nd ed., 1971.

van Stigt, W. P. (1990), *Brouwer's Intuitionism*, North-Holland, Amsterdam.

Veldman, W. (2005), Brouwer's fan theorem as an axiom and as a contrast to Kleene's alternative, preprint, Radboud University, Nijmegen, Netherlands.

Vygotsky, L. S. (1962), *Thought and language*, M.I.T. Press, Cambridge Mass.

Weyl, H. (1918), *Das Kontinuum*, Veit, Leipzig.

Weyl, H. (1967), Comments on hilbert's second lecture on the foundations of mathematics, *in* 'From Frege to Gödel. A Source Book in Mathematical Logic 1879-1931', Harvard University Press, Cambridge Mass. 2nd ed., 1971.

Wilder, R. L. (1953), 'The origin and growth of mathematical concepts', *Bull. Amer. Math. Soc.* **59**, 423–448. A retiring address, as Vice President and Chairman of Section A, American Association for the Advancement of Science, delivered before a joint meeting of Section A, the American Mathematical Society, and the Mathematical Association of America, at St. Louis, Missouri, December 29, 1952; received by the editors March 13, 1953.

Wilder, R. L. (1968), *The origin and growth of mathematical concepts*, John Wiley & Sons Inc.

Wilder, R. L. (1981), *Mathematics as a Cultural System*, Pergamon, Oxford.

Wittgenstein, L. (1953), *Philosophical Investigations*, Blackwell, Malden.

Wittgenstein, L. (1974), *Philosophical Grammar*, Basil Blackwell, Oxford.

Wittgenstein, L. (1978), *Remarks on the Foundations of Mathematics*, 3rd, revised and reset edn, Blackwell. English translation by G. E. M. Anscombe.

Wittgenstein, L. (1980), *Remarks on the Philosophy of Psychology*, University of Chicago Press.

Yandell, B. H. (2002), *The Honors Class. Hilberts Problems and Their Solvers*, A. K. Peters, Natick, MA.

Kazimierz Trzęsicki
Chair of Logic, Informatics and Philosophy of Science
University of Białystok
kasimir@uwb.edu.pl

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 22 (35) 2010

Andrzej Cieśluk
Maria Curie-Skłodowska University in Lublin

DE RE/DE DICTO DISTINCTIONS (SYNTACTIC, SEMANTIC AND PRAGMATIC INTERPRETATION)

Abstract: The aim of this article is the systematization of the most commonly applied *de re/de dicto* distinctions in philosophy. In this work the most important contexts in which these distinctions occur are distinguished, and their syntactic, semantic and pragmatic definitions are given. The article consists of two parts: 1) the main theoretical contexts of this distinction, 2) the modern attempt to systematize different versions of this distinction following Thomas McKay and Michael Nelson, and 3) an attempt to clarify McKay's and Nelson's systematization in terms of intensional operator, designator and conviction of the object.

I

The philosophical characterization of modality up to the 20th century (the Antiquity and the Middle Ages) may be considered from three perspectives:

1) **essentialism** – properties belong to objects in two ways: accidental and essential; properties which belong to an object in the second way determine its essence (Aristotle, Thomas Aquinas) (Bacon 1998, p. 668);
2) **modal realism** (as opposed to **modal nominalism**, professed, among others, by Ockham) – states that modalities exist objectively in the world of particular things and are not just a feature of our way of speaking about such things;
3) **propositionalism** (the contradiction of **objectualism**) – states that there are objective of sentences which are the subjects of modal properties (Thomas Aquinas, Boethius).

Looking into encyclopedias and textbooks (Gallois 1998, pp. 815–817), one can see that the *de re/de dicto* distinction in modern, 20th century philosophical literature concerns many different, but connected theoretical questions. Generally, *de dicto* means "concerning a *dictum*", where a *dictum* is something that has propositional content – it can be a sentence, a judgment in the logical meaning, and so on. *De re* means "concerning a thing (*re*)" that has some defined properties.

The etymological explanation does not give answers to the philosophical questions because the *de re*/*de dicto* distinction concerns many theoretical contexts, positions, conceptions, controversies (and so on) which have crystallized in modern philosophy of language, philosophical logic and analytical ontology. I will discuss the most typical of them.

(1) The context of modal logic of quantifiers. By examining the connection between the modal operators and the quantifiers, one can notice a difference, for the illustration of which authors often use parts of the Barcan[1] formula:

1.1 $$\Box(x)\phi x,$$

where $\Box$ is a modal operator and means: "is necessary that" ($\Diamond$ means: "is possible that"), ϕ represents any predicate. In the Barcan formula, 1.2 is the antecedent, 1.1 is the consequent which says: it is the necessary truth (*dictum*) that everything is ϕ. This sentence points to the modality of *de dicto*. The formula:

1.2 $$(x)\Box\phi x$$

states that everything (each one of the things) is necessarily ϕ. It represents the *de re* modality.

This distinction was more precisely defined by Cresswell and Hughes:

> "The α formula, containing the modal operator $\Box$ or $\Diamond$, will be said to express a modality *de re* if there is a scope of some modal operator in it which contains some free occurrence of an individual-variable; otherwise α will be said to express a modality *de dicto*." (Cresswell, Hughes 1968, p. 184).

(2) The context of essentialism, in which the *de re*/*de dicto* distinction may lead to acknowledging some of the subject's properties as necessary. Recognizing whether a modal sentence containing the subject's name is true or not, may depend on essential properties of the subject. The following example points to the connection between this context and the context of the modal logic of quantifiers. The sentence: "Every man is necessarily a human being" can be represented by a formula in which the necessity operator has the free variable within its reach: $\forall x(Cx \rightarrow \Box Rx)$.

[1] The Barcan formula has the following form: $(x)\Box\phi x \rightarrow \Box(x)\phi x$.

(3) The context of the dispute is between modal realism and modal nominalism. According to the first of these items, it is maintained that predicates such as: "he is necessarily a human", "possibly, he is tall" express a particular modal kind of property of objectively existing subjects or states of affairs. A modal sentence expresses the modal *de re* predication when it assigns a modal property to subjects or states of affairs. In particular, "John is necessarily a human" expresses the modal *de re* predication towards John.

The opposite, proclaiming the thesis of the linguistic character of modality, here called the modal nominalism, characterizes modality exclusively in terms of a given description. According to this thesis, necessity or possibility do not belong to the order of things or happenings, but to the order of the way of speaking about them. This particularly applies to the semantic argument suggested by Quine (Quine 1953, pp. 139–159; Quine 1976, pp. 185–196; Quine 1979, pp. 268–274) which concerns the reduction of the *de re* to *de dicto* modality.

(4) The context of the dispute between propositionalism and objectualism. The general thesis of propositionalism says that propositions are objective correlations of sentences which are the subjects of modal properties. For example, according to Meinong, objectives are the subjects of modal properties.[2] The main difference between an object and an objective is that the first one is the subject of presentation, while the second one is the subject of judgment. On an ontological level, modality is treated as the property of objectives; on the linguistic level – as the property of judgments (Żegleń 1990, p. 194). Let us add that the equivalents for an objective are "situations" (Wittgenstein), "propositions" (Frege) or "noemats" (Husserl).

By accepting the existence of propositional objects these authors usually also assume a stronger thesis about the non-reduction of *de dicto* modality to *de re*: at least some of the modal properties of propositional existences are non-reducible to the properties of their elements. The modality is completely a property of judgments or states of things, not things or relations.

The thesis of the reduction of *de re* modality to *de dicto* is even stronger: all of the modal properties (of things, features and relations) are reducible to the properties of propositional existences. A representative of this position is Alvin Plantiga, who proposes a method of reducing sentences which con-

[2] Ingarden proposes that the notion "objective" can be treated as "a intentional state of affairs", R. Ingarden 1972.

tain *de re* modalities to sentences containing only the *de dicto* modalities. (Cocchiarella 2007, p. 60). One can also encounter the thesis of reduction of the *de re* modality to *de dicto* in the conception of logic atomism.[3]

(5) The context of the rigid designators and the description. Proper names, such as "Napoleon", can be defined as rigid designators, because they designate the same individual in every possible world. One can speak of the *de re* interpretation of a given name when it appears within the reach of the modal operator as the rigid designator. The sentence: "It is possible that Napoleon spoke Chinese" should be interpreted *de re*. However, the sentence: "It is possible that the most famous French general spoke Chinese" should be interpreted *de dicto* (Gallois 1998, pp. 815–817).

By enlarging the range of rigid designators beyond proper names one can read the descriptions in two ways: referential and attributive. The first one refers to the *de re* interpretation the second to the *de dicto* interpretation. The possibilities of these two interpretations are being discussed in modern literature (particularly concerning the so-called Evans modal argument (Evans 1982).

(6) The context of the propositional attitudes theory. A distinction between two kinds of convictions is introduced here (Quine 1976, pp. 185–196). The conviction of the *de dicto* type is understood, according to Frege's conception, as a two-part relation between a person and *dictum*. The *de re* conviction is understood, according to Russell theory, as at least a three-part relation between a person, an object and a property.

Let us consider the following example: "John believes that the leading politicians are wealthy". If this sentence is true, John, without knowing who currently is the president, is convinced that the president of his country is wealthy. Therefore, he has the conviction of the *de dicto* character, consisting in being in a two-place relation with a to *dictum* (an opinion or a judgment).

[3] Jan Woleński is a supporter of the propositional perception theory with consists in two assumptions: that sentences express judgment, and that observation can be treated as a propositional attitude). According to the theory the content of an observation is always communicated by a sentence. Thus, the sentence "I see Kraków" is treated as logically equivalent to the sentence "I see that this is Kraków". Woleński pays attention to the dual possibility of expressing intentional sentences: in direct and indirect speech. "Indirect speech is opposed to simple or direct speech (*oratio recta*). According to this, sometimes the *de modo recto* and *de modo obliquo* presentations are distinguished. [...] The first ones present their subjects directly, the second are in some way relational, for example by the content mediation." Woleński [2005], pp. 400–40. (translating by author). The relations of cognitive acts (*de modo obliquo*) correspond with the modality of *de dicto*, the cognitive of acts themselves (*de modo recto*) – with the modality of *de re*.

However, let us assume that John meets Barack Obama. From Barack Obama appearance John concludes that he is wealthy. Consequently, John believes in the truth of the sentence: "The current president is wealthy". However, the above-mentioned relation is of a three-place kind and takes place between John, Barack Obama and the property of being wealthy.

In case of the *de re* conviction we speak of the relation enriched with the knowledge about the designate, in the case of the *de dicto* conviction that kind of knowledge is not possessed.

In *The Oxford Dictionary of Philosophy* this is the only interpretation given, which shows its significance in literature. Simon Blackburn characterizes the distinction between two kinds of interpretations of these sentences, which can, however not always must be understood, as referring to a particular thing [...] So, when the truth of my words demands for me to refer to a particular thing with the first interpretation, with the second one it makes me refer first of all to a particular sentence, which truth I would like to wish, and which can become the truth because of many things.

Using predicate calculus, the difference between these interpretations for the sentence: "I want to have a yacht" Blackburn expresses in the following way: (i) "$\exists x(Px \wedge I$ wish I had $x)$", and (ii) "I wish for this to happen: $(\exists x)(Px \wedge Mx)$, where P means: "being a yacht", M – "being my property". Galois pays attention to a similar dependence, distinguishing the context of conviction attitudes as the main one.

(7) The context of Montague grammar. For a sentence including a fictional name: "Jack is looking for an unicorn", we can also propose two interpretations: "Jack is looking for an unicorn as a representative of a species" (*de dicto*); and the interpretation: "There exists a particular unicorn which Jack is looking for" (*de re*) (Montague 2002, pp. 29–30). It is assumed here that in case of the *de dicto* intention one can consider any object belonging to a class with a certain property, while in case of *de re* a particular element of that class is considered. These two interpretations became the inspiration for the creation of the PTQ theory by Richard Montague.

II

In the supplement to their SEP article on propositional attitudes, Thomas McKay and Michael Nelson (McKay, Nelson 2005) try to order the vast range of the literature concerning the *de re/de dicto* distinction. They interpret it according to three research perspectives, and suggest suitable definitions. In the first, syntactic presentation, they use the concept of intensional verb range:

Def I:

> "A sentence is syntactically *de re* just in case it contains a pronoun or free variable within the scope of an opacity verb that is anaphoric on or bound by a singular term or quantifier outside the scope of that verb. Otherwise, it is *syntactically de dicto*" (McKay, Nelson 2005).

In order to illustrate this definition, they appeal to the following example:

1. Ralph believes that someone is a spy.

According to the Def I, two interpretations of the 1. can be presented. The authors of the entry give the following interpretation:

1' Ralph believes that $\exists x$ (x is a spy),

1" $\exists x$ (Ralph believes, that x is a spy).

They claim that almost everybody believes, like Ralph, that there are spies (like in 1'), whereas only a few believe such particular people are spies (like in 1"). They suggest a paraphrase for 1": "There is someone of whom Ralph thinks that he is a spy".

It is clear that the difference between 1' and 1" concerns the range of the existential quantifier. It has a limited range in 1' (the *de dicto* interpretation); in 1" it has a wide range and binds the variable which is free in the range of the intensional verb (the *de re* interpretation). The authors notice that Def I can be applied both to the alethic and the temporal modality.

Next, they propose a semantic distinction. They use for this purpose the following definition:

Def II:

> "A sentence is *semantically de re* just in case it permits substitution *salva veritate*. Otherwise, it is semantically *de dicto*" (McKay, Nelson 2005).

They point out that the semantic distinction was criticized by Quine, who did not permit the possibility of quantifying in referentially opaque contexts.[4] In the open formula 1", the intentional verb "believes" refers to any object whose name is put in the place of the x variable. Quine's famous example concerning **Bel** (believe operator) operators speaks about Ortcutt, whom Ralph treats as a public person of significant importance, for example

[4] Issues concerning opaque context were considered for the first time by Quine in: Quine [1943].

a mayor. At the same time, one evening he notices a mysterious man standing on the street corner, whose behavior causes the suspicion of espionage. By calling Ortcutt a mayor in the first situation, and a "mysterious man at the street corner" in the second one, the sentence: a) "Ralph is convinced that he (the mysterious man at the street corner) is a spy" is acknowledged to be true, whereas the sentence: b) "Ralph is convinced that the mayor is a spy" is acknowledged to be false. Whether the name "Ortcutt" fulfills the sentence function: "Ralph is convinced about x that x is a spy" depends on the way in which Ortcutt is described.

Quine's conclusion is as follows: It is a nonsense to claim that Ortcutt, irrespective of the description, fulfills (or not) the condition *salva veritate.* In the example of Ralph and Ortcutt, it is not allowed to substitute x for any co-referential name containing "Ortcutt", because the truth value is also dependent on Ralph's knowledge of the dark side of mayor's personality (Quine 1976, pp. 331–333).

I do not agree with Quine thesis; I will show why it is wrong in section 3 when I analyze Kripke's argument.

McKay and Nelson define the so-called metaphysical *de re/de dicto* distinction as follows:

Def III:

> "An attribution is metaphysically *de re* with respect to an object **o** just in case it directly attributes a property to **o**. It is metaphysically *de dicto* with respect to **o**, if it indirectly involves **o**, and independent of **o** if it doesn't" (McKay, Nelson, 2005).

As an example they use the sentence: "Sally believes that Bill is happy". Let us assume that Sally has reasons to suppose that everybody who is in Disneyland is happy, and that at the moment there is exactly one person there. Let us consider the following reasoning:

(1) Sally believes that every person in Disneyland is happy.
(2) Bill is the only person in Disneyland.
(3) Sally believes that Bill is happy.

The opponents of the "metaphysical" interpretation of *de re* will not agree with the above reasoning, because they will claim that Sally does not attribute the quality of being happy directly to Bill. Kaplan gives a similar example that concerns Ralph being the shortest spy (Kaplan 1969, pp. 178–214). In a particular group of spies there is one who is the shortest; thought Ralph does not know which one it is. Of course this is not a clear case that indirect attribution needs indirect knowledge about the object or it's possible only by a description, Kaplan takes this problem, claims that Ralph does not have the *de re* conviction about a particular person. Similarly, Sally does not have the *de re* conviction about Bill.

It should be mentioned that the matter of the analytical relationships between the syntactic, semantic and metaphysical *de re/de dicto* distinctions in McKay's and Nelson's review presents as a complex problem.[5] It concerns many issues of linguistic, semiotic, ontological and epistemological character.

I will now analyze the above-mentioned definitions and present their modified versions. First, taking Def I and Def II into consideration, then taking Def III into consideration.

III

The following part of this article will clearly distinguish natural language from the language of logic. In the description of the first one we will use the notion of a sentence in the sense of traditional linguistics, in the second one – the logical idea of a closed formula as an expression in which all of the variables are bound. The set of formalizing functions, which enable us to specify the logical form of natural language sentences, will be used as the translation function from natural language sentences to the language of logical formulas. The symbol "f" will be used to mark any function of that sort. We are assuming that f assigns the expression-token of the natural language to appropriate symbol-token of the logical language. In particular, it assigns the appropriate sentences of natural language to the logical formulas; and it assigns the designators to the names (from the logical language), that is either simple name or descriptions (which are treated as complex naming expressions). In that case, a natural language sentence Z corresponds to a sentence $f(Z)$ in formal sense and the designator t corresponds to the name $f(t)$. Let us notice that if there are any variables in the expressions $f(Z)$ or $f(t)$, they are bound variables.

We will use the language of first-order logic with identity, the description operator, as well as modal and assurance sentence functors. The set of this language's expressions is the counter-domain of formalisation functions.

We assume that the deductive apparatus of this language is a conceivably weak extension of ordinary logical laws and rules on the intensional contexts. Particularly, the rule of the intersubstitutivity of identical elements

[5] Kallfelz stresses the role of modal pronouns in the context of the *de re/de dicto* distinction, Kallfelz [2007] p. 3.

is limited to the extensional functors (it does not apply to the intensional functors).

(i) McKay and Nelson have created Def I and II by relativizing the *de re/de dicto* distinction to the concept of a sentence (in the syntactic sense). For example, by speaking about the *de dicto* modality, they mean the modality in the given sentence. But using the concept of an intensional verb in the definition of this distinction, one encounters a difficulty: how can you speak about a part of speech in terms of syntactic scope?

When defining the scope, it is more appropriate to use the concept of a sentence functor, that is the operator in which at least one of the arguments is a sentence expression. The sentence operator has a narrow scope when there exists, within its reach, a free variable bound outside its scope (the *de re* meaning), or a wide range when there is no free variable bound outside its scope (the *de dicto* meaning) (Hughes, Cresswell 1898, p. 184).

Understanding the operator may in this context leads to a mistake, because one operator may exist in several tokens (for each of them we can speak about their individual scope). This may lead to a situation when while saying something about a particular sentence one does not know which token operator we have in mind.

Let me illustrate this with an example: "Ralph thinks that there is somebody of whom Anna thinks that he is immortal". The intensional functor "thinks" appears in this sentence in two copies, *de dicto* in the first case, *de re* in the second one. Its formal representation is as follows: Ralph-thinks[$\exists x$Anna-thinks[Nx]]. The free variable x (in Anna-thinks[Nx]), standing within the scope of the second appearance of the "thinks" operator, is bound by the existential quantifier standing outside of the scope of this operator. According to Def I, this representation defines the syntactic interpretation *de re* of the sentence in question. However, when the first token of the intensional operator "thinks" is taken into consideration, the part of sentence is interpreted as *de dicto*. It is therefore more appropriate to speak, according to Def I', about the *de re/de dicto* distinction for the formal representations of sentences, not the sentences themselves.

Taking this analysis into account, I suggest the following definition for the syntactic *de re/de dicto* distinction:

Df I' the token of the functor F has in sentence Z *de re* meaning due to the formalization f only when there exists a bound variable free within the $f(Z)$ reach in the $f(Z)$ formula. Otherwise it is *de dicto*.

Let us notice that the syntactic *de re/de dicto* distinction is wide enough to apply also to the extensional operators. Def I, which uses the term of a token of the sentence operator, allows us to deal with the above mentioned difficulties in reference to intensional operators. Let us see how. Df I' enables the explanation of the difficulties pointed out by Kripke.[6] He gives the example of a multiple complex sentence:

2. Watson doubts that Holmes believes that Smith's murderer is insane.

It is in three ways ambiguous whether the description is given wide, intermediate, or narrow scope:

2' [the x: x murdered Smith]
(Watson doubts that (Holmes believes that (x is insane)))

2" Watson doubts that ([the x: x murdered Smith])
(Holmes believes that (x is insane)))

2"' Watson doubts that (Holmes believes that ([the x: x murdered Smith]
(x is insane))).

In the light of Df I' resignation from *de dicto/de re* distinction appears to be premature. However, it points out the necessity of speaking about the copies of the operators. Consider example 2) in the above-mentioned interpretations outs the definitions of the syntactic *de re/de dicto* distinctions.

Also, speaking about the semantic interpretation of *de re* or *de dicto* **sentence** (which depends on the *salva veritate* interchangeability of co-referential terms appearing in the sentence) can be confusing. Let us consider a sentence: "Despite the fact that communism was overthrown in Poland, John thinks that communists hold significant power in Poland". Let us assume that the appropriate formal representation of this sentence has the following form: It-was-that[Communism was overthrown in Poland]&John-thinks-that[Communists hold significant power in Poland]. It is easy to notice that the proper name "Poland" is exchangeable *salva veritate* within the scope of the temporal operator "It-was-that" with any co-referential expression (e.g. "country on the Vistula River"), in contrast with the intentional operator John-thinks-that (John may not know that Poland is the country situated by the Vistula River).

[6] Kripke, S. A. "Speakers Reference and Semantics Reference", Contemporary Perspectives in the Philosophy of Language, Minneapolis Univeristy of Minesota Press p. 6–27.

Therefore, one constant may have *de re* meaning, while another one *de dicto* meaning in part of sentence Z.

This analysis indicates quite clearly a need for the relativisation of the semantic interpretation of *de re/de dicto* to the concept of the type of the operator in the given language, and not – as presented by McKay and Nelson – to the concept of a sentence.

Let us accept a definition thanks to which we will be able to deal with the sentences containing two or more intentional operators of different types.

Df II' the designator t has the *de re* meaning in the sentence Z relatively to the f formalization only when the name $f(t)$ is interchangeable with any co-referential name in $f(Z)$ *salva veritate*. Otherwise it is *de dicto*.

Taking into consideration the assumption of the rule of *salva veritate* exchangeability limitation, this definition equals the following expression: the t designator has in the sentence Z the *de dicto* meaning due to the f formalization only if the name $f(t)$ exists within the reach of a given intensional functor in/inside $f(Z)$; otherwise it is *de re*.

Let me illustrate the application of Df I' and Df II' with appropriate examples:

Let $Z =$ "There exists a man whom Adam considers to be a genius", and let $f(Z)$ be a formula: $\exists x\mathbf{Bel}_{\mathrm{a}}[Gx]$. In this meaning, according to Df' and Df II', there exist expressions: "considers" and "Adam" in the Z sentence (*de dicto* or *de re*)?

In the Z sentence the expression "considers" exists in the *de re* meaning, because in the $f(Z)$ formula there is a bound x variable free within the reach of f("considers") $=$ **Bel**. The designator "Adam", existing in Z, has the *de re* meaning when within the $f(Z)$ formula the constant f("Adam") is exchangeable *salva veritate* with any co-referential name constant.

(ii) The so-called metaphysical definition (Def III) speaks of the directness and indirectness of attributing properties to a given object. For an explanation of this difference, I will now come back to the example mentioned before. Let us assume that Sally believes Bill is happy. Can you say that Bill has the property of being the object of Sally's belief that he is happy? Even when assuming that Sally has reasons to suppose that there is exactly one person in Disneyland and that everybody who is in that place is happy, she has the *de dicto*, not the *de re* point of view (because she does not imagine any particular person, for example Bill).

It is known that the metaphysical relation of assigning properties to an object differs from the pragmatic relation of ascribing the properties to an object (by the subject). The so-called "metaphysical *de re/de dicto*" concerns of course the second of these relations. Definition III introduced by McKay and Nelson, can therefore be confusing. Because of this we will use here a more adequate definition, "the pragmatic distinction of *de re/de dicto*".

The pragmatic distinction applies to the way the designators (the denoting expressions are ressed) – the designator is used *de re* when its designation is directly given to the user in a defined context, otherwise it is used *de dicto*. These ways of using the designators may change in different sentence contexts, therefore not only is it necessary to relativise the concept of a person (the user), but also to the concept of a sentence in the distinction's definition. Finally, one can propose the following definition:

Df III':

> The sentence Z expresses the *de re* conviction of the object x with the f formalization only if in the $f(Z)$ formula there exists a simple name of the object x. The sentence Z expresses the *de dicto* conviction of the object x with the f formalization only if in $f(Z)$ there exists a single sentence function, fulfilled only by x.

According to the Df III', the sentence Z: "Bill is happy" expresses the *de re* conviction about Bill, because the name Bill f("Bill") exists in $f(Z)$ and it is a simple name. However, the sentence: "Each person in the Disneyland is happy" expresses the *de dicto* conviction about Bill, because Bill is the only person fulfilling the single sentence function "Dx" in the formula $f(Z)$ = "For each $x(Dx \rightarrow Hx)$".

IV

According to the accepted definitions, one should clearly distinguish three semiotic concepts of *de re* and, suitably, three concepts of *de dicto*: syntactic, semantic and pragmatic. Each of them played an essential role in modern philosophy, especially in discussions on the subject of the language philosophy.

It is not the sentence's category that is the range of the defined concepts, but the suitable semiotic categories, connected with appropriate relativisations. And so, the syntactic *de re* interpretation concerns the **operator** (this interpretation is relative to the syntactic concept of a sentence), the semantic – **designator** (this interpretation is relatives to the semantic concept of a language), and the pragmatic – **conviction of the object** (this interpretation is relative to the concepts of a sentence and an object).

BIBLIOGRAPHY

Bacon J. B., McCarty D. M. Ch. [1998]–, *Essentialism*, [in:] E. Craig (eds.), Routledge Encyclopedia of Philosophy, Taylor & Francis.

Biłat A. [1995]–, *Prawda i stany rzeczy*, UMCS.

Blackburn, S. [2004], *De re/de dicto*, [w:] Blackburn, S. (eds.), *Oksfordzki Słownik Filozoficzny*, Książka i wiedza, Warszawa.

Bocheński, I. M. [1961], *History of formal logic*, University of Notre Dame Press, Amsterdam.

Borkowski L. [1991], *Wprowadzenie do logiki i teorii mnogości*, Towarzystwo Naukowe Katolickiego Uniwersytetu Lubelskiego, Lublin.

Camp J. L. [1971], „Plantinga on *de re* and *de dicto*", *Nous.*

Ciecierski T., *Modalny argument Evansa przeciwko referencjalnemu rozumieniu deskrypcji*, „Studia Semiotyczne".

Cocchiarella N. [2009], *Formal ontology and conceptual realism*, Springer.

Cresswell M. J. [1968], Hughes E., *An introduction to modal logic*, G. Methuen.

Evans G., J. H. McDowell [1982], *The varieties of reference*, Oxford University Press.

Gallois A. [1998], *De re, de dicto* [w:] *Routledge Encyclopedia of Philosophy*, (eds.) E. Craig, Taylor & Francis.

Ingarden [1972] – Ingarden R., *Z teorii języka i filozoficznych podstaw logiki*, Warszawa, PWN.

Kallfelz W. M. [2007], *The role of pronouns in the de re/de dicto modal distinction.*

Kaplan D. [1969], *Quantifying in*, Synthese.

McKay T. [2005], Nelson M., *The de re/de dicto distinction*, http://plato.stanford.edu/; suplement of: *Propositional attitude reports*, *Stanford Encyclopedia of Philosophy*, http://plato.stanford.edu/entries/prop-attitude-reports/index.html

Meinong A. [2004], *Supozycje*, Adam Marszałek, Toruń.

Montague R. [2002], *The proper treatment of quantification in ordinary English*, [in:] P. Portner, B. H. Partee, *Formal semantics: the essential reading*, Wiley-Blackwell.

Novares C. D. [2004], *A medieval reformulation of the de dicto/de re distinction*, Logica Yearbook 2003, Prague, Philosophia.

Plantinga A. [2003], *De re et de dicto*, [in:] A. Plantinga, M. Davidson *Essays in the metaphysics of modality*, Oxford University Press US.

Quine W. V. O. [1979], *Intentional Revisited* [in:] P. A. French, T. E. Vehlig, H. K. Wettstein (eds.), *Contemporary perspectives in the philosophy of language*, Mineapolis MN: University of Minnesota Press.

Quine W. V. O. [1943], *Notes on existence and necessity*, Journal of Philosophy, 40.

Quine W. V. O. [1999], *Słowo i przedmiot*, przeł. C. Cieśliński, Aletheya, Warszawa.

Quine W. V. O. [1953], *Reference and modality*, [in:] *From a logical point of view*, New York 1953.

Quine W. V. O.[1976], *Quantifiers and propositional attitudes* [in:] *Ways of paradox and other essays*, Cambridge University Press.

Tomas Akwinatê [2004], *De propositionibus Modalibus*, Opera Omnia, Tom, XLII, Rzym, [in:] Novares, C. D., *A medieval reformulation of the de dicto/de re distinction*, Logica Yearbook 2003, Prague, Philosophia.

Woleński J. [2005], *Epistemologia*, PWN Warszawa.

Żegleń U. M. [1990], *Modalność w logice i filozofii*, Biblioteka Myśli Semiotycznej, Warszawa.

Andrzej Cieśluk
Maria Curie-Skłodowska University in Lublin
a.ciesluk.umcs@gmail.com

Karol Pąk
University of Białystok
Institute of Computer Science, Białystok, Poland

THE ALGORITHMS FOR IMPROVING AND REORGANIZING NATURAL DEDUCTION PROOFS

Abstract: It can be observed in the course of analyzing nontrivial examples of natural deduction proofs, either declarative or procedural, that the proofs are often formulated in a chaotic way. Authors tend to create deductions which are correct for computers, but hardly readable for humans, as they believe that finding and removing inessential reasoning fragments, or shortening the proofs is not so important as long as the computer accepts the proof script.
This article consists of two parts. In the first part, we present some types of unnecessary deductions and methods of reorganizing proof graphs in order to make them closer to good quality informal mathematical reasoning. In the second part, we describe tools implemented to solve the above-mentioned problems. Next, we demonstrate their usability by analysing statistical data drawn from the Mizar Mathematical Library.

1. Introduction

The databases of formalized mathematics are constantly growing and have achieved considerable sizes through the addition of numerous articles. Unfortunately, this enlargement of the databases is not always accompanied by the improvement of the quality of the formalization of the articles. Obviously, the legibility is a subjective notion and is an individual matter for different authors.

There are various reasons of database's illegibility. Firstly, the articles are written not only by the users who know the content of the database and are able to use it, but also by inexperienced users. These new users do not know the whole database and because of that they prove unintentionally theorems which have been already proved. Moreover, new users learn from simple and not sophisticated articles which were written in the older versions of the system. Because of this fact, inexperienced users are often repeating the old proof strategies and do not use all possibilities of the database and of the system for verifying correctness of the proofs.

The second reason of the illegibility consists of the fact that the majority of authors who develop and revise subsequent versions of a proof often add statements that might have been useful at some point of revision, but are not actually necessary for the final version of the proof. Finally, the lack of the legibility of the database results from the fact that some proofs are excessively large, often because of manual revisions which unwittingly introduce unnecessary items.

To avoid this problem the uniform criteria of the proofs' legibility must be elaborated and then the tools which will cause that the theses criteria will be fulfilled. There are two possibilities of doing it. Firstly, all articles can be improved manually, what requires thousands of hours of work. Secondly, it can be done with innovative programs which automatically shorten the considerable part of this manual and arduous work. It is possible because these auxiliary applications automatically standardize and shorten certain steps of the deduction. The user of this application can establish the hierarchy of the criteria which must be used during the reorganization of the proof by this program. The criteria described in our paper are illustrated by the Mizar system [6] in order to improve the quality of the Mizar Mathematical Library (MML) [12][1]. However, theses techniques are useful in every declarative system based on the natural deduction, created by S. Jaśkowski and F. B. Fitch [2, 3, 5].

The structure of this article is as follows. In Section 2 we present the abstract representation of proofs in the form of a graph, based on natural deduction of Jaśkowski. Subsequently, Section 3 presents selected types of excessive steps of deduction, which can appear in declarative formal proofs. This section also contains the description of problems occurring during the elimination of these steps of deduction. The knowledge of the Mizar system is not required to understand the problem even if the proofs which are described are represented in the Mizar style. In Section 4 we present the algorithms for the reorganization and elimination, and also we report the statistical results obtained through the MML database.

2. The Proof Graph

When proofs written in a formal system are considered, graphs and directed trees are often employed to express the intuitions connected with the reasoning. In this section, on the basis of the natural deduction system,

[1] Results of the improvement of the quality of the base MML, which contains more than 1000 articles verified the Mizar proof checker, are on the website [10]

we will attempt to give a definition of the proof graph, which will express the most general approach to this question.

For analyzing the relations between consecutive steps in formal proofs in the system of S. Jaśkowski, the interpretation of the proof graph as a directed graph (digraph) of the reference $\mathfrak{R}$ is often used, where the individual steps of reasoning are the vertices and the directed edges define the relations between an expression and a previously justified fact used as the justification for that expression.

More precisely, the expressions α and β are connected with a directed edge pointing directly from α to β, if and only if there exists set $\mathcal{R}$, such that $\alpha \in \mathcal{R}$ and the computer system can verify the correctness of β using premises from $\mathcal{R}$ Obviously, we would like that $\mathcal{R}$ not contain conclusions from β (then such a graph does not have directed cycles) and the number of $\mathcal{R}$ is possibly the least.

In such interpretation, the flow of information in the proof is well presented but the structure of the proofs is not preserved. In order to represent the structure, it is necessary to extend the analyzed graph with the relations describing the dependence between the expression and its proof.

Therefore, we can consider the proof $\mathcal{D}$ with possible nested lemma as a finite sequence of families of proof graphs without nested lemma. To construct such a sequence, the first family has to cosist of only one element (the proof graph of $\mathcal{D}$ with cut nested lemma). In the second step, we create a second family which consists of proof graph nested lemma cut from the proof graph of $\mathcal{D}$. In the third step, we cut the nested lemma from all proof graphs from the second family and we create from it the third family. We repeat the third step until the last family will not contain any nesting. Then, to regain the lost relations between the expression and its proof, we introduce an additional set of arcs, these we will call meta-edges. A meta-edge leads from β to α if and only if β is one of the steps in the reasoning of the proof of α.

The meta-edges describe relations between suitable vertices of families i and $i+1$. Obviously, the extended graph does not contain directed cycles, and for an arbitrary arc resulting from the reference connected directly from α belonging to the graph $\mathfrak{G}$ from the n-family to β belonging to the graph from the k-family we can say that

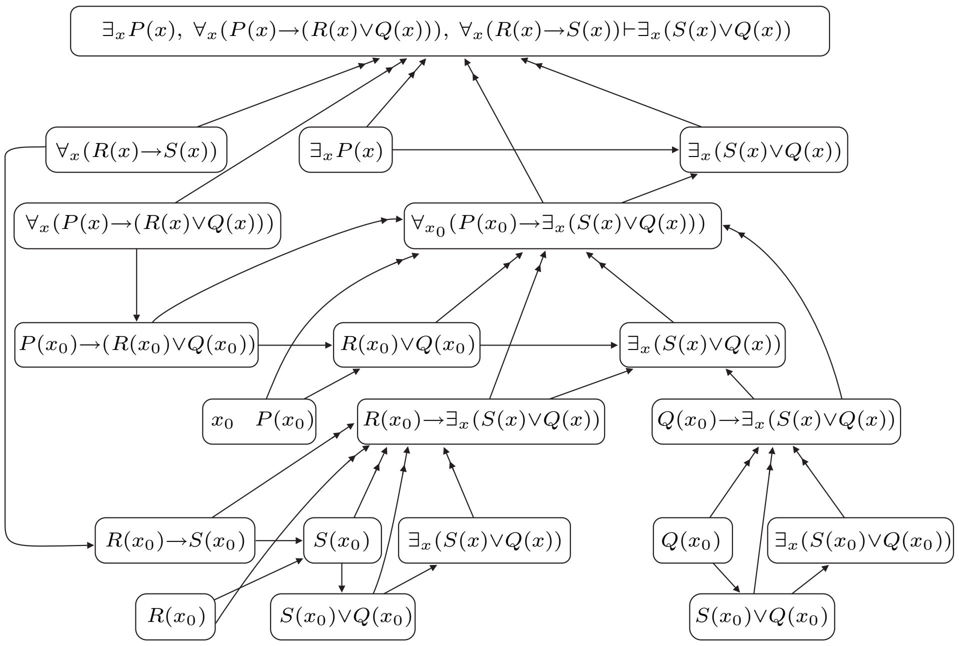
- $n \leq k$,
- there exists a path directed to meta-edges joining β and a chosen vertex in $\mathfrak{G}$.

This approach was created by Milewski [7] however, it does not describe the sufficient number of relations necessary to proof's reorganization.

As an illustration of the above-mentioned reasoning, let us consider the following example based on the Fitch notation of natural deduction. This exemple will demonstrate this lack of the relations.

1		$\exists_x P(x)$			premise
2		$\forall_x\,(P(x) \to (R(x) \vee Q(x)))$			premise
3		$\forall_x\,(R(x) \to S(x))$			premise
4	x_0	$P(x_0)$			assumption
5		$P(x_0) \to (R(x_0) \vee Q(x_0))$			$\forall_x$e 2
6		$R(x_0) \vee Q(x_0)$			$\to$e 5,4
7		$R(x_0)$	assumption	$Q(x_0)$	assumption
8		$R(x_0) \to S(x_0)$	$\forall_x$e 3	$S(x_0) \vee Q(x_0)$	$\vee$i$_2$ 7
9		$S(x_0)$	$\to$e 9,8	$\exists_x\,(S(x) \vee Q(x))$	$\exists_x$i 8
10		$S(x_0) \vee Q(x_0)$	$\vee$i$_1$ 9		
11		$\exists_x\,(S(x) \vee Q(x))$	$\exists_x$i 10		
12		$\exists_x\,(S(x) \vee Q(x))$			$\vee$e 6,7-9;7-11
13		$\exists_x\,(S(x) \vee Q(x))$			$\exists_x$e 1, 4-12

The interpretation of the above-mentioned formal proof as a graph with meta-edges has the following structure, where $\to$ arrows represent the references and $\twoheadrightarrow$ arrows illustrate the meta-edges.



$\exists_x P(x),\ \forall_x(P(x)\to(R(x)\vee Q(x))),\ \forall_x(R(x)\to S(x))\vdash\exists_x(S(x)\vee Q(x))$
$\forall_x(R(x)\to S(x))$
$\exists_x P(x)$
$\exists_x(S(x)\vee Q(x))$
$\forall_x(P(x)\to(R(x)\vee Q(x)))$
$\forall_{x_0}(P(x_0)\to\exists_x(S(x)\vee Q(x)))$
$P(x_0)\to(R(x_0)\vee Q(x_0))$
$R(x_0)\vee Q(x_0)$
$\exists_x(S(x)\vee Q(x))$
$x_0\quad P(x_0)$
$R(x_0)\to\exists_x(S(x)\vee Q(x))$
$Q(x_0)\to\exists_x(S(x)\vee Q(x))$
$R(x_0)\to S(x_0)$
$S(x_0)$
$\exists_x(S(x)\vee Q(x))$
$Q(x_0)$
$\exists_x(S(x_0)\vee Q(x_0))$
$R(x_0)$
$S(x_0)\vee Q(x_0)$
$S(x_0)\vee Q(x_0)$

The structure defined in this way enables the use of known facts and algorithms from the graph theory. In order to use this structure during the reorganization of reasoning we have to extend it with the omitted dependencies between steps of the type "the natural deduction way of reasoning", called "skeleton steps" in further parts of this article:

- the universal quantifier introduction and the introduction of the existential quantifier,
- the implication introduction and the indication of the thesis,
- the introduction of reasoning by cases.

There are also omitted relations between the place of introduction, use and redefinition of type of variables(e.g. between x_0 $P(x_0)$ and $P(x_0) \rightarrow (R(x_0) \vee Q(x_0)))$ and other dependencies characteristic to a particular system (e.g. the Mizar system). The extension of the definitions to other above-mentioned dependencies would limit the legibility of the definition, and it would require the reader's intuitive understanding of the dependencies which can appear in the system of natural deduction. However, it is possible to generalize the definition of a mode, were the definition is separated from the notion of the proof. It contains only three conditions.

Let us take a non empty set V, and disjoint families M, E of ordered pairs from V.

The Proof Graph

The structure $\mathfrak{P} := \langle V, M, E \rangle$ will be called the proof graph, if and only if,

1. $\mathfrak{M} := \langle V, M \rangle$ is a forest, i.e. a disjoint union of trees, in which every connected maximal tree is an arborescence, i.e a rooted tree with inverted direction (all arcs go in the direction from leaves to the root) ([4]).
2. An arbitrary arc (u, v) in the directed graph $\mathfrak{E} := \langle V, E \rangle$ fulfills the condition: every nearest successor of u is a predecessor of v in the forest $\mathfrak{M}$.
3. The directed graph $\mathfrak{G} := \langle V, M \cup E \rangle$ is acyclic.

The effect of inversion of direction consists in swapping the notions of child and parent (or predecessor and successor) in comparison to the natural trees.

To explain why we use the definition of the forest, in which every connected maximal tree is an arborescence with the root and inverted directions, let us define the auxiliary function $\mathfrak{l} : V \rightarrow \mathbb{N}$. The value of $\mathfrak{l}(v)$ is equal to the length of the directed path from v to the root plus one in the connected maximal tree, which contains v.

Then the value of the function $\mathfrak{l}$ represents vertices which belong to the suitable family from the sequence of graphs (more exactly, the vertex v

belongs to $\mathfrak{l}(v)$ the family). Additionally, graphs from individual families are defined by the notion of the nearest successor (inverted relation of siblings). Let us also notice that such introduction of the sequence of the families of graphs on the basis of forest $\mathfrak{M}$, causes that meta-edges connect statements of the family $i + 1$ with statements of the family i.

In graph theory the two conditions describe limitations imposed on the formal proof. Namely, the second condition says that the statements of reasoning of the provided theorem are invisible beyond that proof. Whereas, the third condition rejects existence of directed cycles which is equivalent to the application of provided theorem or conclusion from that theorem inside the proof.

Inside the family of arcs E we can distinguish three subfamilies

- $\mathcal{R}$ – the arcs resulting from the use of facts which have been already proved in the justification of another rule;
- $\mathcal{V}$ – arcs defining the dependence between the introduction of the variable and its use;
- $\mathcal{S}$ - arcs defining the order of skeleton steps.

Obviously, the above-mentioned families are not disjoint and they do not exhaust the set E.

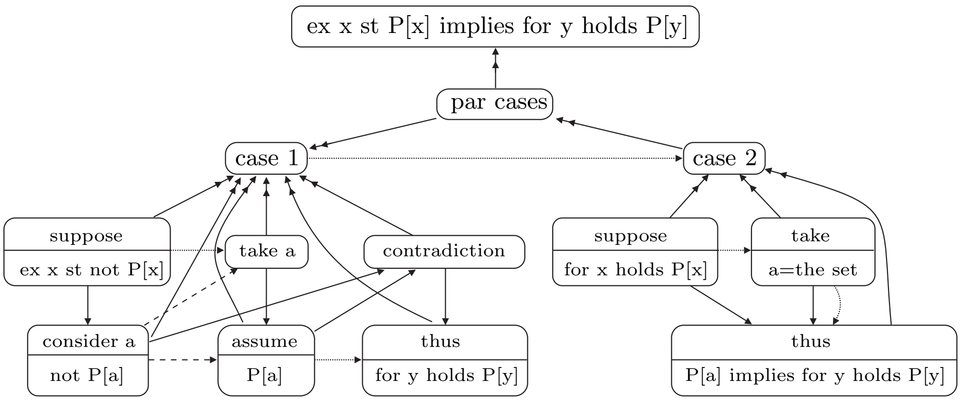
To illustrate the above-mentioned definition, let us consider the following example. We will represent the drinker's principle described in the article [14]. "This says that in every group of people one can point to one person in the group such that if that person drinks, then all the people in the group drink". The quoted proof is not indispensable for the Mizar system (an empty "semicolon" justification suffices to have it accepted by the checker), but a proof graph based on this reasoning illustrates well the subfamilies of the family E.

The reasoning in the Mizar style:

```
ex x st P[x] implies for y holds P[y]
proof
  per cases;
   suppose A0: ex x st not P[x];
      consider a such that A1: not P[a] by A0;
      take a;
      assume A2: P[a];
      A3: contradiction by A1,A2;
      thus A4: for y holds P[y] by A3;
   end;
   suppose A5: for x holds P[x];
      take a=the set;
```

```
        thus P[a] implies for y holds P[y] by A5;
      end;
    end;
```

The proof graph looks like:



where $\twoheadrightarrow$ arrows are subordinated to meta-edges and the continuous, dashed and dotted arrows are subordinated to suitable families $\mathcal{R}$, $\mathcal{V}$, $\mathcal{S}$.

The large number of arcs reduces the legibility of the graph, but enables to reconstruct the dependencies in reasoning. Such a graph form generally does not enable an unambiguous reconstruction of the order of reasoning steps. In the above-mentioned example, when we close transitively the graphs of individual cases, we obtain complete graphs, what one can easily prove, impose an unambiguous order of steps, however the removal of the arc "case 1 $\rightarrow$ case 2" enables changing the order of two cases.

3. The Chosen Forms of Inessential Steps of Deduction

The presentation of unnecessary types of deduction in declarative formal proofs is hard to explain, but easy to illustrate. The basic types of redundant steps of deduction, such as:

– unnecessary references used in the justification of an expression,
– references which can be replaced by all references used to justify statements they point to,
– steps of deduction which are not used in any proof leading to the fact that some thesis or steps of deduction have no references pointing to them (vertices for which every thesis is not a successor in the proof graph),

– steps of deduction which can be totally replaced by external references, which can be found in justifications of these steps of deduction.

are already resolved (see [8]) and in this aim the necessary auxiliary applications (e.g. Relprem, Relinfer, Inacc or Trivdemo) have already been created.

Theses auxiliary applications have enabled to discover the next important problem that was not resolved. We can qualify it as "covered with &". To describe this problem let us consider two deductions merged by the author into one reasoning.

```
$\alpha_1$ implies $\alpha_n$    $\beta_1$ implies $\beta_n$    $\alpha_1$ & $\beta_1$ implies $\alpha_n$ & $\beta_n$
proof                proof                proof
  assume $\alpha_1$;         assume $\beta_1$;         assume $\alpha_1$ & $\beta_1$;
  then $\alpha_2$;           then $\beta_2$;           then $\alpha_2$ & $\beta_2$;
  then $\alpha_3$;           then $\beta_3$;           then $\alpha_3$ & $\beta_3$;
  ⋮                  ⋮                  ⋮
  then $\alpha_n$;           then $\beta_n$;           then $\alpha_n$ & $\beta_n$;
  hence thesis;      hence thesis;      hence thesis;
end                  end                  end
```

Such a merge does not cause mistakes in the reasoning, but can contain some repeated expressions (if the lengths of deduction are different). Let us also notice that in such a proof, e.g. to prove the rule α_{i+1} an unnecessary step, β_i, is used. Moreover, the rule α_i & β_i can be necessary in reasoning despite the fact that one of the steps α_i or β_i is not necessary.

Another problem which can occur in the merged parallel deduction, consists of removing by the author some part of the thesis (e.g. β_n) without changing the assumptions and proof. Such change limits, in an important way, the statement, however the proof is still correct. None of the above-mentioned auxiliary applications can detect such cases. It is easy to notice that the creation of algoritm that could automatically find such cases is incomparably more difficult than the creation of the above-mentioned auxiliary application. Such algoritm should consider all deductions and not only one step as it is done by the auxiliary applications created until now.

Our auxiliary application enables resolving of much more complicated problems, such as the following example illustrates:

```
$\alpha_1$ & $\beta_1$ implies $\alpha_4$
proof
  assume A1: $\alpha_1$ & $\beta_1$;
  $\alpha_4$ & $\beta_4$
   proof
```

```
          $\alpha_2$ & $\beta_2$ by A1;
          then $\alpha_3$ & $\beta_3$;
          then $\alpha_4$ & $\beta_4$;
          hence thesis;
         end;
        hence thesis;
      end;
```

Our auxiliary application has found numerous cases of such problem not only in lemma but also in 541 theorems in MML (it means, on average, one in every 100 theorems).

The presented solution which consists of breaking chosen conjunctions and removing unnecessary steps that were created in this way, in the whole base MML, is the subject of Section 4 of this article. Is too easy to notice that the majority of the steps of deduction which contain conjunctions has become illegible. The analyzed proof of the step α_1 `&` β_1 `implies` α_n `&` β_n, can look as follows:

```
$\alpha_1$ & $\beta_1$ implies $\alpha_n$ & $\beta_n$
proof
  assume that A1: $\alpha_1$ and A2: $\beta_1$;
  A3: $\alpha_2$ by A1;
  A4: $\beta_2$ by A2;
  A5: $\alpha_3$ by A3;
  A6: $\beta_3$ by A4;
  A7: $\alpha_4$ by A5;
  ⋮
```

In order to present this problem precisely, let us consider a simple Mizar-style proof, whose proof graph well illustrates typical situations met during the reorganization of the proof.

```
theorem
  i in Seg n implies i+m in Seg(n+m)
proof
  assume i in Seg n;
  then 1 <= i & i <= n & i <= i+m by NAT_1:11,FINSEQ_1:3;
  then 1 <= i+m & i+m <= n+m by XREAL_1:9,XXREAL_0:2;
  hence thesis by FINSEQ_1:3;
end;
```

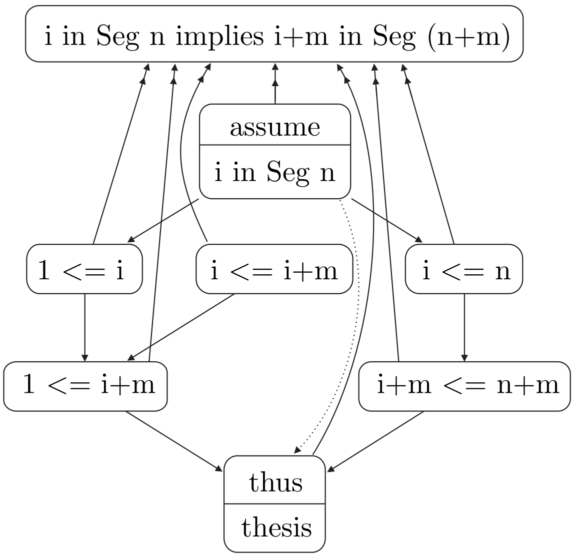
were `i, n, m` are natural numbers and `Seg n={1,...,n}`. We can also analyze the above-mentioned reasoning in a different system, e.g. in the Isabelle style [11]:

```
lemma
  fixes n i m :: nat
  assumes a: "i \<in> {k :: nat . 1 <= k & k <= n}"
  shows "i + m \<in> {k :: nat . 1 <= k & k <= n + m}"
proof -
  have "1 <= i & i <= n & i <= i+m" using a by auto
  then have "1 <= i+m & i+m <= n+m" by auto
  then show ?thesis by auto
qed
```

Having broken all conjunctions of the reasoning and having simplified the lists of the reference, we obtain:

```
theorem
  i in Seg n implies i+m in Seg (n+m)
proof
  assume A1:i in Seg n;
  then A2:1 <= i by FINSEQ_1:3;
  A3:i <= n by A1,FINSEQ_1:3;
  i <= i+m by NAT_1:11;
  then A4:1 <= i+m by A2,XXREAL_0:2;
  i+m <= n+m by A3,XREAL_1:9;
  hence thesis by A4,FINSEQ_1:3;
end;
```

and proof graph:



We have left in the above-mentioned graph only the arcs resulting from references in order to enable simple understanding of the next steps of reasoning and finding references used in consecutive steps.

The presentation of the above-mentioned graph in equally legible way in the system of natural deduction is a crucial idea of the proof's reorganization. This essential point consists of determining the criteria which can improve the legibility of formal proofs in the system of natural deduction. Having analyzed the different opinions of users of database we propose the following four criteria of legibility of deduction:

1. maximization of the length of the paths in which every consecutive justification should refer to a previous line and, if it is possible, to a minimal number of different labels,
2. minimization of the quantity of introduced labels,
3. minimization of the total length of jumps to distant, previously justified facts,
4. presentation, in the coherent entirety of the proof, of reasoning steps which in the proof graph locally create the sub-deduction.

As it has been mentioned in the introduction, establishing of the degree of importance of particular criteria is a controversial matter. So we created a flexible application which can be used even by users with opposed criteria's hierarchy of legibility. The users chose the most often two first criteria or the third one as the predominant. If we compare the results obtained for theses two different situations, we get two various figures, which are presented below:

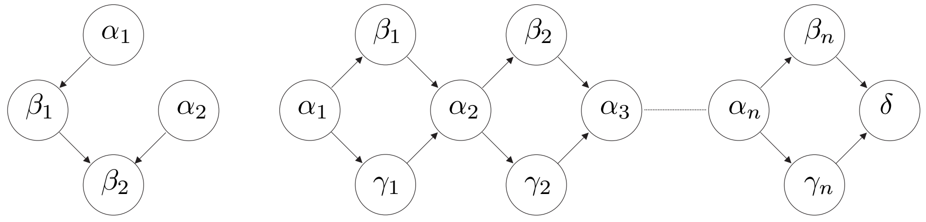
```
theorem
  i in Seg n implies
  i+m in Seg(n+m)
proof
 A1: i<=i+m by NAT_1:11;
 assume A2: i in Seg n;
 then i<=n by FINSEQ_1:3;
 then A3: i+m<=n+m by XREAL_1:9;
 1<=i by A2,FINSEQ_1:3;
 then 1<=i+m by A1,XXREAL_0:2;
 hence thesis by A3,FINSEQ_1:3;
end;
```

```
theorem
  i in Seg n implies
  i+m in Seg(n+m)
proof
 assume A1: i in Seg n;
 then A2: 1<=i by FINSEQ_1:3;
 i<=n by A1,FINSEQ_1:3;
 then A3: i+m<=n+m by XREAL_1:9;
 i<=i+m by NAT_1:11;
 then 1<=i+m by A2,XXREAL_0:2;
 hence thesis by A3,FINSEQ_1:3;
end;
```

In the first case, we find two chains of three elements, whereas in the second case there is only one chain of three elements and two chains containing two elements. The first criterion does not define whether it is more natural to create one maximal chain with many, often one-element chains, or to formulate several chains of average length, without chains containing only one-element.

Having analyzed the total distance of jumps between a label and its use, we observe that creating longer chains enlarged the total distance of jumps exactly by 2.

If we count the number of labels in the above-mentioned reasoning, we see that there are exactly three in both cases. We can prove that this is the minimal number of labels for this proof. Moreover, a maximal anti-chain in the transitive, closed proof graph of this reasoning has at most three elements. This dependence is often a loose relationship. The number of labels can be just a little smaller, e.g. in graphs of references which look as in the first example, or many times larger (the second example).



(in the second example, an arbitrary maximal anti-chain has at most two elements, but the number of labels is estimated by $2 \cdot n$).

The best way of estimating the number of labels in a reference graph consists of counting the vertices whose outdegree is at least two; and the number of the arcs (u, v) for which the indegree of v is at least two and the vertex v does not have yet the label (the number of entering arcs without a label is at the most). If we take into consideration all arcs in the proof graph, it enlarges, in the general case, the number of labels counted in this way.

4. Auxiliary Applications and Statistical Results

The problems described in the previous section can be solved with two independent sets of programs. The aim of the first one consists of finding and removing as many as possible unnecessary steps of reasoning hidden in "&". To this end, the existing utilities have been extended with five programs, which we describe below. The aim of the second sort of programs is the reorganization of the order of proof steps. This sorting was made with the application SortItem, which preserves correctness of the reasoning and relations in the proof graph. Statistical results were obtained on the MML database version `4.121.1054` and were introduced to the version `4.127.1060` [10].

BreakBinaryAnd

The application breaks in a simple way (with some not so important exceptions) all statements that contain a conjunction. The application changes the sequence of expressions joint with the conjunction into the sequences of consecutive steps of reasoning, which include the suitable elements of that sequence of expressions and the identical list of the reference used in the reasoning. Breaking all of the above-mentioned conjunctions has enabled finding in MML of 767139 unnecessary references in the justifications, which caused a transmission of unnecessary sequences of conjunctions. Moreover, we found 39745 steps in deductions which were unnecessary for their correctness and we could remove it.

DelBlock

The application significantly breaks the proofs of conjunction statements which do not contain implicit universal quantifiers. Moreover, the deductions cannot contain the elimination of a universal or existential quantifier and the introduction of reasoning per cases.

We can describe the transformation made in the general case in the following style:

```
Lab1: α & β
 proof
  ...              ...
  thus α;          Thus1: α;
  ...              ...
  thus β;          Thus2: β;
end;             Lab1: α & β by Thus1,Thus2;
```

RenInfer

The application collaborates with the auxiliary application RELINREF. The program changes selected references for which Relinfer reports the message "604" (it means references which can be replaced by all references used to their justification). The program creates a list of labels sorted using three criteria with decreasing signification, such as:

1. the number of references to a particular label without the message "604" is minimal,
2. the number of all references to a particular label is minimal,
3. the number of references with message "604", used as justification for the statement assigned to this label, is minimal.

Then, for a label selected in this way, the program replaces some of its uses by all references used to justify the expressions joined with the label.

Obviously, every time such a label is chosen, the labels of the statement whose list of justifications will be modified, are ignored in the next search. The described algorithm does not remove all "604" messages. After one use of the program, on average 88,7% of messages "604" is removed, and all theses messages are removed on average after 1,532 uses of this program.

TrivConsider

This application found 6154 cases in which removing introduced variables was possible using construction "consider" (the incorrectness in modified reasoning occurred only in two cases). Such introduction of existential quantifiers enabled finding new unnecessary steps in the deductions. It is interesting to notice that after removing unnecessary steps, the application could find other 59 cases.

MergeItems

The program finds statements always used together in the reasoning, and then it tries to merge them into a conjunction. Obviously, none of the statements can be a successor of the other one in the proof graph. To avoid creating long list of references, which can lengthen the time needed to verify joint statements, in the justification of joint statements it is required that the lists of the references are compatible. The level of comptibility is determined by users.

The use of the five above-mentioned programs enabled finding in MML the theorems with unnecessary assumptions. Removing these assumptions enabled finding next unnecessary steps and statements in the deduction. The statistical results are presented in the following table:

Stages	Unnecessary references	Message "604"	Unnecessary inferences	s Altered articles	Altered theorems
1	755196	37304	38944	1017	461
2	1640	23	633	118	64
3	596	2	168	55	16

SortItem

The program creates a proof graph for a particular article and, on the basis of it, it reorganizes the order of statements in reasoning. The algorithm of reorganization is based on a successive recurrent joining of sequences of sub-deduction chosen with the imposed criteria which do not cause the conflict in the graph (i.e. no vertex from first reasoning is the predecessor of a vertex from the second reasoning). Using a greedy algorithm, which solves the problem of making the locally optimal choice at each stage, often does not enable to find the global solution, but in contrast to other algorithms, enables a oherent presentation of local sub-deduction.

Let us introduce the auxiliary functions for a proof graph, in order to describe the basic criteria in a legible way $\langle V, M, E\rangle$.

Let us take a subset A of the set E. We define functions:

$$deg_A^-(v) = |\{w : w \in V \ \wedge \ (w, v) \in A\}|,$$

$$deg_A^+(v) = |\{w : w \in V \ \wedge \ (v, w) \in A\}|,$$

$$I_A(F) = |\{(i, j) : 1 \leq i, j \leq n : \wedge (F(i), F(j)) \in A\},$$

where v is an arbitrary vertex, and F is an n-length sequence of vertices from the family E.

In this case the minimization of the number of labels on the base of two basis criteria with decreasing significance is important (were F_i is i-sequence of the n_i-length).

1. The aim of of the first criterion is finding the pairs of sequences for which the following condition is fulfilled: $(F_1(n_1), F_2(1)) \in R$, and than it chooses the pairs for which the values of function
$$deg_R^+(F_1(n_1)), \left(\sum_{i=1}^{n_1} deg_R^+(F_1(i))\right) - I_R(F_1)$$
are minimal.
2. The aim of the second criterion is finding the two pairs of sequences for which the number of dysfunctions (i.e. the increase of total distance between places of introduction of the label and its uses) created after merging these sequences is minimal. The number of dysfunctions is described on the basis of the relation:
$$n_1 \cdot \left(\sum_{i=1}^{n_2} deg_R^-(F_2(i))\right) + n_2 \cdot \left(\sum_{i=1}^{n_1} deg_R^+(F_1(i))\right) - (n_1 + n_2) \cdot I_R(F_1^\frown F_2).$$

Apart from the main criteria there are several auxiliary criteria, quite difficult to describe. Thanks to all these criteria the order of writing the proof tree becomes much more unequivocal.

Statistical Results

Having measured (in tokens) the length of articles we observe that despite an important number of modifications, the length of articles did not change in a significant way (the articles were reduced on average by 0,31%). On the other hand, having analyzed the new form of the articles, we observed that the modification caused a reduction of the length of formulas on average by 3,12%, but the length of the references was extended by 3,7%. The verification time of an article (time of mizaring) is another measure which can be used to describe the modifications. This time on average was reduced by 5,13%.

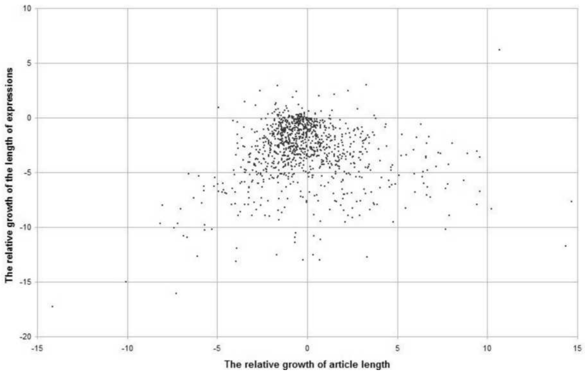


Fig. 1. The change of article length

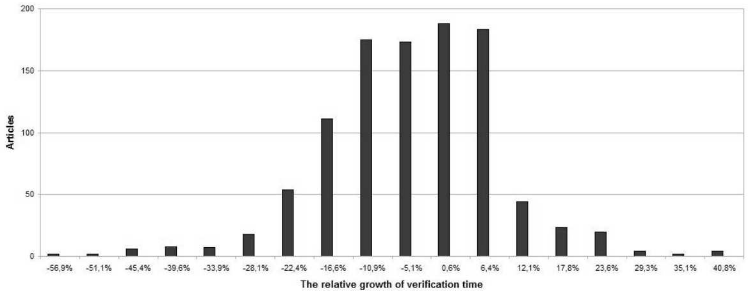


Fig. 2. The change of verification time

Final Remarks

Thanks to algorithms presented in our paper many advantages have been obtained. Firstly, our algorithms constitute one of the first (and the first in the Mizar system) fully automatic ways of standardizing the reasoning's structure that is based on the imposed criteria in the aim of legibility's improvement (so far, it was done mainly manually). Secondly, the reasoning's structure was improved in an important way and the particular steps of the deduction have became more readable for users of the database. In consequent, using the database has become easier. Thirdly, our algorithm of the proof's reorganization can be used independently to introducte of the results of different experiments which have not been able to maintain the readable proof's structure. It is really important because databases are public and the perseverance of the high level of quality is a priority. It does not change the fact that our algorithms are the first complex tool which enable both the shortening of the database and than perseverance and even improvement of its legibility. Fourthly, an average time of the verification of articles of the database has been shortened. And last but not least, shortening of the number of steps of the deduction (about 3%) and the reduction of the number of assumptions in about 1% theorems may does not seem at a first glance an impressive result. However, the time necessary to obtain such effects manually is comparable with time of manual calculation of 35 decimals of π by Ludolph van Ceulen.

R E F E R E N C E S

[1] E. Bonarska, *An Introduction to PC Mizar*, Fondation Ph. le Hodey, Brussels, 1990.

[2] F. B. Fitch. *Symbolic Logic: an Introduction.* The Ronald Press Co., New York, 1952.

[3] S. Jaśkowski, *On the Rules of Supposition in Formal Logic*, Studia Logica I, 1934, Warszawa Reprinted in Polish Logic, ed. S. McCall, Clarendon Press, Oxford 1967.

[4] B. Jorgen, G. Gregory, *Digraphs: Theory, Algorithms and Applications*, Springer, ISBN 1-85233-268-9, (2000).

[5] W. Marciszewski, *A Jaśkowski-Style System of Computer-Assisted Reasoning*, Philosophical Logic in Poland, Kluwer, 1993.

[6] R. Matuszewski, P. Rudnicki, *MIZAR: the first 30 years*, Mechanized Mathematics and Its Applications, 4 (**1**), pp. 3–24, 2005.

[7] R. Milewski, *Algorithms analyzing formal deduction support systems* – PhD thesis, The Computer Science Faculty of Białystok Technical University, Białystok 2008.

[8] R. Milewski, *New Auxiliary Software for MML Database Management Mechanized Mathematics and Its Applications*, ISSN 1345-8272 1345-8272, Vol. 5, No. 2: 1–10, 2006.

[9] R. Milewski, *Transformations of MML Database's Elements Lecture Notes in Computer Science*, Springer-Verlag, ISSN 0302-9743, Vol. 3863/2006: 376–388, 2006.

[10] *Mizar Home Page*, http://mizar.uwb.edu.pl/.

[11] L. C. Paulson, *The Isabelle Reference Manual*, 2000.

[12] P. Rudnicki, *An Overview of the Mizar Project*, Proceedings of the 1992 Workshop on Types for Proofs and Programs, Chalmers University of Technology, Bastad, 1992.

[13] A. Trybulec, *Some features of the Mizar language*, Presented at a workshop in Turin, Italy, 1993.

[14] F. Wiedijk, *Mizar Light for HOL Light*, Proceedings of the 14th International Conference on Theorem Proving in Higher Order Logics, p. 378–394, September 03–06, 2001.

Karol Pąk
University of Białystok,
Institute of Computer Science,
Białystok, Poland
pakkarol@uwb.edu.pl

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 22 (35) 2010

Andrew Schumann
Belarusian State University (Minsk)
University of Białystok

UNCONVENTIONAL PROBABILITIES AND FUZZINESS IN CADIAG'S COMPUTER-ASSISTED MEDICAL EXPERT SYSTEMS

Abstract: In the paper I sketched some applications of non-Kolmogorovian probabilities (including complex- and non-Archimedean-valued) and non-Archimedean fuzziness in the CADIAG expert system. This work I started this year.

1. Introduction

Recently, a huge number of implementations and developments of knowledge-based systems in medicine aiming at providing support for physicians in the decision-making process has been proposing. One of them is CADIAG's computer-assisted medical expert systems. The latter are fuzzy consultation systems in that fuzzy methods for almost all knowledge processing tasks are used: in them fuzzy inputs, operates on fuzzy sets with fuzzy rules, and produces fuzzy sets as output are accepted. The aim of CADIAG is to help in medical diagnosis. Notice that "medical diagnosis is the art of determining a person's pathological status from an available set of findings. Why is it an art? Because it is a problem complicated by many and manifold factors, and its solution involves literally all of a human's abilities including intuition and the subconscious" [8]. CADIAG is aimed to provide diagnosis with automatic inferences, i.e. to transform the art of diagnosis into knowledge.

In CADIAG-1, there were 187 diagnoses and 1,213 symptoms, signs, and findings within the rheumatological differential diagnostic group CADIAG-1/RHEUMA [5]. On the other hand, CADIAG-2s knowledge base contained more than 20,000 rules expressing causal relationships between patient's symptoms, signs, laboratory test results, clinical findings and diagnoses on the basis of the patient data and the knowledge base [3]. These relationships are formulated as IF–THEN rules, between symptoms, signs, test results and clinical findings on the one hand and diagnoses on the other hand. Since

CADIAG-2 fuzzy logic has been used for defining IF–THEN rules. In particular, fuzzy logic has been aiming to (i) perform a formal consistency check of the rules of CADIAG-2 and CADIAG-4, (ii) formally justify the choice of the operators (t-norms) and the way of combining the rules of the systems and (iii) allow for computing satisfactory results in the presence of incomplete information.

CADIAG-2 and CADIAG-4 systems employ fuzzy set theory by using T-norm-based fuzzy logics to capture: uncertainty as to whether a patient's symptoms (signs, laboratory test results) are pathological or normal; uncertainty as to whether symptoms necessarily have to occur with a disease; and uncertainty as to whether symptoms sufficiently confirm or exclude a diagnosis [9]. A consequence relation for graded inference of those systems may be regarded within the frame of infinite-valued Łukasiewicz semantics [6].

In next sections we are proposing to apply hybrid fuzzy logics and hybrid probability logics to describe the knowledge representation and the knowledge acquisition procedures that support medical experts to add, edit and update their knowledge.

2. Non-Kolmogorovian Probabilities

Kolmogorov's probability theory is defined on $\{e_1, e_2, \ldots\} = E$, a collection of elements called elementary events, and F, a set of subsets of E called random events. This theory is based on the following axioms:

- F is a field of sets.
- F contains the set E.
- A non-negative real number $\mathbf{P}(A)$, called the probability of A, is assigned to each set A in F.
- $\mathbf{P}(E)$ equal 1.
- If A and B have no elements in common, the number assigned to their union is $\mathbf{P}(A \cup B) = P(A) + P(B)$; hence, we say that A and B are disjoint; otherwise, we have $\mathbf{P}(A \cup B) = \mathbf{P}(A) + \mathbf{P}(B) - \mathbf{P}(A \cap B)$.

Kolmogorov's probabilities have a huge number of applications to information science and computer technologies. For instance, such probabilities were applied in conditional probability interpretation of CADIAG-2's inferences, namely in identifying degrees of confirmation with probabilities and the rules themselves with conditional probabilistic statements [7]. This interpretation was proposed in terms of degrees of belief of the doctor (or doctors) on the truth of the consequent given that the antecedent of the rule holds is also possible though. However, it is possible to set non-Kolmogorovian probabilities, which have much better opportunities to express real life probabilities and to simulate medical data.

2.1. Probabilities in Complex Dimension

Now let us define probabilities on $\mathbf{C}$, the set of complex numbers. Each member of $\mathbf{C}$ is a number comprising a real and imaginary part. Let $\mathbf{R}_C$ be the set of real numbers playing role of real parts of complex numbers and $\mathbf{I}_C$ the set of imaginary numbers playing role of imaginary part. We add the following statements to Kolmogorov's list of axioms

- Let $\mathbf{P_I}(A) = i(1 - \mathbf{P}(A))$, where $i^2 = -1$, be the probability taking values in $\mathbf{I}_C$. It is an imaginary part associated to the probability $\mathbf{P}(A)$ of event A taking values in $\mathbf{R}_C$. The probability $\mathbf{P_I}(A)$ is called imaginary.
- The complex number $c = \mathbf{P_C}(A) = \mathbf{P}(A)+\mathbf{P_I}(A) = \mathbf{P}(A)+i(1-\mathbf{P}(A))$ has an absolute value $|c|^2 = \mathbf{P}(A)^2+(1-\mathbf{P}(A))^2$. The probability $\mathbf{P_C}(A)$ is called complex.

Interpretation of complex probabilities:

Complex probabilities allow us to consider hidden variables, called imaginary probabilities, as unknown forces causing the output as well as known forces, called real probabilities. As a result, it is possible to examine the same phenomena not only within realized possibilities, but also taking into account unrealized possibilities.

Advantages of complex probabilities:

1. We could calculate the degree of knowledge by considering $|c|^2$ as an appropriate degree. Let us exemplify this opportunity by the game of coin tossing. Hence, P('getting head') $= 1/2$ and $\mathbf{P}$('getting tail') $= 1/2$. Then $|c|^2 = 1 - 2 \cdot P$('getting head') $\cdot (1 - P$('getting head')$) = 1 - (2 \cdot 1/2) \cdot (1 - 1/2) = 1/2$.
2. $\mathbf{C}$ can be presented as a two-dimensional real vector space. Continuing in the same way, we could define complex probabilities as 2-dimensional quantities, too. In the vector representation, the rectangular coordinates are typically referred to simply as x and y and we may offer geometric images of probabilities.

In CADIAG's fuzzy medical consultation system, uncertainty and probability concerning the confirmation or exclusion of diagnoses D_j in a patient P is modelled by degrees of their compatibility. *Using complex pro-*

babilities and fuzziness, we may present the confirmation or exclusion of diagnoses as a real and imaginary parts of complex quantities. This way is much simpler.

2.2. Probabilities in non-Archimedean Dimension

Now consider non-Archimedean probabilities. The ultrapower $\Theta^I/\mathcal{U}$, where $\mathcal{U}$ is ultrafilter that contains all complements for finite subsets of I, is said to be a proper nonstandard extension of Θ and it is denoted by $^*\Theta$. There exist two groups of members of $^*\Theta$: (1) functions that are constant, e.g. $f(\alpha) = m \in \Theta$ for an infinite index subset $\{\alpha \in I\} \in \mathcal{U}$; a constant function $[f = m]$ is denoted by $^*m$; (2) functions that are not constant. The set of all constant functions of $^*\Theta$ is called standard set and it is denoted by $^\sigma\Theta$. The members of $^\sigma\Theta$ are called standard. It is readily seen that $^\sigma\Theta$ and Θ are isomorphic: $^\sigma\Theta \simeq \Theta$. If Θ was a number system, then members of $^*\Theta$ are called hypernumbers. If $\Theta = \mathbf{R}$, then the new numbers are called hyperreal numbers and if $\Theta = \mathbf{Q}$, then the new numbers are called hyperrational numbers.

Define a structure $\langle \mathcal{P}(^*\Theta), \wedge, \vee, \neg, {}^*\Theta \rangle$ as follows (1) for any $A, B \in \mathcal{P}(^*\Theta)$, $A \wedge B = \{f(\alpha)\colon f(\alpha) \in A \wedge f(\alpha) \in B\}$, (2) for any $A, B \in \mathcal{P}(^*\Theta)$, $A \vee B = \{f(\alpha)\colon f(\alpha) \in A \vee f(\alpha) \in B\}$, (3) for any $A \in \mathcal{P}(^*\Theta)$, $\neg A = \{f(\alpha)\colon f(\alpha) \in {}^*\Theta \backslash A\}$. Then a structure $\langle \mathcal{P}(^*\Theta), \wedge, \vee, \neg, {}^*\Theta \rangle$ is not a Boolean algebra if $|\Theta| \geq 2$, because the set $\mathcal{P}(^*\Theta)$ is not closed under intersection. Indeed, by definition of non-standard extension, some elements of $^*\Theta$ have a non-empty intersection that does not belong to $\mathcal{P}(^*\Theta)$, i.e. they are not atoms of $\mathcal{P}(^*\Theta)$ of the form $[f]$. The same situation is observed in the field $\mathbf{Q}_p$ of p-adic numbers.

Consequently, there is a problem how it is possible to define probabilities on non-Archimedean structures if we have no opportunity to put them on a field of subsets. A. Khrennikov who considered non-Archimedean probability theory on the set of p-adic numbers proposed a semi-algebra that is closed only with respect to a finite unions of sets, which have empty intersections. However, there is a better way if we get non-Archimedean probabilities on a class $\mathcal{F}(X)$ of fuzzy subsets A of the set X of non-Archimedean numbers (hypernumbers or p-adic numbers).

Interpretation of non-Archimedean probabilities:

Non-Archimedean probabilities are defined within non-Archimedean metric and taking into account that non-Archimedean structures are non-well-founded (the set-theoretic foundation axiom is violated there), in non-Archimedean probability theory we assume that reality is non-well-founded too and, as an example, we could accept phenomena with circular relations and calculate their probabilities.

Advantages of non-Archimedean probabilities:

1. we could set multi-hierarchical probabilities (higher-order probabilities), e.g. multi-hierarchical Bayesian networks,
2. we could analyze probabilities of overlapping phenomena (we are dealing with fuzzy probabilities there),
3. this theory allows us to regard probabilities of non-well-founded phenomena (phenomena with feedback and circular relations).

Medical cause-effect relationships (the relations between diagnoses and their symptoms) are hardly ever one-to-one, because diagnoses often share an overlapping range of symptoms[1] *and furthermore symptoms may assume feedback and circular relations among themselves.* As a result, fuzzy probabilities, on the one hand, and non-well-founded probabilities, on the other hand, are suitable tools for analyzing medical diagnosis. Notice that now there exist many researches showing advantages of non-Archimedean mathematics in mathematical modeling of natural (biological or medical) systems.

3. Non-Archimedean Fuzziness

Non-Archimedean valued fuzzy logics (including the p-adic case) may find many applications to rule-based systems and demonstrate their importance as a powerful design methodology. So, on the one hand, novel logics are complete and, on the other hand, they have been extended to *infinite hierarchy of fuzzy sets*, therefore they have more design degrees of freedom than do conventional fuzzy logics with values distributed in the standard unit interval $[0, 1]$.

Usually, the medical knowledge is presented as a flat relation between the set of symptoms and the set of diseases. Let S and D be the set of symptoms and diagnoses, respectively, and $\mathcal{F}(S)$ and $\mathcal{F}(D)$ be fuzzy power sets of S and D. Medical knowledge is then the relationship between symptoms and diagnoses expressed by a fuzzy relation $\mathcal{R} \subset \mathcal{F}(S) \times \mathcal{F}(D)$. However, there are tasks, when medical data need to be interpreted as a type-2 fuzzy set.

[1] For instance, we observe the overlapping syndromes in a patient in whom, on the one hand, "haemolytic anaemia, positive LE cells" and "proteinuria" and, on the other, "sclerodactyly, impaired oesophageal motion" and "lung fibrosis" are found, because there are symptoms present of both "systemic lupus erythematosus" and "progressive systemic sclerosis".

In this case we could use fragments of multi-hierarchies of non-Archimedean fuzzy logics.

Their multi-hierarchies are connected to properties of non-Archimedean numbers. Geometrically, we can imagine a system of these numbers in two ways:

- in case of hyperreal or hyperrational numbers they may be regarded as a homogeneous infinite tree with $[0, 1]$-branches splitting at each vertex,
- in case of p-adic integers as a homogeneous infinite tree with p-branches splitting at each vertex.

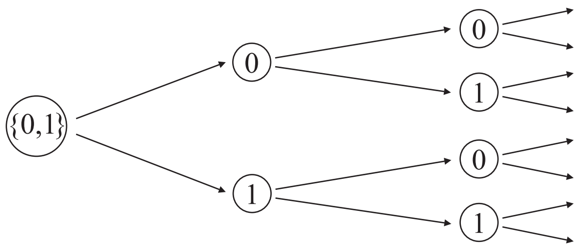


Figure 1. The 2-adic tree

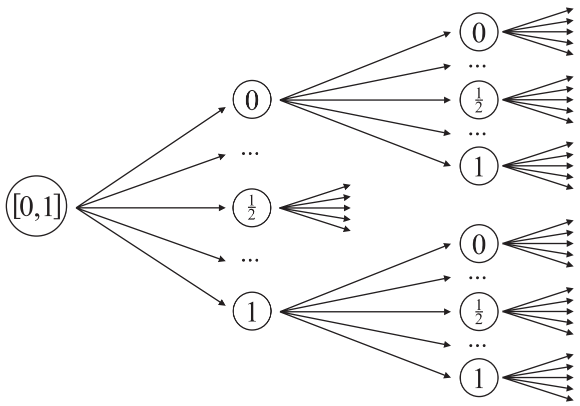


Figure 2. The non-Archimedean tree

The first one is said to be *non-Archimedean tree*, the second one *p-adic tree*. For instance, a 2-adic tree is pictured in Fig. 1 and a non-Archimedean tree with branches whose number runs over the set $[0, 1]$ in Fig. 2. This geometrical presentation allows us to consider infinite hierarchies of conditional fuzzy (resp. finitely many-valued) data described in the standard form as that distributed in the set $[0, 1]$ (resp. $\{0, \ldots, p-1\}$). In order to exemplify this feature, let us take some fuzzy (resp. finitely many-valued) data and build up using them *vectors of fuzzy information* generating a hierarchical structure between digits of these vectors. If $x = \langle x_1, x_2, \dots, x_n, \dots \rangle$, where $x_j \in [0,1]$ in the hypervalued case (resp. $x_j \in \{0, 1, \dots, p-1\}$ in the p-adic case), is a fuzzy information vector, then digits x_j have different weights. The digit x_0 is the most important, x_1 dominates over x_2, x_2 over x_3 in turn and so on. The distance between uncertain states $x = \langle x_1, x_2, \dots, x_n, \dots \rangle$ and $y = \langle y_1, y_2, \dots, y_n, \dots \rangle$ is determined just by the length of their common root: close uncertain states have a long common root (i.e. then there exists a large integer j such that $x_i = y_i$ for any $i = 0, \dots, j$).

This distance between uncertain states could be regarded within ultrametric space $\langle X, \rho \rangle$, where the distance ρ satisfies the strong triangle inequality: $\rho(x,y) \le \max(\rho(x,z), \rho(z,y))$. For $r \in R_+, a \in X$, we set

$$U_r(a) := \{x \in X \colon \rho(x,a) \le r\}, \;\; U_r^-(a) := \{x \in X \colon \rho(x,a) < r\}.$$

Both sets are called *balls* of radius r with center a. It can be readily shown that balls in ultrametric space have the following properties:

- For any balls U and V in X, either they are ordered by inclusion (i.e. $U \subset V$ or $V \subset U$) or they are disjoint.
- Each point of a ball is a center.

Let us try to present results of a symptom recognition in the form of 2-adic valued logic. Suppose that our system fixes the observations of people to define diagnosis "asthma" and the first task is to detect "cough". Then $x_0 = 1$ if a person who is observed has cough and $x_0 = 0$ if does not. Notice that coughing may be caused by a respiratory tract infection, choking, smoking, air pollution, asthma, gastroesophageal reflux disease, post-nasal drip, chronic bronchitis, heart failure and medications such as ACE inhibitors, etc. Further, we have $x_1 = 1$ if a person has respiratory failure and $x = 0$ otherwise. Then $x_2 = 1$ if a person has chest tightness and $x_2 = 0$ otherwise. After that we set $x_3 = 1$ if (s)he has breathing difficulty and $x_3 = 0$ otherwise, etc. As a result, we will obtain a 2-adic tree if we continue. Results of such symptom recognition can be regarded as the following balls in 2-adic ultrametric space:

- the symptom of *cough* (tussis) $U_{1/2} = \{x = \langle x_0, x_1, \dots, x_n, \dots \rangle \colon x_0 = 1\}$,
- the symptom of *respiratory failure* $U_{1/4} = \{x = \langle x_0, x_1, \dots, x_n, \dots \rangle \colon x_0 = 1, x_1 = 1\}$,
- the symptom of *chest tightness* $U_{1/8} = \{x = \langle x_0, x_1, \dots, x_n, \dots \rangle \colon x_0 = 1, x_1 = 1, x_2 = 1\}$,

- the symptom of *breathing difficulty* $U_{1/16} = \{x = \langle x_0, x_1, \dots, x_n, \dots\rangle$: $x_0 = 1, x_1 = 1, x_2 = 1, x_3 = 1\}$,
- ...

For processing these results we can use operations of 2-adic valued logic $BL\forall_\infty$.

Thus, while the key notion of conventional fuzzy logics is that truth values are indicated by a value on the range $[0, 1]$, with 0.0 representing absolute falseness and 1.0 representing absolute truth, in non-Archimedean and p-adic fuzzy logics we come cross non-Archimedean or p-adic trees (Fig. 1 and 2) and truth values are indicated by a value on the range ${}^*[0, 1]$ of hypernumbers or on the range $\mathbf{Z}_p$ of p-adic integers. We build up infinite tuples of the form $\langle \mu_1, \mu_2, \dots, \mu_n, \dots\rangle$, where μ_j is a fuzzy measure such that μ_j depends on μ_i for any natural number $i < j$. For instance, we can interpret this dependence as types of higher-order logic. Let us take a fuzzy set $\mathcal{P}_i \in \mathcal{F}_i(D)$, where $\mathcal{F}_i(D) := \underbrace{[0,1]^{\cdots^{[0,1]^{[0,1]^D}}}}_{i+1}$, then a membership function $\mu_{\mathcal{P}_i}(x)$ is defined as the degree of membership of x in $\mathcal{P}_i$.

The further advantages of non-Archimedean and p-adic valued fuzzy logics consist in a possibility to be considered as behavior fuzzy logics. Let us recall that behaviors can be viewed as a labelled transition system. The set of finite sequences over a set A will be denoted by A^*, and the empty sequence by ϵ. The transition system is understood as a tuple $\Upsilon = \langle S, S_0, L, \longrightarrow\rangle$, where S is a potentially infinite collection of states, S_0 is the set of initial states, L is a potentially infinite collection of labels, $\longrightarrow \subseteq S \times L \times S$ is a transition relation that models how a state $s \in S$ can evolve into another state $s' \in S$ due to an interaction $\alpha \in L$. Usually, $\langle s, \alpha, s'\rangle \in \longrightarrow$ is denoted by $s \xrightarrow{\alpha} s'$. A state s' is reachable from a state s if $s \xrightarrow{\alpha} s'$.

The *finite trace of transition system* is a pair $\langle s_0, \sigma\rangle$, where $s_0 \in S_0$ and $\sigma = \alpha_1\alpha_2 \dots \alpha_n$ is a finite sequence of labels such that there are states $s_0, s_1, \dots, s_n$ satisfying $s_i \xrightarrow{\alpha_{i+1}} s_{i+1}$ for all i such that $0 \leq i < n$. The *infinite trace of transition system* is a pair $\langle s_0, \sigma\rangle$, where $s_0 \in S_0$ and $\sigma = \alpha_1\alpha_2 \dots$ is an infinite sequence of labels such that there are states $s_0, s_1, \dots$ satisfying $s_i \xrightarrow{\alpha_{i+1}} s_{i+1}$ for all $i \geq 0$. The set of all finite (resp. infinite) traces is denoted by $trace^*(\Upsilon)$ (resp. by $trace^\omega(\Upsilon)$). Notice that non-Archimedean and p-adic trees may be naturally interpreted as sets of traces of transition system, where the set of states runs either over $[0, 1]$ or $\{0, \dots, p-1\}$. In other words, non-Archimedean and p-adic trees may be regarded as behavior trees.

Now let us construct a *fuzzification of transition system* Υ by defining the following functions

- in case S is infinite, $fuz^{*}\colon trace^{*}(\Upsilon) \mapsto [0,1]^{*}$, where $[0,1]^{*}$ denotes the set of finite tuples of members of $[0,1]$.
- in case S is infinite, $fuz^{\omega}\colon trace^{\omega}(\Upsilon) \mapsto {}^{*}[0,1]$.
- in case S is finite and $|S| = |\{0,\dots,p-1\}|$, i.e. they are of the same cardinality, then

$$fuz^{*}\colon trace^{*}(\Upsilon) \mapsto \{0,\dots,p-1\}^{*},$$

  where $\{0,\dots,p-1\}^{*}$ denotes the set of finite tuples of members of $\{0,\dots,p-1\}$.
- in case S is finite and $|S| = |\{0,\dots,p-1\}|$, i.e. they are of the same cardinality, then

$$fuz^{\omega}\colon trace^{\omega}(\Upsilon) \mapsto \mathbf{Z}_p.$$

Suppose that all these functions are injections such that

- $fuz^{*}(\epsilon) = \langle 0,\dots,0\rangle$, where ϵ is the empty sequence of $trace^{*}(\Upsilon)$,
- $fuz^{\omega}(\epsilon) = {}^{*}0$, where ϵ is the empty sequence of $trace^{\omega}(\Upsilon)$,
- an i-th projection of $fuz^{*}(\langle s_0,\sigma\rangle)$ (resp. $fuz^{\omega}(\langle s_0,\sigma\rangle)$) is a fuzzy measure of s_i for any $i = 0,1,2,\dots$

Thus, we can analyze an evolution of transition system Υ by setting its fuzzification and further by using the non-Archimedean valued logic $Ł\Pi\forall_{\infty}$ and the p-adic valued logic $BL\forall_{\infty}$ for processing results.

Let us assume that the following n ($n > 6$) attributes should be identified for a medical diagnosis: palpation of an appropriate part of body P, result of X-ray X, body temperature T, blood pressure P, rate of heartbeats H, rate of breathing B, etc. Each experiment consists of obtaining data for each attribute in the fuzzy diapason: [norm, pathology]. Let us denote by S^n the set of all possible data (signs) that could be obtained in those experiments. By S_0^n we will denote results of the first experiment. Our diagnosis model could be presented as a transition system $\Upsilon = \langle S^n, S_0^n, L^n, \longrightarrow\rangle$, where L^n is a set of labels for each attribute. Suppose that L^n consists of the following items [5]:

- *OC*: a relation between two signs if the first sign shows obligatory occurrence with a second one, and if it occurs it is confirming for that second sign. *Example*: The X-ray finding "endoprothesis of the knee" is obligatory occurring and confirming for the diagnoses "arthroplasty of the knee".

- *FC*: a relation between two signs if the first sign shows facultative occurrence with a second one, but if it occurs it is confirming for that second. *Example*: The lab result "intracellular uric acid crystals in joint effusion" is facultative occurring yet confirming for the diagnosis "gout".
- *ON*: a relation between two signs if the first sign has obligatory occurrence with a second one, but does not confirm it. *Example*: The clinical finding "HEBERDEN's nodes" is obligatory occurring and not confirming for the diagnosis "HEBERDEN's arthrosis".
- *FN*: a relation between two signs if the first sign is both facultative occurring with a second one and not confirming for that sign. *Example*: The lab finding "elevated ESR" isfacultative occurring and not confirming for the diagnosis "rheumatoid arthritis".
- *E*: a relation between two signs if the first sign excludes a second sign. *Example*: The lab finding "WAALER ROSE titre 1:128" excludes the diagnosis "seronegative rheumatoid arthritis".

Thus, in a diagnosis model Υ we could deal with labels defined as fuzzy relations between states s_i and s_{i+1} of an appropriate attribute, where s_i and s_{i+1} are states obtained in the i-th experiment and $i+1$-st experiment respectively. The sequences of such fuzzy labels could be regarded as traces of Υ within the non-Archimedean valued logic $Ł\Pi\forall_\infty$.

Thus, unconventional probabilities and fuzziness may find a lot of applications in medical diagnosis within fuzzy logical approach.

REFERENCES

[1] Matthias Baaz, Agata Ciabattoni, Norbert Preining. SAT in Monadic Gödel Logics: a borderline between decidability and undecidability, *WoLLIC '09 Proceedings of the 16th International Workshop on Logic, Language, Information and Computation.* 2009, pp. 113–123.

[2] Karl Boegla, Klaus-Peter Adlassniga, Yoichi Hayashic, Thomas E. Rothenfluhd, Harald Leitich. Knowledge acquisition in the fuzzy knowledge representation framework of a medical consultation system, *Artif. Intell. Med.* 30(1), 2004, pp. 1–26.

[3] Agata Ciabattoni, Pavel Rusnok. On the Classical Content of Monadic $G^{\sim}$ and its Application to a Fuzzy Medical Expert System, Fangzhen Lin, Ulrike Sattler, Miroslaw Truszczynski (Eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010*, Toronto, Ontario, Canada, May 9–13, 2010. AAAI Press 2010, pp. 435–444.

[4] Agata Ciabattoni & Thomas Vetterlein. On the (fuzzy) logical content of CADIAG-2, *Fuzzy Sets and Systems*, 161(14), 2010, pp. 1941–1958.

[5] Gernot Kolarz & Klaus-Peter Adlassnig. Problems in Establishing the Medical Systems CADIAG-1 and CADIAG-2 in Rheumatology, *J. Med Syst.* 10(4), 1986, pp. 395–405.

[6] David Picado Muiño. A graded inference approach based on infinite-valued Łukasiewicz semantics, *Multiple-Valued Logic (ISMVL), 2010 40th IEEE International Symposium*, 2010, pp. 252–257.

[7] David Picado Muiño. The (probabilistic) logical content of CADIAG2. Rule-Based Probabilistic Approach, *ICAART (1)*, 2010, pp. 28–35.

[8] F. Steimann & K.-P. Adlassnig. Fuzzy Medical Diagnosis, E. Ruspini, P. Bonissone, W. Pedrycz (eds) *Handbook of Fuzzy Computation.* IOP Publishing and Oxford University Press, 1998, pp. 1–6.

[9] Thomas Vetterlein, Klaus-Peter Adlassnig. T-norm-based fuzzy logics and logics for reasoning under vagueness, *IFSA-EUSFLAT* 2009, pp. 1085–1090.

[10] Thomas Vetterlein. Fuzzy logic as a logic of the expressive strength of information, *Soft Computing – A Fusion of Foundations, Methodologies and Applications*, Volume 12, Number 5, pp. 479–485.

Andrew Schumann
Belarusian State University (Minsk)
andrew.schumann@gmail.com

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 22 (35) 2010

Dominik Tomaszuk
Institute of Computer Science
University of Bialystok

DOCUMENT-ORIENTED TRIPLESTORE BASED ON RDF/JSON

Abstract: This paper defines dokument-oriented database as RDF triplestore. It proposes an alternative mean of serializing RDF triples using JavaScript Object Notation (JSON), a lightweight representation format which emphasizes legibility and brevity. This paper proposes declarative SPARQL mapping method to object-oriented imperative query language. This query language uses RDF/JSON syntax. It means that the dokument-oriented database, such as MongoDB, could be a triplestore.

Keywords: Semantic Web, Resource Description Framework (RDF), knowledge representation, triplestores, document-orientet databases, object-oriented programming (OOP).

1. Introduction

A simplified view of the Semantic Web is a collection of web retrievable RDF documents, each containing a Resource Description Framework (RDF) graphs. RDF has come to be used as a general method for conceptual description or modeling of information that is implemented in web resources. In other words, RDF is a general-purpose language for representing information on the Web. RDF is designed to model and store information. It is also developed to provide interoperability between applications that exchange machine-understandable information on the Web.

RDF Recommendations [1, 2, 3] explain the meaning of subject, predicate and object. These expressions are known as triples in RDF terminology. The subject denotes the resource. The predicate means traits or aspects of the resource and expresses a relationship between the subject and the object. A collection of RDF statements intrinsically represents the labeled, directed multi-graph (figure 1). The nodes of an RDF graph are its subjects and objects.

More formally, let U to be the set of all URI references, B an infinite set of blank nodes, L the set RDF plain literals, and D the set of all RDF

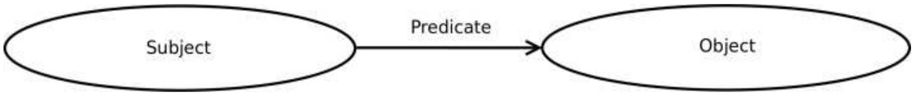


Fig. 1. RDF graph data model

typed literals. All the four sets are defined in [1]. U, B and L are pairwise disjoint. Let $O = U \cup B \cup L \cup D$ and $S = U \cup B$, then $T \subset S \times U \times O$ is set of all RDF triples[1].

A data model is an abstract model that describes how data is represented and accessed. Data models formally define data elements and relationships among data elements for a domain of interest. A data model explicitly determines the structure of data or structured data. Typical applications of data models include database models. A database model or database schema is the structure or format of a database, described in a formal language supported by the database management system. In other words, a database model is the application of a data model when used in conjunction with a database management system.

This paper studies the RDF model from a database perspective. From this point of view it is compared with other database models, in particular with graph database models, which are very close to RDF approach. A graph database uses nodes, edges and properties to represent and store information. This model is an alternative to relational databases or other structured storage systems based on columns.

In this paper a new document-oriented triplestore based on RDF/JSON is presented. Section 2 provides a related work on triplestores. In the next section document-oriented, semi-structured, JSON and key-value databases are presented. In section 4 three mapping SPARQL to Mongo Query Language algorithms are proposed. In the next section results of the experiments are presented. Conclusions end the paper.

2. Related work

A triplestore is a purpose-built database for the storage and retrieval of RDF data. A triplestore is optimized for the storage and retrieval of many triples. Much like a relational database, one stores information in a triple-

[1] The RDF core Working Group noted that it is aware of no reason why literals should not be subjects and a future WG with a less restrictive charter may extend the syntaxes to allow literals as the subjects of statements.

store and retrieves it via a query language. A difficulty with implementing triplestores over SQL is that triples may be stored and implemented efficiently by querying of a graph-based RDF model. One of the query languages for the RDF model is SPARQL [4]. Mapping from SPARQL onto SQL queries is difficult. There are a lot of triplestores that use relational database to store RDF triples.

Jena [5] is a Semantic Web framework for Java. It provides an API to extract data from and write to RDF graphs. The graphs are represented as an abstract model. A model can be sourced with data from files, databases, URLs or a combination of these. A Model can also be queried through SPARQL. Jena supports serialisation of RDF graphs to a relational database, RDF/XML [6], Turtle [7] and Notation3 [8]. Joseki is a part of Jena. It is an RDF server. The goal of Joseki is to provide an HTTP interface to RDF data. It fully supports SPARQL for querying. Joseki can also be run a stand-alone server.

Sesame [9] is a framework for querying and analyzing RDF data. It contains a triplestore. One of the primary components of Sesame is the RDF query language SeRQL. Sesane can be sourced with data from memory, native store that stores and retrieves its data directly to/from the disk and relational database.

Another framework is RDF API for PHP (RAP) [10]. RAP includes in-memory and database model storage. The MemModel implementation stores statements in an array in the system memory. The DbModel implementation stores statements in an relational database. RAP implements the SPARQL query.

4Store [11] is an example of a triplestore that is a storage and query engine. It holds RDF data. It does not provide many features over and above RDF storage and SPARQL queries.

Mulgara [12] is another triplestore. It is massively scalable, and transaction-safe. Mulgara instances can be queried via the iTQL query language and the SPARQL query language. Mulgara is not based on a relational database due to the large numbers of table joins encountered by relational systems when dealing with metadata. Instead, Mulgara is a completely new database optimized for metadata management. Mulgara models hold metadata in the RDF triples. Metadata may be imported into or exported from Mulgara in RDF or Notation 3 form [8].

JRDF [13] is an open source RDF framework for Java. It provides an object oriented model of RDF graphs. JRDF API allow RDF to be created and written using various formats such as RDF/XML [6] and Notation3 [8]. It has an implementation of SPARQL [4]. By default JRDF uses in memory store. It also can use a disk based store through various persistence layers. JRDF also implement a triple store across Hadoop using a technique called RDF molecules [14].

AllegroGraph [15] is another example of a triplestore. In contrast with a relational database, a AllegroGraph considers each stored item to have any number of relationships. It is designed to store RDF tuples. It implements the SPARQL query. AllegroGraph does not use relational database. It loads triples through RDF/XML [6], N-Quads[2] and N-Triples [16].

3. Document-oriented databases

As opposed to relational databases, document-based databases do not store data in tables with uniform sized fields for each record. Instead, each record is stored as a document that has certain characteristics. Any number of fields of any length can be added to a document. Fields can also contain multiple pieces of data.

Unlike a relational database where each record would have the same set of fields and unused fields might be kept empty, in document-oriented database there are no empty fields in either record. Document-oriented database allows information to be added any time without wasting storage space for empty fields as in relational databases. For these cases, JavaScript Object Notation (JSON) is often used.

JSON is a lightweight data-interchange format. It is effortless for humans to read and write. It is easy for machines to parse and generate. JSON is based on a subset of the [17]. JSON is a text format that is completely independent from programming languages. These properties make JSON a perfect data-interchange language.

JSON is built on two structures: a collection of name/value pairs and an ordered list of values. In most languages, collection of name-value pairs is realized as an object, struct, record, hash table, associative array, dictionary or keyed list. In various languages, ordered list of values is realized as an array, list, vector or sequence. JSON's basic types are numbers, strings, booleans, arrays, object and null. Number in JSON is integer, real, or floating point[3]. The string is double-quoted Unicode with backslash escaping. The boolean contains true or false value. The array is an ordered sequence

[2] N-Triples with context.

[3] The octal and hexadecimal formats are not used.

of values, comma-separated and enclosed in square brackets. The object is collection of key:value pairs, comma-separated and enclosed in curly brackets. Null contains empty value [18]. Main advantage of JSON is that it translates directly into universal data structures.

In JSON each record can have a non-standard amount of information. Such information is properly called semi-structured data. Semi-structured data is a form of data model that does not conform with the formal structure of tables and data models associated with databases but contains nonetheless tags or other markers to separate semantic elements and hierarchies of records and fields within the data. In this model, there is no separation between the data and the schema, and the amount of structure used depends on the purpose. The advantage of this model is that it provides a flexible format for data exchange between different types of systems and the data transfer format may be portable [19].

Most of document-oriented databases are also key-value databases. The key-value database characterized by the fact that each key is associated with one value or set of values. The relationship between a key and its value is sometimes called a mapping or binding. It is closely related to the mathematical concept of a function with a finite domain. JSON helps store key-value pairs in a formatted way. A document can have any number of key-value pairs. Instead of using a schema, documents of the same time all have a similar set of key-value pairs.

4. MongoDB as a triplestore

Document-oriented key-value stores such as MongoDB [20] are different than triple stores. The key-value store consists of two terms: keys and values, the triple stores consists of three terms: subjects, predicates and objects. It is difficult to map a key-value pairs to RDF triples.

It is a proposed method for mapping an RDF graph to the structure of JSON. Such syntax, being the equivalent to RDF model, would be more legible and brief. It describes JSON structure for RDF graph that expresses the whole RDF model that does not lose information.

4.1. MongoDB overview

MongoDB [20] is a document-oriented database written in the C++ programming language. The goal of MongoDB is to bridge the gap between key-value stores and traditional relational databases. MongoDB manages collections of JSON-like documents. This allows many applications to model data in a more natural way, as data can be nested in complex hierarchies and still query-able.

In MongoDB all strings are UTF-8. Non-UTF-8 data can be saved, queried, and retrieved with a special binary data type – BSON [21]. This is nominally a superset of JSON types. BSON types include string, integer, double, date, byte array[4], boolean, null and BSON object. It is a binary format in which zero or more key/value pairs are stored as a single document.

BSON documents consist of a well ordered list of elements. Each element consists of a field name, a type and a value. Field names are strings. Compared to JSON, BSON is efficient both in storage space and scan-speed. Large elements in a BSON document are prefixed with a length field to facilitate scanning.

MongoDB has object query language. Main methods are:

- find([query], [fields]) – returns a result set of records. It can also contain statement should only affect records that meet specified criteria,
- limit() and skip() – restrict the range of results,
- distinct() – retrieves only unique data entries,
- count() and size() – returns the number of records with and without skip() and limit(),
- insert() and update() – add and modify data,
- explain() and stats() – show various statistics.

Listing 2 presents an example of MongoDB methods.

```
db.mydb.find({name:"Jan Kowalski"}).limit(3).forEach(printjson);
```

Listing 1. Simple MongoDB query in JavaScript

MongoDB supports cursors that comprises a control structure for the successive traversal of records in a result set. Main cursor methods are: forEach(func), print(), map(), hasNext() and next().

Listing 2 presents an example of MongoDB cursor.

```
var cursor = db.mydb.find();
while (cursor.hasNext()) printjson(cursor.next());
```

Listing 2. Simple MongoDB cursor with iterator (JavaScript)

[4] Binary data

4.2. Data structure schema

It is proposed to use a modified version of the RDF/JSON serialization [22] to store triples in the MongoDB. Three keys symbolize subject, predicate and object. All three keys should be JSON string. The key named ‘subject’ should contain RDF subject that should be blank node or URI reference type[5]. The key named ‘predicate’ should contain RDF predicate that should be URI reference type. The key named ‘object’ should contain RDF object that should be blank node, URI reference, plain literal or typed literal type. Table 1 presents types of RDF terms.

Table 1

Types of RDF terms in MongoDB

Types	Subject	Predicate	Object	Example
URI	allowed	allowed	allowed	`<http://example.org/path/>`
Blank node	allowed	forbidden	allowed	`_:me`
Plain literal	forbidden but possible	forbidden	allowed	`"chat"@en`
Typed literal	forbidden but possible	forbidden	allowed	`"f"^^<http://example.org/char>`

When type of value is a plain literal it optionally can be defined with language suffix. Languages are determined by appending the simple literal with ‘at’ sign and the language tag, that is defined in [23] and if supplied, it must not be empty. When the type of value is typed literal it should contain URI similarly append double caret signs followed by any legal URI full form. Literals are written inside double quotation marks.

When the type of value is URI, value should be full URI, not just short QName [24]. The URIs are in enclosed in inequality signs. Blank nodes are represented by the underscore sign, colon sign and node identifier.

Listing 3 presents a schema of a single object (record). Listing 4 presents a simple RDF statement based on JSON Schema [25] and used in MongoDB.

4.3. Mapping SPARQL to Mongo Query Language

It is proposed to introduce a triplestore based on document-oriented MongoDB. RDF has different query languages. The official and most widely used is the SPARQL [4][6].

[5] There is no problem to literals will be supported in the future. SPARQL allows such solution.

[6] It is a recursive acronym that stands for SPARQL Protocol and RDF Query Language.

```
{"description":"RDF/JSON Schema for MongoDB",
 "type":"object",
 "properties":{
 "subject":{
 "type":"string",
 "description":"Subject, possible types of values: bnode and uri
               (in future literal and typed-literal)",
},
"predicate":{
 "type":"string",
 "description":"Predicate, possible types of values: uri",
},
"object":{
 "type":"string",
 "description":"Object, possible types of values: bnode, uri, literal
               and typed-literal",
}
}
}
```

Listing 3. Schema of a single object in JSON Schema

```
{subject: "_:a",
 predicate: "<http://xmlns.com/foaf/0.1/name>",
 object: "\"Dominik Tomaszuk\""}
```

Listing 4. RDF/JSON in MongoDB

SPARQL is adapted to the specific structure of RDF graphs and is based on triples that constitute them. SPARQL can be used to express queries across diverse data sources, whether the data is stored natively as RDF or viewed as RDF via middleware. SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions. It also supports extensible value testing and constraining queries by source RDF graph. The results of SPARQL queries can be results sets or RDF graphs.

SPARQL has a similar structure as SQL [26]. There are a lot of keywords that are similar to SQL, such as: SELECT, DISTINCT, WHERE, ORDER BY etc. The SELECT form of results returns all or a subset of, the variables bound in a query pattern match. The WHERE clause provides the basic graph pattern to match against the data graph. The WHERE clause may have a FILTER keyword. In FILTERs restrict solutions to those for which the filter expression evaluates to true. The ORDER BY clause establishes the order of a solution sequence. Following the ORDER BY clause is a sequence of order comparators, composed of an expression and an optional order modifier either ASC() or DESC(). The OFFSET causes the solutions generated to start after the specified number of solutions. The LIMIT clause puts an upper bound on the number of solutions returned.

To make queries concise, SPARQL allows the definition of prefixes and base URIs in a fashion similar to Terse RDF Triple Language (Turtle) [7].

SPARQL support variables. A query variable is a member of the set V where V is infinite and disjoint from O (RDF term, see section 1). Variables are indicated by a "?" or "$" prefix. Most forms of SPARQL query contain a set of triple patterns called a basic graph pattern. Triple patterns are like RDF triples except that each of the subject, predicate and object may be a variable. More formally, a triple pattern is member of the set: $(O \cup V) \times (U \cup V) \times (O \cup V)$. A basic graph pattern matches a subgraph of the RDF data when RDF terms from that subgraph may be substituted for the variables and the result is RDF graph equivalent to the subgraph. Triple Patterns are written as a whitespace-separated list of a subject, predicate and object. Triple patterns with a common subject can be written so that the subject is only written once and is used for more than one triple pattern by employing the semicolon notation. If triple patterns share both subject and predicate, the objects may be separated by comma sign. Listing 5 presents a simple SPARQL query.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE
{?x foaf:name ?name .
 ?x foaf:mbox ?mbox . }
```

Listing 5. Simple SPARQL query

MongoDB uses own query syntax and do not support SPARQL. Mongo Query Language is an object-oriented imperative language. SPARQL is a domain-specific declarative language. Therefore, it is difficult to replace the SPARQL to MongoDB Query Language.

It is proposed an algorithm that maps SPARQL to Mongo Query Language. This proposal allows MongoDB change in the tripletore and SPARQL endpoint. The proposals return results in Turtle. Therefore, results can be processed again. The first algorithm matching:

- single pattern,
- multiple patterns,
- literals with language tags,
- literals with arbitrary datatypes,
- optional patterns.

The algorithm supports restrictions the values of strings with regex function with and without flags. Regex function is equivalent to XPath [31] fn:matches and supports regular expressions [30]. The algorithm supports also restricting numeric values using >, <, <= and >= operators, that are equivalent to XPath op:numeric-greater-than and op:numeric-less-than with or without logical disjunction. The second algorithm supports distinct matching. The third algorithm matching alternatives. Algorithms 1, 2 and 3 present an idea of mapping SPARQL to an MongoDB object-oriented language.

```
Input: RDF/JSON
Output: Turtle
TP←triple patterns
LTP←singly-linked list of TP
V←query variables
RT←RDF term (literal ∪ typed-literal ∪ URI ∪ black node), RU←RDF URI,
RS←RDF URI ∪ black node
T←RDF triples, T = RT × RU × RO
TQV←three elements array of singly-linked list of query variables or RDF terms
TQV[0]←singly-linked list of V ∪ RS, TQV[1]←V ∪ RU, TQV[2]←V ∪ RT
JO←JSON object where key ∈ "subject" ∪ "predicate" ∪ "object"
C←cursor
LC←singly-linked list of cursors
 parse SELECT clause, WHERE clause
 add V || RT to TQV, TP to LTP
 Foreach LTP do
  create JO, C
  add C to LC
  Foreach TQL do
   If V.previous == V then
    add to JO in C condition with value of previous RT in find method
   end
   If TQL value != V then
    add to JO in C condition in find method
   end
  end
  If FILTER ∈ WHERE than
   If regex function ∈ FILTER then
    add to JO in C condition with regular expression in find method
   Else
    add to JO in C condition with numeric values operator in find method
   end
  end
  If OPTIONAL ∈ WHERE then
   mark C as optional
  end
  If ORDER BY clause ∉ ∅ then
   parse ORDER BY
   create JO
   If DESC() ∈ ORDER BY then
    add to JO with value 1 to sort method
   Else
    add to JO with value -1 to sort method
   end
   add to C
   move C to top of LC
  end
  If LIMIT clause ∉ ∅ then
   parse LIMIT
   add to C limit method
  end
  If OFFSET clause ∉ ∅ then
   parse OFFSET
   add to C skip method
  end
 end
 Foreach LC do
  create loop
  print results in loop
  If optional == true then
   print results in previous loop that is not optional
  end
 end
```

Algorithm 1. Mapping SPARQL to Mongo Query Language main algorithm

```
Input: RDF/JSON
Output: Turtle
TP←triple patterns
LTP←singly-linked list of TP
V←query variables
RT←RDF term (literal ∪ typed-literal ∪ URI ∪ black node),
RU←RDF URI, RS←RDF URI ∪ black node
T←RDF triples, T = RT × RU × RO
TQV←three elements array of singly-linked list of query variables or RDF terms
TQV[0]←singly-linked list of V ∪ RS, TQV[1]←V ∪ RU,
TQV[2]←V ∪ RT
JO←JSON object where key ∈ "subject" ∪ "predicate" ∪ "object"
C←cursor
LC←singly-linked list of cursors
 parse SELECT clause, WHERE clause
 add V || RT to TQV, TP to LTP
 Foreach LTP do
  create JO
  create C
  add C to TDC
  Foreach TQL do
   If V.previous == V then
    add to JO in C condition with value of previous RT in distinct method
   end
   If TQL value != V then
    add to JO in C condition in distinct method
   end
  end
  If FILTER ∈ WHERE then
   If regex function ∈ FILTER then
    add to JO in C condition with regular expression in distinct method
   Else
    add to JO in C condition with numeric values operator in distinct method
   end
  end
  add to C limit method with 1 in parameter
 end
 create C condition with value of RT in find method
 print results
```

Algorithm 2. Mapping SPARQL DISTINCT to Mongo Query Language algorithm

```
Input: RDF/JSON
Output: Turtle
TP←triple patterns
LTP←singly-linked list of TP
V←query variables
RT←RDF term (literal ∪ typed-literal ∪ URI ∪ black node), RU←RDF URI,
RS←RDF URI ∪ black node
T←RDF triples, T = RT × RU × RO
TQV←three elements array of singly-linked list of query variables or RDF terms
TQV[0]←singly-linked list of V ∪ RS, TQV[1]←V ∪ RU, TQV[2]←V ∪ RT
JO←JSON object where key ∈ "subject" ∪ "predicate" ∪ "object"
C←cursor
 parse SELECT clause, WHERE clause
 create JO, C
 parse left union group, right union group
 add to JO in C condition with values of previous RT in find method using $or key
 If ORDER BY clause ∉ ∅ then
  parse ORDER BY
  create JO
  If DESC() ∈ O then
   add to JO with value 1 to sort method
  Else
   add to JO with value -1 to sort method
  end
  add to C
  move C to top of LC
 end
 If LIMIT clause ∉ ∅ then
  parse LIMIT
  add to C limit method
 end
 If OFFSET clause ∉ ∅ then
  parse OFFSET
  add to C skip method
 end
 Foreach C do
  print results
 end
```

Algorithm 3. Mapping SPARQL UNION to Mongo Query Language algorithm

5. Experimental results

Speed criterion, which is adopted to measure the document-oriented triplestore, is time of load and select triples. Loading is the process of adding one or more records, objects or statements to the database. In SQL and SPARQL 1.1 it is INSERT clause. In addition, triples could be also added through various APIs. In Mongo Query Language loading uses insert() or save() methods. Selection is the process of return of a result set of records, objects or statements from the database. These results may include order and specified criteria. In SQL and SPARQL it is SELECT clause. In Mongo Query Language selection uses find() or distinct() methods.

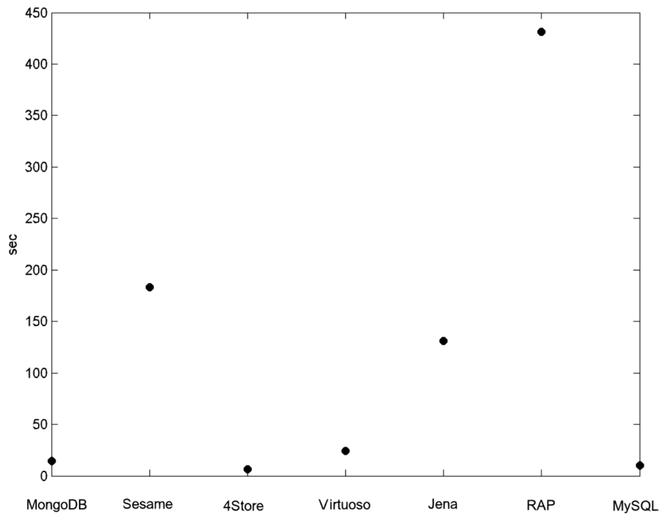


Fig. 2. Times of loading triples

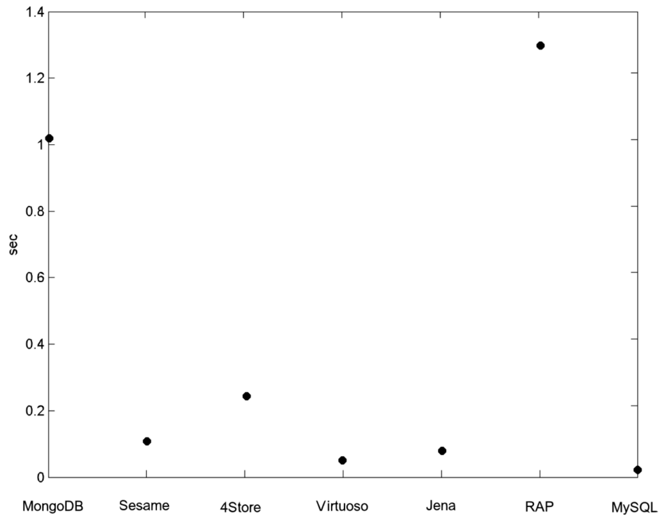


Fig. 3. Times of selecting triples

Main tests are executed on typical desktop computer with two Intel Core 2 Quad 2666MHz CPUs and 4GB RAM. Testing computer running with Ubuntu 9.10 in Ext3 filesystem. Tests are based on the Berlin SPARQL Benchmark [27]. Tests are performed on 1 000 313 triples. Selection is based on the Query 6. Tests concern MongoDB [20] compared to triplestore such as Sesame [9], 4Store [11], Virtuoso [28], Jena SDB with MySQL [5], RAP [10] and MySQL with InnoDB engine [29].

Figure 2 presents times of loading triples. The chart shows that the best times have 4Store, MySQL and MongoDB. It is important that MySQL is a relational database, which is not triplestore and data in this database are not triples, so the best times belongs to 4Storne and MongoDB with triples in RDF/JSON.

Figure 3 shows times of matching multiple patterns. The chart shows that the worst times have RAP and MongoDB. Mongo Query Language do not select multiple patterns well but match simple patterns, distinct patterns (algorithm 2) and alternatives (algorithm 3); it has time close to others triplestores and relational database.

6. Conclusions

The problem of how to store RDF triples has produced many proposals and models. Some of them are relational, object-oriented or use a graph. I have produced a thought-out and simple proposal. I believe my document-oriented triplestore based on RDF/JSON is an interesting approach.

MongoDB is a document-oriented database that uses JSON. I propose algorithms that transform SPARQL queries to Mongo Query Language. This allows MongoDB to be a triplestore. I also present RDF/JSON that can be used in this triplestore.

The experiments show that MongoDB and document-oriented key-value with semi-structured JSON database could be a good triplestore. The advantage of MongoDB as triplestore is that it is very scalable. It means that MongoDB provides more efficient work with increasing number of objects in the database.

I realized that some further work on this issue is still necessary to extend proposed algorithms to support CONSTRUCT, ASK, DESCRIBE queries and source graphs.

Acknowledgements

The author would like to thank Ivan Herman from the World Wide Web Consortium. Professor Henryk Rybinski's comments and support were invaluable.

REFERENCES

[1] G. Klyne and J. J. Carroll. Resource Description Framework (RDF): Concepts and Abstract Syntax. World Wide Web Consortium, 2004.

[2] P. Hayes. RDF Semantics. World Wide Web Consortium, 2004.

[3] F. Manola E. and Miller. RDF Primer. World Wide Web Consortium, 2004.

[4] E. Prud'hommeaux and A Seaborne. SPARQL Query Language for RDF. World Wide Web Consortium, 2008.

[5] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne and K. Wilkinson. Jena: implementing the semantic web recommendations. International World Wide Web Conference, Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, 2004.

[6] D. Beckett, RDF/XML Syntax Specification (Revised). World Wide Web Consortium, 2004.

[7] D. Beckett, T. Berners-Lee, Turtle – Terse RDF Triple Language. World Wide Web Consortium, 2008.

[8] T. Berners-Lee, Notation3. World Wide Web Consortium, 2006.

[9] J. Broekstra, A. Kampman and F. van Harmelen. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema, The Semantic Web – ISWC 2002, Springer, 2002.

[10] R. Oldakowski, C. Bizer, D. Westphal. RAP: RDF API for PHP. In Proceedings International Workshop on Interpreted Languages, MIT Press, 2004.

[11] S. Harris, N. Lamb and N. Shadbolt. 4store: The Design and Implementation of a Clustered RDF Store, The 5th International Workshop on Scalable Semantic Web Knowledge Base Systems, 2009.

[12] A. Muys. Building an EnterpriseScale Database for RDF Data.

[13] A Fnewman. Getting Started with JRDF. 2010.

[14] A. Newman, J. Hunter, Y. Li, C. Bouton and M Davis. A Scale-Out RDF Molecule Store for Distributed Processing of Biomedical Data. Semantic Web for Health Care and Life Sciences Workshop, 2008.

[15] J. Aasman. Allegro Graph: RDF Triple Database. Technical Report 1, Franz Incorporated, 2006.

[16] J. Grant and D. Beckett. RDF Test Cases. World Wide Web Consortium, 2004.

[17] M. Cowlishaw. ECMAScript language specification. International Organization for Standardization, 1998.

[18] D. Crockford. The application/json Media Type for JavaScript Object Notation (JSON). Internet Engineering Task Force, 2006.

[19] S. Abiteboul. Querying semi-structured data. Database Theory-ICDT'97, Springer, 1997.

[20] K. Chodorow. MongoDB manual. 10gen. 2009.

[21] M. Dirolf. BSON specyfication. 2009.

[22] D. Tomaszuk. Serializing RDF graphs in JSON. III Krajowa Konferencja Naukowa Technologie Przetwarzania Danych, 2010.

[23] H. Alvestrand, Tags for the Identification of Languages. Internet Engineering Task Force, 2001.

[24] T. Bray, D. Hollander, A. Layman and R. Tobin. Namespaces in XML (Second Edition). World Wide Web Consortium, 2006.

[25] K. Zyp. A JSON Media Type for Describing the Structure and Meaning of JSON Documents. Internet Engineering Task Force, 2010.

[26] D. D. Chamberlin and R. F. Boyce. ISO/IEC 9075:1992 – Database Language SQL. International Organization for Standardization, 1992.

[27] C. Bizer and A. Schultz. Berlin SPARQL Benchmark. International Journal On Semantic Web and Information Systems, 2009.

[28] O. Erling and I. Mikhailov. RDF Support in the Virtuoso DBMS. Networked Knowledge – Networked Media. Springer, 2009.

[29] M. Widenius and D. Axmark. MySQL reference manual: documentation from the source. O'Reilly Media, 2002.

[30] M. Davis and A. Heninger. Unicode Regular Expressions, Unicode Consortium, 2008.

[31] A. Malhotra, J. Melton and N. Walsh. XQuery 1.0 and XPath 2.0 Functions and Operators, World Wide Web Consortium, 2007.

Dominik Tomaszuk
Institute of Computer Science
University of Bialystok
dtomaszuk@ii.uwb.edu.pl

Tomasz Kaczmarek
Poznan University of Economics, Faculty of Informatics and Electronic Economy, Department of Information Systems

Dominik Zyskowski
Poznan University of Economics, Faculty of Informatics and Electronic Economy, Department of Information Systems

Adam Walczak
Poznan University of Economics, Faculty of Informatics and Electronic Economy, Department of Information Systems

Witold Abramowicz
Poznan University of Economics, Faculty of Informatics and Electronic Economy, Department of Information Systems

INFORMATION EXTRACTION FROM WEB PAGES FOR THE NEEDS OF EXPERT FINDING

Abstract: This paper describes a mechanism for the extraction of relevant information about people from Polish portals for professionals. The method of information extraction is based on hierarchical execution of XPath commands and regular expressions depending on the structure of processed documents. The extraction component EXT is a part of the eXtraSpec system, which task is to support Human Resources departments of Polish companies during recruitment and team building. EXT is able to deal with several sources of information and with user profiles that are acquired from professionals' portals. In this article we also discuss the advantages of the chosen extraction method in the context of the goals of the whole eXtraSpec system and we show the directions of future research.

Key words: web information extraction, eXtraSpec, hierarchical algorithm, XPath, Polish language

1. Introduction

Several information sources may be useful when searching for candidates in the process of staff recruitment. These sources can vary, from peers' opinions expressed during informal conversation to formalized CV documents prepared by candidates. In-between there is a whole universe of data that one can benefit from in the recruitment process. It is worth noting that information relevant to the expert finding may have traditional form, but also an electronic one. Moreover, not only textual sources can be useful, but also information that appears as the result of social networking between professionals from the particular business domain.

Currently, more and more information is available in an electronic form. The requirement for most recruitment processes is to provide a resume in a form of a computer file, which is later processed by internal workflow systems. One should also consider the development of social activities on the Web that result in the creation of professional profiles on dedicated portals like LinkedIn. It is easily imaginable that the amount of relevant information is enormous. This information may be utilized in the recruitment process, through combination of data from various sources, its cross-validation and complementing. Thanks to using standardized vocabulary it is possible to match a profile built in such a way against concrete requirements defined for a precisely described job offer.

The structure of this paper is as follows. First, we present the general picture of the eXtraSpec system along with a motivating example that justifies a practical usability of such a system. Then, we show the state of the art solutions in this field and comment on the related work already done. In the fourth section we outline our method of web information extraction. We describe the hierarchical extraction algorithm, then present the structure of profile extracted and the structure of extraction rules. Finally, we conclude our work and show the directions of future research.

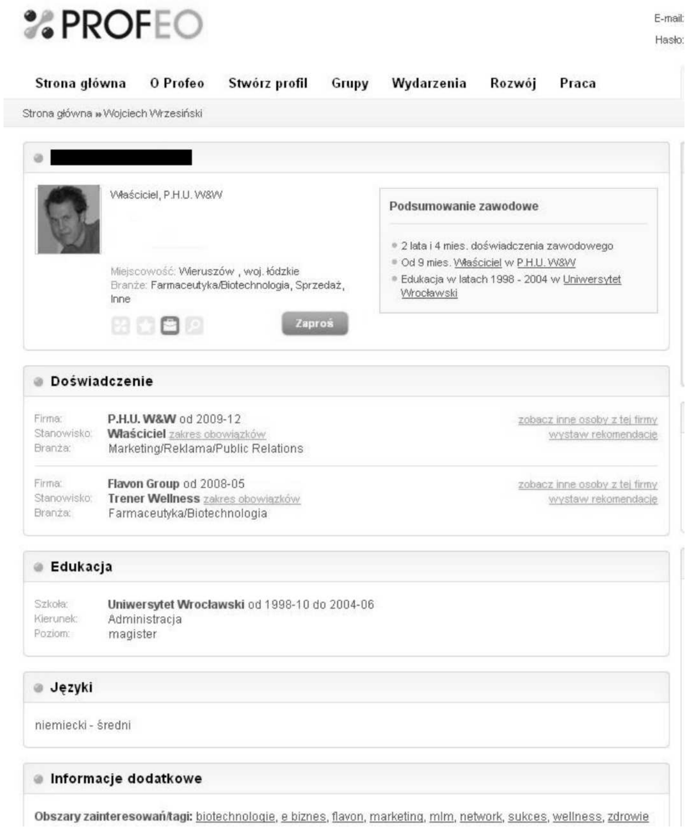
2. Overview of eXtraSpec project

The goal of the eXtraSpec[1] project is to create tools that can help automate the recruitment process and expert finding. Our special interest is the analysis and extraction of information from Web sources, especially available in the Polish language.

The architecture of the eXtraSpec system consists of several components that play their roles in subsequent stages of the supported process [11]. As an input for the system serve documents retrieved from Polish portals for

[1] The work published in this article was supported by the project titled: “Advanced data extraction methods for the needs of expert search” (http://extraspec.kie.ue.poznan.pl) financed under the Operational Program Innovative Economy and partially supported by European Union’s European Regional Development Fund programme (contract no. UDA-POIG.01.03.01-30-150/08-01).

professionals, like Profeo[2] or GoldenLine[3] (see Figure 1 and Figure 2). An important fact is that the system is dedicated to process documents written in the Polish language, which is a hard task due to the complexity of the Polish grammar. To our best knowledge, such research in the domain of the Polish language is quite a novelty. These documents are being processed by the extraction component (EXT), which role is to validate, preprocess them and later to create extracted profiles that store all relevant information from the point of view of the recruiter.



Strona główna » Wojciech Wrzesiński

[redacted]

Właściciel, P.H.U. W&W

Miejscowość: Wieruszów , woj. łódzkie
Branże: Farmaceutyka/Biotechnologia, Sprzedaż, Inne

Podsumowanie zawodowe

- 2 lata i 4 mies. doświadczenia zawodowego
- Od 9 mies. Właściciel w P.H.U. W&W
- Edukacja w latach 1998 - 2004 w Uniwersytet Wrocławski

Doświadczenie

Firma:	**P.H.U. W&W** od 2009-12	zobacz inne osoby z tej firmy
Stanowisko:	**Właściciel** zakres obowiązków	wystaw rekomendację
Branża:	Marketing/Reklama/Public Relations	
Firma:	**Flavon Group** od 2008-05	zobacz inne osoby z tej firmy
Stanowisko:	**Trener Wellness** zakres obowiązków	wystaw rekomendację
Branża:	Farmaceutyka/Biotechnologia	

Edukacja

Szkoła:	**Uniwersytet Wrocławski** od 1998-10 do 2004-06
Kierunek:	Administracja
Poziom:	magister

Języki

niemiecki - średni

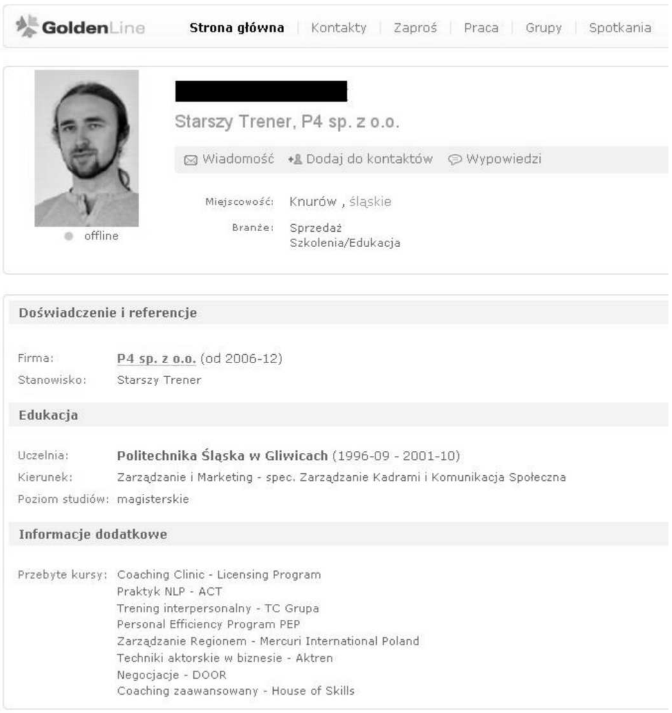
Informacje dodatkowe

Obszary zainteresowań/tagi: biotechnologie, e biznes, flavon, marketing, mlm, network, sukces, wellness, zdrowie

Figure 1. Profeo.pl – exemplary profile

The next step is to normalize profiles using internal domain ontologies in order to perform disambiguation and allow for reasoning over experts' profiles. Subsequent component – Fusion (FUS) – is responsible for detecting

[2] http://profeo.pl/
[3] http://www.goldenline.pl/


GoldenLine
Strona główna | Kontakty | Zaproś | Praca | Grupy | Spotkania
offline
Wiadomość
Dodaj do kontaktów
Wypowiedzi

[redacted]

Starszy Trener, P4 sp. z o.o.

Miejscowość:	Knurów , śląskie
Branże:	Sprzedaż Szkolenia/Edukacja

Doświadczenie i referencje

Firma:	P4 sp. z o.o. (od 2006-12)
Stanowisko:	Starszy Trener

Edukacja

Uczelnia:	**Politechnika Śląska w Gliwicach** (1996-09 - 2001-10)
Kierunek:	Zarządzanie i Marketing - spec. Zarządzanie Kadrami i Komunikacja Społeczna
Poziom studiów:	magisterskie

Informacje dodatkowe

Przebyte kursy:	Coaching Clinic - Licensing Program Praktyk NLP - ACT Trening interpersonalny - TC Grupa Personal Efficiency Program PEP Zarządzanie Regionem - Mercuri International Poland Techniki aktorskie w biznesie - Aktren Negocjacje - DOOR Coaching zaawansowany - House of Skills

Figure 2. GoldenLine.pl – exemplary profile

whether profiles coming from diverse sources describe the same person. If so, an aggregated profile is created to keep an up-to-date information along with a revision history. These profiles are stored internally in the system. End users communicate with the system through a graphic user interface that allows for composing queries and visualizing the resulting profiles. In this paper we concentrate on the first step of the information flow in the eXtraSpec system, namely on the extraction component – EXT.

2.1. Motivating example

The example we describe here relates to the problem of recruitment of new employees to be involved in a new project launched by a Polish company XYZ S.A. Suppose HR executives of XYZ didn't find appropriate employees already working for the company who can take new tasks related to another project. XYZ has to start looking for new people by posting job advertisements or hiring "head-hunting" companies. These two methods are costly, so HR department workers start to browse social portals for professionals to find prospective employees. However, this task is very time consuming, as each profile has to be checked manually for validity and against a profile of the ideal candidate. Moreover, people who have accounts on several portals often do not update all of them, so HR department can not be sure that the information is up-to-date.

All these issues reveal the need for a tool that is able:

- to process automatically user profiles from several popular Polish portals for professionals,
- to extract from web pages the most relevant information about candidates,
- to retrieve profiles of candidates that possess concrete characteristics,
- to track changes in candidates' profiles manifested on various portals,
- to cope with different vocabulary used to describe the same concepts.

Having a system that performs these tasks automatically, the process of recruitment can by shortened. Even if the proper candidate is not found, the system should suggest to compose a team of people that together would be able to accomplish all required tasks in the new project of XYZ.

The place of extraction component within such system is to retrieve profiles from websites, extract from them only relevant information and transform this information to the form processable by other components of the system, responsible for query answering and teams composition.

2.2. Information lifecycle in eXtraSpec

The complete architecture of the system is presented in [11]. The following components are responsible for the creation and processing of profiles in eXtraSpec:

1. Extraction (EXT) – responsible solely for retrieval of documents from dedicated sources, extraction of relevant content and creation of extracted profiles (see section 4).
2. Normalization (NOR) – its task is to disambiguate the names of organizations, jobs, and capabilities. In order to fulfill this task heuristic methods and dictionaries are used. Additionally, NOR takes care of time references. In effect, normalized profiles are created.
3. Fusion (FUS) – aggregates information from multiple profiles. One of the most important tasks of FUS is to determine whether several profiles belong to the same person or not.
4. Reasoning (REA) – processes user query for finding an expert having specific skills or composes a team that together can possess requested skills.
5. Graphical user interface (GUI) – a tool for users to formulate queries to the system and to visualize the results.

The modular architecture of the system allows for clear distinction of the stages of documents processing and verification of efficiency of each component. The flow of documents within eXtraSpec architecture and dependencies between components are presented on the Figure 3.

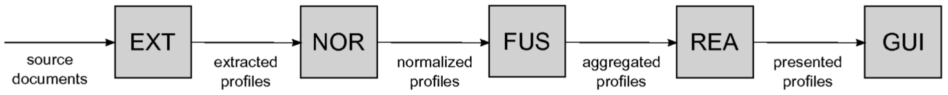


Figure 3. Dependencies between components of eXtraSpec system (based on [11])

3. Related work

The task of automatic information extraction from Web pages is rather well recognized in literature [1–4], but to our best knowledge, there were no attempts to develop a method of Web information extraction from Polish portals for professionals.

The ultimate goal of information extraction in the context of WWW is the translation of semi-structured information being in the form of HTML document, to a structured information (most frequently relational) [13, 14]. In contrast to text extraction, information extraction from Web pages uses less formal grammars, but mostly relies on the structure of processed documents (HTML, XML, PDF) and links between them. When used together with a Web source navigation techniques, extraction can compose the data retrieved from different sources and create a coherent data sets [15].

Manual extraction is a very time-consuming task, especially for a big amount of web pages. It is possible to develop a wrapper that extracts and navigates in the source document. The systems that were already presented can be categorized according to their capabilities and degree of realization of extraction tasks [16–18]. Most authors consider such distinctive criteria like: dependence on the source structure, ability to extract from multiple pages, need to manual creation of wrappers, used formalism (grammars, rules, statistics, logics)

The only approach to processing CVs in Polish was signalized only once [5]. A few exceptions are the standardized CV documents like Europass CV[4] or HR-XML Consortium[5] CV template that may be easily processed for further team building [6].

[4] http://europass.cedefop.europa.eu

[5] http://www.hr-xml.org

Therefore, in further phases of extraction the technique of Named Entity Recognition with statistical classification of entities is taken advantage of. Apart from this, eXtraSpec system uses semantics in the form of domain ontologies to prepare meaningful profiles. In literature, several semantically enhanced information extraction systems like Vulcain [7], SOBA [8], OBIE [9] or KIM [10] were presented, but again no method was dedicated for the Polish language.

4. Extraction method

In the described extraction component of the eXtraSpec system we are concentrated on the processing of semi-structured documents. The main task of this component is to extract concrete attributes of the profile from source documents. To do this, we create a tree of extraction rules, in which each rule is responsible for extracting elements of the profile. EXT component performs a transformation of HTML tree to another tree structure – in this case a profile tree, which is represented as XML document (see Figure 4). After one run of EXT over retrieved HTML documents there is one corresponding profile for each document. Such transformation could be performed with XSLT, but his technique is not optimal for the needs of eXtraSpec due to cumbersome maintenance and complexity. In order to allow for using the same extraction method for different Web sources, we proposed a hierarchical extraction algorithm that operates on extraction rules represented as XPath expressions and regular expression for advanced string transformations. Extraction component processes mainly HTML documents that come from business-related Polish social portals like GoldenLine or Profeo. Because of their internal form, they can be processed using analysis of tree structure created by HTML tags.

In the case of already mentioned business portals, the HTML structure enforces only the data schema. This means that apart from the structured data, there is information stored also in an unstructured way, like the description filled in by hand in natural language.

4.1. Structure of extracted profile

Like it was presented on Figure 4, the extracted profile has a tree-like structure. A single profile is created for every document that is processed in the system Leafs in the tree contain text passages which were extracted from the document. For the purpose of representation of a profile we use XML, for which adequate schema was prepared.

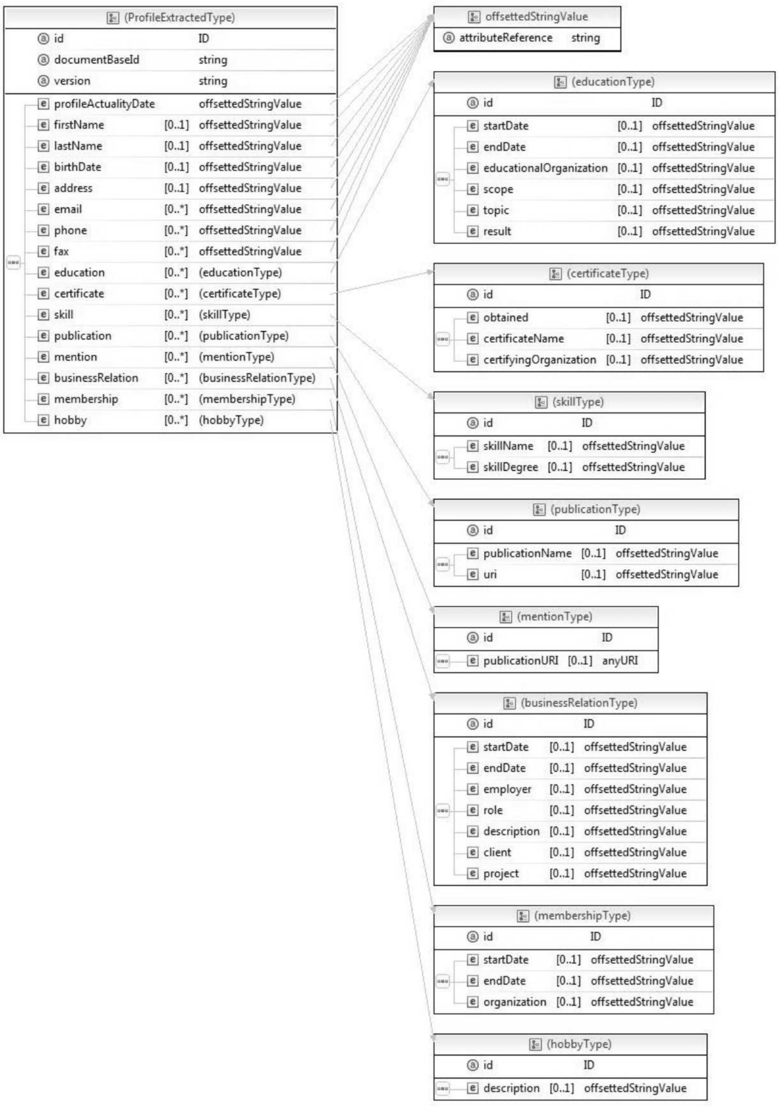

(ProfileExtractedType)
id ID
documentBaseId string
version string
profileActualityDate offsettedStringValue
firstName [0..1] offsettedStringValue
lastName [0..1] offsettedStringValue
birthDate [0..1] offsettedStringValue
address [0..1] offsettedStringValue
email [0..*] offsettedStringValue
phone [0..*] offsettedStringValue
fax [0..*] offsettedStringValue
education [0..*] (educationType)
certificate [0..*] (certificateType)
skill [0..*] (skillType)
publication [0..*] (publicationType)
mention [0..*] (mentionType)
businessRelation [0..*] (businessRelationType)
membership [0..*] (membershipType)
hobby [0..*] (hobbyType)
offsettedStringValue
attributeReference string
(educationType)
id ID
startDate [0..1] offsettedStringValue
endDate [0..1] offsettedStringValue
educationalOrganization [0..1] offsettedStringValue
scope [0..1] offsettedStringValue
topic [0..1] offsettedStringValue
result [0..1] offsettedStringValue
(certificateType)
id ID
obtained [0..1] offsettedStringValue
certificateName [0..1] offsettedStringValue
certifyingOrganization [0..1] offsettedStringValue
(skillType)
id ID
skillName [0..1] offsettedStringValue
skillDegree [0..1] offsettedStringValue
(publicationType)
id ID
publicationName [0..1] offsettedStringValue
uri [0..1] offsettedStringValue
(mentionType)
id ID
publicationURI [0..1] anyURI
(businessRelationType)
id ID
startDate [0..1] offsettedStringValue
endDate [0..1] offsettedStringValue
employer [0..1] offsettedStringValue
role [0..1] offsettedStringValue
description [0..1] offsettedStringValue
client [0..1] offsettedStringValue
project [0..1] offsettedStringValue
(membershipType)
id ID
startDate [0..1] offsettedStringValue
endDate [0..1] offsettedStringValue
organization [0..1] offsettedStringValue
(hobbyType)
id ID
description [0..1] offsettedStringValue

Figure 4. The schema of a profile produced by EXT component

We took into account all the interesting features of the expert at a fine level of granularity, which results in a document similar in structure to a complex CV. The following classes of information were taken into account:

- basic personal data (first and last name, birth date, address, email, phone and fax numbers)
- education – for each entry concerning education we record start and end date of a given education phase, organization that was responsible for

this stage, topic and scope of education and its level (primary school, secondary school, college, university etc.)
- certificates – which are characterized by date, its name and the name of certifying organization
- skills with skill name and declared level of proficiency
- publications – described with the name (or reference to the publication) and link to the publication on the Web
- mentions – this category contains a list of references to locations in different documents, where the person was mentioned
- work experience – each entry contains start and end dates for a work period, name of the employer, occupied post and the description, and, for projects – project name and name of the customer
- membership – start and end dates of the membership, the name of the organization
- hobby with the description field

Except the above mentioned categories, certain metadata about the profile are stored: its identification number, the date that it was created and reference to the original document.

4.2. Hierarchical extraction algorithm

The extraction algorithm works by processing HTML web pages discovered in the selected sources and producing for each document an extracted profile. As the sources of data for the EXT component selected social portals were chosen, where people publish their CVs or similar information, using predefined forms with varying (but usually limited) degree of flexibility.

The extraction approach that was chosen is based on two assumptions. First, that the target and source data is a tree, and second, that the source data has a fixed structure that follows tree-based approach. A fixed structure means that a) all categories of nodes in a tree are known and fixed, b) their position in relation to each other does not change (although a node may be missing) and c) recursion is not allowed (nodes can not be nested within each other infinitely). The first assumption is true for the target data (profile extracted) because it was defined to follow it, but it might not be true for certain web pages, which do not follow tree-like organization of data. That would include for example web pages which are built in tabular orientation of data. An important distinction has to be drawn between using tables to position the data – which does not preclude using tree structures, and using tabular data orientation. For example, a web page where people would be listed in columns of a table, with their personal data placed in rows is not a tree-like structure and consequently can not be handled in the approach chosen for extraction of information in our case. However, by observing a number of portals we found that a non-tree structuring does not occur in practice, therefore this limitation is not significant.

The approach to information extraction that we have chosen is based on hierarchy of extraction rules. The hierarchy of rules resembles hierarchy of data in the target profile to be extracted – each rule extracts part of the HTML document that relates to certain element in the extracted profile. That part can be further processed by child rules to extract more detailed pieces of information.

In each rule hierarchy there is a single root rule, that would extract the part of HTML document tree which includes the whole information required to build a profile.

Each extraction rule may point to a target element of the extracted profile. There can be rules which do not point to a profile element – they are used as grouping "facilities" for child rules.

Each extraction rule indicates, whether the element that it points to is atomic (an expected result is a simple leaf element – for example first name), compound (target element may be a complex element like education entry), or is a collection of elements of the same type (expected extraction result is a set of education entries).

Each rule fulfills a condition, that its extraction result has to encompass all the information required to fill the target of the rule (even though it can be done by child rules). This limits the child rules to extract only part of the subtree extracted by the parent rule.

We transform each obtained HTML document into a valid XML document by cleaning it with standard HTML Tidy[6] utility and the extraction rules are applied in a descending order by processing the extraction hierarchy.

Such approach can be formalized as transformation of a source tree into a target tree, and consequently described for example in XSLT. However, in order to avoid the complexity of XSLT (or other formalisms similar in power) we chose to limit ourselves to a subset of transformations that are possible, which is motivated by the assumptions described above about the structure of source and target data, and our desire to simplify the creation of the extraction rule hierarchies.

Since the extraction rules hierarchy is dependent on the target data structure, which is fixed, the hierarchies do not differ in general structure.

[6] http://tidy.sourceforge.net

The most important differences concern the rules, which in turn are mainly dependent on the source structure. As a consequence, each document source has to have it's own hierarchy of extraction rules prepared. At a current stage we don't use any mechanisms to automatically discover the source structure and build the rules automatically.

4.3. Structure of extraction rules

Hierarchical extraction performed by EXT takes advantage of the extraction rules to extract specific data from source documents. For the sake of easier management of extraction rules, we decided to keep the rules for different sources separately. This is also due to the uniqueness of each source, because each portal sticks to its own presentational structure of the business profile. Therefore, it is impossible to apply the same extraction rule in a variety of sources, since there are differences in the DOM models of various business portals.

Extraction rules, similarly to the extraction method, are hierarchical rules. This means that each rule possesses a given parent rule which must be executed before. Otherwise, the execution of child rule will not be successful and the requested fragment of source document will not be extracted. In the current version of EXT component, extraction rules are being formed twofold:

- as XPath expressions
- as regular expressions (Regex)

Because of the fact that source documents are Web pages, the majority of information is being extracted by XPath expressions. As a result, content blocks or just text strings are being obtained. If such result can not be directly stored in some field of profile extracted, due to data type mismatch, there is an additional need to process it. Sometimes, more detailed XPath rules are used, and sometimes we use Regex commands to process text strings. Regex allows for a sophisticated transformations of strings, which is useful is many cases. As a simple example may give the phrase "John Smith" which we can obtain in a result of XPath expression. Then we can use regular expression to split this string into two: "John" and "Smith", and later put them directly in the PE.

An exemplary extraction rule is presented in the Listing 1. This rule has id=5, and its result is being stored in the PE field named "address". As one can see, this rule is formulated in XPath. It has a parent extraction rule (id=1), that must be executed before. Arity value means that the result of this rule is a single, simple object, that can be directly put to the profile extracted.

```
<extractionRule id="5" relatedElement="address"
parentExtractionRuleId="1" language="XPath" arity="single">
/html/body/div[12]/div/div/table/tbody/tr/td[2]/div/dl[1]/dd
</extractionRule>
```

Listing 1. Example extraction rule with its target element (address in the extracted profile), indication of parent rule and arity

Although we have developed separate sets of extraction rules for various data sources, all rule sets use the same schema. This allows the user to easily analyze the syntax of rules for the documents coming from different sources. The XML schema of extraction rule is presented below on the Listing 2.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified"
elementFormDefault="qualified"
targetNamespace="http://extraspec.org/schema/ExtractionRulesSchema"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="extractionRules">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="0" maxOccurs="unbounded" name="extractionRule">
<xs:complexType>
<xs:simpleContent>
<xs:extension base="xs:string">
<xs:attribute name="language" type="xs:string" use="required" />
<xs:attribute name="parentExtractionRuleId" type="xs:string" use="required" />
<xs:attribute name="id" type="xs:unsignedByte" use="required" />
<xs:attribute name="relatedElement" type="xs:string" use="required" />
<xs:attribute name="arity" use="required">
<xs:simpleType>
     <xs:restriction base="xs:string">
     <xs:enumeration value="single">
     </xs:enumeration>
     <xs:enumeration value="compound">
     </xs:enumeration>
     <xs:enumeration value="collective">
     </xs:enumeration>
     </xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="sourceName" type="xs:string" use="optional" />
<xs:attribute name="representationMimeType" type="xs:string" use="optional" />
</xs:complexType>
</xs:element>
</xs:schema>
```

Listing 2. XML schema of the extraction rule

Firstly, we define the namespace used by extraction rules. Then, the following attributes are being defined:

- language – defines the language of the extraction rule. Currently two options are possible: XPath and Regex. This attribute is required for well-formedness of the rule.
- parentExtractionRuleId – the id of parent rule that must be executed before. This attribute is required for well-formedness of the rule.
- id – unique id number of the extraction rule. This attribute is required for well formedness of the rule.
- relatedElement – denotes the name of the field of PE, in which the result of this rule is stored. This attribute is required for well-formedness of the rule.
- arity – denotes the type of content block that is extracted by this rule. This attribute is required. We have three types of arity:
  - single – for extracted elements that in PE are of type Offsetted-StringValue. This is the simplest data type used in PE.
  - collective – used when one rule extracts several elements of the same type. An example may serve the extraction of block with employers, in which the element "professional experience" is described by the same attributes.
  - compound – complex type in which one extraction rule processes a content block consisting several elements of various type or a collection of elements.
- source name – the name of data source processed by a given set of extraction rules. This attribute is optional.
- representationMimeType – describes document type, on which the rules are executed. This attribute is optional.

Extraction rule sets are stored in XML format, so no special toolkit is required to manage them. Rule sets are kept as XML files and stored in a dedicated folder, accessible only by EXT component.

5. Discussion

There are two questions to discuss regarding the described approach to information extraction: efficiency of the approach and the limitations of the method.

To consider efficiency of the hierarchical extraction rules, one has to take into account, that the rules in our case (contrary to other information extraction scenarios) are not statistical in nature. After an analysis of popularity of Polish business portals, we decided to serve the two most popular in the first version of extraction component. For GoldenLine we created a set of 32 extraction rules, whereas Profeo is being processed by a separate set of 34 extraction rules. We base on the assumption that the sources of documents have a fixed structure. It stems from the fact, that web portals do not allow for upload of arbitrary document as a user profile, but allow only to fill in the form, which has fixed, hierarchical structure. Therefore, if this structure is guessed correctly, and the extraction rules are crafted to match it, there is no possibility of a wrong result of the extraction, if only the form was filled correctly. The only limit of the method in that respect is the granularity of information it can extract – if certain fields in the form allow for arbitrary text (for example descriptions), it is hard to create extraction rules to handle it because this type of content does not adhere to the fixed tree-like structure assumption. For that we envision applying other techniques (originating in Natural Language Processing) which are more suitable for dealing with free text content.

The limitation of the method stems from assuming tree-like structure of the source. Although it is supported by our observation that other structures do not occur in our application domain, it is worth to discuss other motivations for such simplification. One of our goals was to simplify the creation of the whole extraction rules bundle. We achieved it by organizing it into a hierarchical, non-recursive structure which is the simplest and yet still powerful enough to extract into a required target structure. Second, such a simplification enables automated discovery of source structure. Although complex, it is feasible for a limited number of sources. Algorithms for discovery of web page structures were already researched [12] and they can be tuned to the tree-like character of the sources in our application domain.

6. Conclusions and future work

In this paper we have presented an extraction component called EXT, being a part of the eXtraSpec project. EXT is capable of processing web pages written in the Polish language in order to extract the information relevant for the needs of expert finding and team building. To perform its tasks, EXT uses an original algorithm of content extraction, which is done in accordance to the hierarchy of a given HTML document. This means that the structure of the web page is reflected in a resulting profile extracted and the content of specific fields is transformed by XPath or Regex and settled in a corresponding field.

The directions of future work also include development of text processing techniques to cope with fields that are manually filled by humans, because automatic extraction from such fields using XPath or regular expressions is cumbersome and sometimes does not give satisfactory results. So far, the developed extraction rules cope very well with structured information presented on these portals, but currently we do not process natural language full text phrases that are also present in the business profiles on these portals. Natural language processing of Polish phrases is one of the possible future research tracks in the project. This technique would be especially interesting in case of processing the content blocks that can be freely filled by users. In such cases, it is not possible to use predefined extraction rules, as some heuristic methods are needed.

We also plan to prepare an extensive evaluation of extraction rules by comparing the results obtained automatically and manually.

BIBLIOGRAPHY

[1] Rahardjo, B. and Yap, R. H. 2001. *Automatic information extraction from web pages.* in Proceedings of the 24th Annual international ACM SIGIR Conference on Research and Development in information Retrieval (New Orleans, Louisiana, United States). SIGIR '01. ACM, New York, NY, 430–431.

[2] Chang, C., Hsu, C., and Lui, S. 2003. *Automatic information extraction from semi-structured Web pages by pattern discovery.* Decis. Support Syst. 35, 1 (Apr. 2003), 129–147.

[3] Vadrevu, S. 2008. *Automated Information Extraction from Web Pages Using Presentation and Domain Regularities.* Doctoral Thesis. UMI Order Number: AAI3304898., Arizona State University.

[4] Novotny, R., Vojtas, P., and Maruscak, D. 2009. *Information Extraction from Web Pages.* in Proceedings of the 2009 IEEE/WIC/ACM international Joint Conference on Web intelligence and intelligent Agent Technology – Volume 03 (September 15–18, 2009). Web Intelligence & Intelligent Agent. IEEE Computer Society, Washington, DC, 121–124.

[5] Kaczmarek, T., Kowalkiewicz, M., and Piskorski, J. (2005). *Information Extraction from CV.* in W. Abramowicz (Ed.), 8th International Conference on Business Information Systems (pp. 185–189). Poznań: Wydawnictwo Akademii Ekonomicznej w Poznaniu.

[6] De Meo, P., Plutino, D., Quattrone, G. and Ursino, D. (2010). *A team building and team update system in a projectised organisation scenario.* in Int. J. Data Mining, Modelling and Management, Vol. 2, No. 1, pp. 22–74.

[7] Todirascu, A., Romary, L., and Bekhouche, D. 2002. *Vulcain – An Ontology-Based Information Extraction System.* in Proceedings of the 6th international Conference on Applications of Natural Language To information Systems-Revised Papers (June 27–28, 2002). B. Andersson, M. Bergholtz, and P. Johan-

nesson, Eds. Lecture Notes In Computer Science, vol. 2553. Springer-Verlag, London, 64–75.

[8] Buitelaar, P., Cimiano, P., Frank, A., Hartung, M., and Racioppa, S. 2008. *Ontology-based information extraction and integration from heterogeneous data sources.* in Int. J. Hum.-Comput. Stud. 66, 11 (Nov. 2008), 759–788.

[9] K. Bontcheva. 2004. *Open-source tools for creation, maintenance, and storage of lexical resources for language generation from ontologies.* in 4th Conf. on Language Resources & Evaluation, Lisbon, Portugal.

[10] Popov, B., Kiryakov, A., Ognyanoff, D., Manov, D., and Kirilov, A. 2004. *KIM – a semantic platform for information extraction and retrieval.* Nat. Lang. Eng. 10, 3–4 (Sep. 2004), 375–392.

[11] Abramowicz W., Kaczmarek T., Stolarski P., Węcel K., Wieloch K., *Architektura Systemu Wyszukiwania Ekspertów eXtraSpec*, TWZP 2010, Prace Naukowe Akademii Ekonomicznej we Wrocławiu, Katowice 2010.

[12] Kowalkiewicz, M., Orlowska, M., Kaczmarek, T. i Abramowicz, W. (2006). *Towards more personalized Web: Extraction and integration of dynamic content from the Web.* in Proceedings of the 8th Asia Pacific Web Conference APWeb 2006. Harbin, China: Springer Verlag.

[13] Line Eikvil. *Information extraction from world wide web – a survey.* Technical report. 1999.

[14] Andrew McCallum, William W. Cohen. *Information extraction from the world wide web.* Tutorial, 2002.

[15] Ion Muslea. *Extraction patterns for information extraction tasks: A survey.* The AAAI-99 Workshop on Machine Learning for Information Extraction, 1999.

[16] Boris Chidlovskii, Uwe M. Borghoff, Pierre-Yves Chevalier. *Towards sophisticated wrapping of webbased information repositories.* 5th International RIAO Conference, p. 123–135, 1997.

[17] Jane Yung jen Hsu, Wen tau Yih. *Template-based information mining from html documents.* 14th National Conference on Artificial Intelligence, strony 256–262, 1997.

[18] Juliana S. Teixeira, Berthier A. Ribeiro-Neto, Alberto H. F Laender, Altigran S. da Silva. *A brief survey of web data extraction tools.* SIGMOD Record, 31(2): 84–93, 2002.

Tomasz Kaczmarek
Poznan University of Economics
Faculty of Informatics and Electronic Economy
Department of Information Systems
t.kaczmarek@kie.ue.poznan.pl

Dominik Zyskowski
Poznan University of Economics
Faculty of Informatics and Electronic Economy
Department of Information Systems
d.zyskowski@kie.ue.poznan.pl

Adam Walczak
Poznan University of Economics
Faculty of Informatics and Electronic Economy
Department of Information Systems
a.walczak@kie.ue.poznan.pl

Witold Abramowicz
Poznan University of Economics
Faculty of Informatics and Electronic Economy
Department of Information Systems
W.Abramowicz@kie.ue.poznan.pl

Wiesław Półjanowicz
Department of Applied Informatics in Education,
Institute of Computer Science, University of Bialystok

Robert Latosiewicz
Department of Rehabilitation and Physiotherapy,
Medical University of Lublin

THE EFFECTIVENESS OF E-LEARNING IN THE CONTEXT OF SELECTED MEDICAL SUBJECTS

Abstract: The paper presents an attempt to assess methods for measuring the efficiency and quality of e-learning classes conducted on the basis of the selected medical subjects at the Faculty of Health Sciences of Medical University of Bialystok. Presented are the most important elements that should be used in the study of the quality and efficiency of e-learning. The effectiveness of e-learning ranged from 76% to 82% according to Kirkpatrick's model. Moreover, the e-learning method of teaching was rated by students as more convenient and easier due to continuous access to interactive learning materials.

Key words: e-learning, the LMS/LCMS class system, efficiency and quality of education

Introduction

Recently, new forms and methods of teaching are increasingly used in education at the academic level. Traditional teaching methods (lectures, seminars) are, in many cases, replaced or supplemented by distant methods. They include interactive multimedia presentations, films, sound recordings and computer animations and others. Educational content is transmitted via the Internet or locally via intranet or extranet.

Higher educational courses related to medical disciplines such as nursing, midwifery, physiotherapy, emergency medicine require students to master the wide range of information in many medical fields. Traditional lectures or seminars are usually limited to place and time. That makes assimilation of knowledge often difficult and inadequate. Distant learning forms (like e-learning platform class LMS/LCMS) provide different opportunities.

The content is delivered by means of modern forms of communication and throughout a wide range of time. It provides the opportunity to make study more effective. A well prepared e-learning course contains an appropriate selection of didactic material and forces students to improving their efficiency in the educational process. The specificity of medical studies requires many practical skills that are hard to be replaced by e-learning form of activity. But many elements of theoretical knowledge can be successfully transmitted in a form of distant learning [1, 2, 3, 4].

In the article, the authors attempt to assess the effectiveness and quality of e-learning courses conducted on the basis of the selected medical subjects at the Faculty of Health Sciences, Medical University of Bialystok, Poland.

Efficiency and quality of teaching

The traditional process of teaching of medical students is a complex process consisting of several essential elements: theoretical (lectures, seminars), practical and specialist courses (mainly clinical). Some components of this process can be replaced by the formula of e-learning. This means that a well prepared e-learning course is able to improve the traditional teaching process in the field of medicine. But the question remains: how effective are these changes?

An important factor in assessing the quality of education is its efficiency. In the evaluation of the effectiveness of distant learning a model of an "e-course" is commonly implemented, in which the teaching material is placed as separate lessons on an IT platform in an electronic format.

The quality of education consists of substantive and methodological content of teaching materials and learning purposes compatible with the selected model of teaching: a model of traditional or e-learning. In the case of an on-line model, it is important to add technical aspect of informatics solutions (reliability, functionality, ease of use of the e-learning method).

The effectiveness of education is usually understood as the degree of fulfillment of educational goals. Effectiveness of teaching can also be defined as an improvement in the performance of a chosen activity. The result of education is mastery in new content, that is the ability to acquire knowledge (amount of knowledge absorbed per unit of time for the total amount of knowledge). It can be measured by a spectrum of different tests, colloquies and speeches to the forum [5].

The growth in knowledge can be calculated according of the following formula:

$$\begin{array}{c}\text{Indicator of}\\ \text{growth of}\\ \text{knowledge}\end{array} = \frac{\begin{array}{c}\text{test results after the training –}\\ \text{– test results before the training}\end{array}}{\begin{array}{c}\text{results to be achieved –}\\ \text{– the test results before the training}\end{array}} \times 100\%$$

The results obtained are values ranging from 0 to 100%, which are in direct proportion to the amount of knowledge students have acquired during the course [8].

Studies on the quality of e-courses shall include the following elements:

- evaluation of prepared teaching materials (in terms of content, logical structure, and understanding),
- personal needs of students due to lack of direct contact with the lecturer replaced by forums, chat, video conferencing,
- evaluation of teaching methodology used to set learning objectives (learning by assimilation – lecture, work with a book, learning by discovery – the method of cases, exchange ideas, educational games, learning by experiencing (video, slideshow, art), learning by doing (the method of cases, simulation of the phenomena in question exercises),
- evaluation of the way of organization of an e-course,
- evaluation of self-activity of students (statistics of logins students to the platform, time spent on the acquisition of knowledge, the activity of students in the various modules (based on the latter, it can be defined as the complexity of the course),
- assessing the teaching process (mentoring, guidance on the accuracy of the solutions, hint system).

According to Kirkpatrick [7] an evaluation of effectiveness of e-learning should include:

- student satisfaction survey (system evaluation questionnaires) – this is the <u>level of response</u> according to the Kirkpatrick's model,
- testing students' knowledge growth rate (in the way mentioned above – this is the <u>level of science</u> according to the Kirkpatrick's model,
- study of behavior and performance of students (observation and analysis of work) – this is the <u>level of behavior</u> according to the Kirkpatrick's model,
- examination of the financial effects of the project (level of return of investment in education) – that is the <u>level of results</u> by Kirkpatrick's model [6, 7].

In summary, the problems in assessing the quality and effectiveness of e-learning can be grouped in two interdependent aspects: assessing the level of e-course preparation and the evaluation of blended learning (traditional and e-learning) and their compatibility.

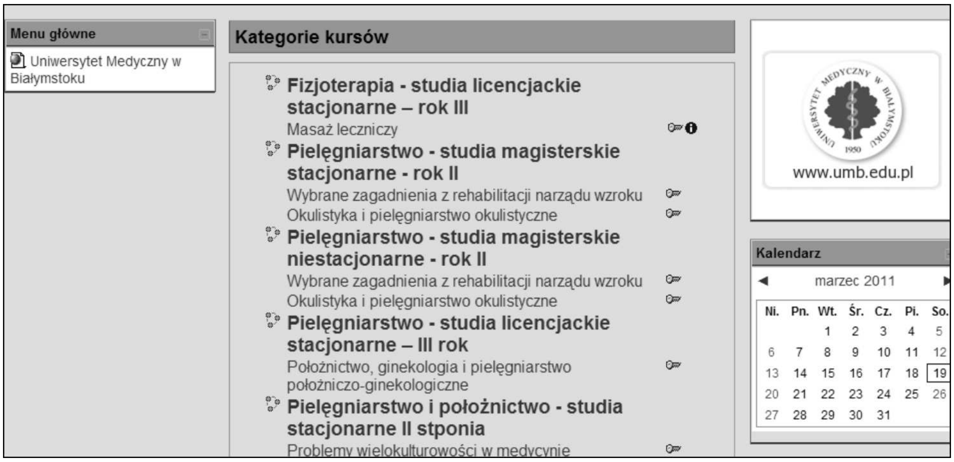
Measuring the effectiveness of e-learning based on selected medical subjects

Material and methods

The study was conducted among 325 full-time, second degree students of Medical University of Białystok, in the Faculty of Health Sciences. The investigation was approved by the Regional Committee for Medical Research Ethics, and all students gave their consent before participation.

For the purposes of research four subjects in the form of distant learning (e-courses) have been prepared. They were: obstetrics, gynecology and obstetrics-gynecology nursing, ophthalmology and ophthalmic nursing, rehabilitation of visual organ and therapeutic massage.

The students of each group (subject) were divided into two subgroups within the same subject. One subgroup had a distant model of education, based on e-learning platform implemented on the class LMS/LCMS (Moodle) system (Figure 1). The second subgroup had classes in the traditional model of education (control group).



Menu główne

Uniwersytet Medyczny w Białymstoku

Kategorie kursów

Fizjoterapia - studia licencjackie stacjonarne – rok III
Masaż leczniczy

Pielęgniarstwo - studia magisterskie stacjonarne - rok II
Wybrane zagadnienia z rehabilitacji narządu wzroku
Okulistyka i pielęgniarstwo okulistyczne

Pielęgniarstwo - studia magisterskie niestacjonarne - rok II
Wybrane zagadnienia z rehabilitacji narządu wzroku
Okulistyka i pielęgniarstwo okulistyczne

Pielęgniarstwo - studia licencjackie stacjonarne – III rok
Położnictwo, ginekologia i pielęgniarstwo położniczo-ginekologiczne

Pielęgniarstwo i położnictwo - studia stacjonarne II stponia
Problemy wielokulturowości w medycynie

www.umb.edu.pl

Kalendarz

◄ marzec 2011 ►

Ni.	Pn.	Wt.	Śr.	Cz.	Pi.	So.
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Figure 1. The main window of e-learning platform system class LMS/LCMS (Moodle)

At the end of each course evaluation survey was carried out on the above mentioned aspects of testing the quality and effectiveness of e-learning as compared to the traditional teaching model. The questionnaire survey consisted of three parts: sociodemographic data, opinions about the activities carried out and opinions about the effectiveness of education. For each subject final exams were conducted in form of a single-choice test simultaneously in both subgroups. Professional knowledge in both subgroups was compared by assessing final scores ranging from 2 (bad) to 5 (very well). Degree of satisfaction with the selected activities (process of teaching, material availability, etc.) was assessed using specially designed questionnaire.

The computer software package Statistica 9.0 PL (Statsoft Inc., Poland) was used for the data analysis. A $p < 0.05$ was considered statistically significant.

Results and discussion

In the study dominated students from cities of more than 80 thousand inhabitants (table I). All the persons from the e-learning group had continuous access to the Internet, while among the 173 people in the control group only 22 persons (12%) did not have permanent access to the Internet. Daily use of the Internet declared more than 68% of the students in both subgroups.

Table I

Characteristics of socio-demographic groups of students surveyed

	e-learning method (n = 152)		traditional method (n = 173)	
	No. of persons	(%)	No. of persons	(%)
Domicile				
City > 80 thus. inh.	98	64%	106	61%
City < 80 thus. inh	36	24%	26	15%
village	18	12%	41	24%
Permanent access to the Internet				
yes	152	100%	151	87%
no	0	0%	22	13%
How often do you use the Internet?				
daily	120	79%	117	68%
almost daily	20	13%	17	10%
regularly 2–3 times a week	11	7%	19	11%
once a week	1	1%	3	2%
several times a month	0	0%	11	6%
occasionally	0	0%	6	3%
do not use	0	0%	0	0%

Comparable results were obtained with final exams in both subgroups with a slight predominance of the average ratings in subgroups of e-learning at the mean level of 0.23 pts, which was statistically significant ($p < 0.05$). The difference was mostly evident in the subject – obstetrics, gynecology and obstetrics-gynecology nursing (0.33 pts), which was statistically significant ($p < 0.05$). There was no statistically significant difference between both subgroups in two subjects related to ophthalmology (0.12 pts, $p > 0.05$). Slightly higher average final score difference (0.2 pts.) was obtained in the subgroup of traditional method in the subject: therapeutic massage. Still, it was not statistically significant ($p > 0.05$).

All the students (100%) of the e-learning group declared their willingness to participate in this method of teaching again while only 62% students of the traditional group were interested in this method. Degree of preparation of the learning content has been rated good and very good by 96% of students in the group of e-learning and by 93% in the traditional group. Equally high was rated the possibility of distant learning methods to enhance the effectiveness of education (94% students of e-learning group *vs.* 91% students of traditional method).

E-learning method as a way to widen the knowledge and skills of students positively evaluated 93% respondents from the e-learning group.

The effectiveness of learning can be presented in two ways: educational and financial [9]. Educational dimension is, inter alia, an increase in knowledge of the student. In this study the e-course students remembered better theoretical knowledge of the course (68% compared to 63% in the traditional method). The effectiveness of the e-learning process was as high as 90% in the e-learning group, which can be related to good accessibility of the e-learning platform and the readability of educational materials placed there. The financial aspect is the cost (meaning relationship between of acquired knowledge and the implementation cost) has not been performed yet.

Our results indicate a high level of usefulness of e-learning classes. High popularity of the e-learning form may result from permanent and interactive access to the knowledge offered in the on-line course (lesson, quiz, task, activity SCORM, and forum). A student may repeatedly refer to the previously analyzed issue, which eases learning and acquiring knowledge, and also checking his/her knowledge in the area [1, 2].

The obtained results allow us to conclude that further development of distant education can be a very good form which will supplement traditional education rather than replace it completely. It is essential to continue additional research and analysis in this area and discuss the problem of efficiency and quality of e-learning in higher education, especially in terms of widening learning opportunities for people studying in medical fields.

Conclusions

1. Basing on the performed analysis it can be concluded that the effectiveness of e-learning ranged from 76% to 82% according to Kirkpatrick's model.
2. The e-learning method of teaching was rated by students as more convenient and easier due to continuous access to interactive learning materials.
3. Automated testing of knowledge implemented on the e-learning platform encourages students to regular work and self-study.
4. Substantively and methodologically well prepared didactic materials presented full-time on the e-learning platform increase the opportunity to prepare students themselves for final examinations in the respective fields of knowledge.

REFERENCES

[1] Półjanowicz W., Latosiewicz R., Niewiński A., Milewski R.: E-learning in students' education in Medical University of Bialystok. Bio-Algorithms and Med-Systems. Medical College, Jagiellonian University, 2009; 5:111–115.

[2] Półjanowicz W., Latosiewicz R., Kulesza-Brończyk B., Piekut K., Kalisz A., Piechocka D. I., Terlikowski S. J.: Comparative analysis of e-learning and traditional teaching methods in the field of nursing in the Medical University of Bialystok. The chosen aspects of woman and family's health Vol. 2, s. 94–104, Bydgoszcz 2010.

[3] Półjanowicz W., Citko U.: Wykorzystanie b-learningu w kształceniu studentów informatyki Uniwersytetu w Białymstoku. Fenomen Internetu. Szczecin, 2008; 568–574.

[4] Hyla M.: Przewodnik po e-learningu, Oficyna Ekonomiczna, Kraków 2005.

[5] Bramley P.: Ocena efektywności szkoleń, Dom Wydawniczy ABC, Kraków 2001.

[6] Dąbrowski M.: Analiza pomiaru efektywności kosztowej procesów e-learningowych. E-mentor 2008; 5 (27): 18–26

[7] Kirkpatrick D. L.: Ocena efektywności szkoleń, Wydawnictwo Studia Emka, Warszawa 2001.

[8] Ryl-Zaleska M.: Metody oceny efektywności kształcenia on-line, http://www.e-edukacja.net/_referaty/9_e-edukacja.pdf.

[9] Mischke J., Stanisławska A. K.: Jak ocenić jakość i efektywność e-nauczania? COME UW, 2006.

Wiesław Półjanowicz
Department of Applied Informatics in Education
Institute of Computer Science
University of Białystok
wpoljan@uwb.edu.pl

Robert Latosiewicz
Department of Rehabilitation and Physiotherapy
Medical University of Lublin
rlatos@piasta.pl