

Logic Analysis of Natural Language Based on Predicate Linear Logic

**Zuzana Bilanová, Ján Perháč, Eva Chovancová,
Martin Chovanec**

Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics, Technical University of Košice

Letná 9, 042 00 Košice, Slovak Republic

e-mail: {zuzana.bilanova, jan.perhac, eva.chovancova, martin.chovanec}@tuke.sk

Abstract: This work discusses the formalization of sentence composition and the discovery of the semantic ambiguities of natural language. It also discusses the original connection between the logic area represented by predicate linear logic and ludics, as well as, the linguistic area represented by the Montague grammar. Montague grammar is a linguistic tool that allows analysing sentences in their extensional and intentional contexts. Predicate linear logic is a non-traditional logic of actions and resources where assumptions are consumed after the use of linear implication. Ludics uses proofs from predicate linear logic to analyse the strategies of actors in dialogues. The contribution of this work is to practically demonstrate this approach by translating a natural-language sentence into a predicate linear formula and describe it in time-spatial calculus.

Keywords: ludics theory; Montague grammar; predicate linear logic

1 Introduction

Logic analysis of natural language (LANL) [1] is a linguistics - logical discipline dealing with the interpretation of the meanings of natural language and the removal of various semantic ambiguities [2]. LANL has developed independently in two distinctive approaches called Montague grammar (by Richard Montague) [3] [4] [5] and transparent intensional logic (by Pavel Tichý) [6] [7]. In this article, the authors decided to rely on Montague grammar, widespread worldwide, despite the fact that many linguists agree that Tichý's global intensional approach is simpler and more transparent [8].

The aim of this paper is to combine the linguistic area (the Montague principles) with the resource-oriented character of linear logic combined with predicates, i.e. predicate linear logic [9] [10] and polarized locus trees of ludics [11] [12]. The original Montague theory operates with first-order predicate logic (FOPL) [13]

while ludics comes directly from propositional linear logic [14]. Predicate linear logic (extending propositional linear logic by predicate symbols and quantifiers) seems to be the appropriate tool allowing the combination of the following: handling the predicates, a resource-oriented character and modelling the meaning by extensions/intensions.

Therefore, the main focus, herein, is the link between linguistics and logic as shown in Fig. 1, focusing on a demonstration of new uses of predicate linear logic in computer science, deviating from the standards.

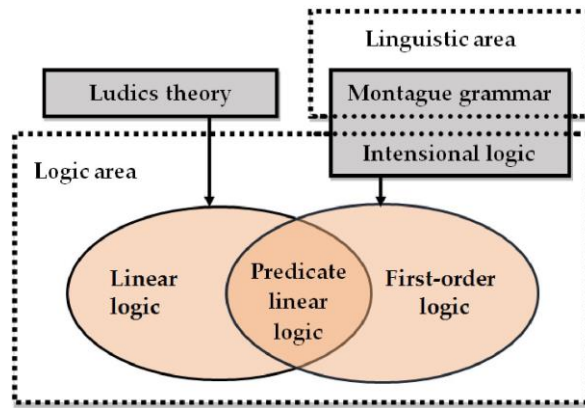


Figure 1

Interconnection of linguistics and logical areas

2 Montague Grammar

Richard Montague, author of the **Montague grammar** (MG) [15] [16] [17], claims that syntax and semantics of natural language and formal languages may be described only using a mathematically precise theory.

MG, despite its name referring to syntax, comprises two elements - syntactic and semantic - being in a definite relationship. Syntactic-semantic analysis of sentences is possible by using the principles of compositionality, the concept of possible worlds, FOPL, typed λ -calculus, categorical grammar and formal intensional language [18]. MG consists of these elements:

- Vocabulary – a finite set of elementary terms
- Grammar – a set of rules that allows creating complex expressions from simple expressions from the vocabulary
- Assign a meaning to elemental expressions using a basic set of objects

- Rules designed to determine the meaning of a compound term – carried out in the order of importance of the individual elements of the particular complex expression

The first two elements are related to the syntactic part - they allow keeping the infinite set of all possible sentences, while the latter two elements deal with the semantics - they allow to assign meanings to language expressions.

The syntactic part of Montague's work contains:

- Vocabulary - lexical units of the vocabulary are assigned to the appropriate categories
- Basic and derived categories - by using a rich categorical grammar
- Syntactic rules – these describe how the lexical units of basic and derived categories are transformed into compound terms

The semantic part of Montague's work contains:

- Syntactic and semantic definition of meaningful expressions of the language of intensional logic – a formal language is used for natural language interpretation
- Semantic types – these come with the corresponding syntactic categories
- Semantic rules – these come with the corresponding syntactic rules

The logical analysis of the language is carried out in two stages:

- 1 By applying **Montague categorical grammar** - natural language is reconstructed using a formal categorical language without semantics.
- 2 By applying **Montague intensional logic** - the resulting language is translated into another formal language of intensional logic, already having semantics. The basic principle in this is that all expressions in the process of translation are first "intensionalized"; however, a part of these are "extensionalized" later on.

Montague's approach is locally intensional - the meaning of an expression is its extension and it only accesses intensions in some specific contexts. Montague defines operators $\wedge$ and $\vee$. The $\wedge$ unary operator increases the intension - it modifies the expression E to an expression whose extension is intension E . The $\vee$ unary operator is inverse to $\wedge$ and decreases the intension.

3 Predicate Linear Logic

In this article, the authors introduce a predicate linear logic (PLL) [19] instead of propositional linear logic. It is necessary to describe the syntax, semantics and proof system of the PPL. Permissible forms of PLL formulas may be as follows:

- Elementary formula p and metavariables A, B , which express the action, reaction, literal or source
- Logical constants $\mathbf{1}, \mathbf{0}, \perp, \top$
- Atomic predicates $P(t, \dots, t)$, which represent an application of a predicate symbol P on a finite number of terms t
- Intensional logical connections $\otimes, \wp$ and extensional logical connections $\oplus, \&$
- Linear implication $\multimap$, negation $(.)^\perp$
- Exponentials "of course" $!$ and "why not" $?$
- Quantifiers - universal quantifier $\forall$ and existential quantifier $\exists$

Using the previous definitions, the **PLL syntax** can be described as:

$$\begin{aligned} A ::= a_n \mid \mathbf{1} \mid \mathbf{0} \mid \top \mid \perp \mid P(t, \dots, t) \mid A \otimes B \mid A \&B \mid \\ A \oplus B \mid A \wp B \mid A \multimap B \mid A^\perp \mid \\ !A \mid ?A \mid (\forall x)A \mid (\exists x)A \end{aligned} \quad (1)$$

The syntax of the terms t is the following:

$$t ::= x \mid c \mid f(t, \dots, t) \quad (2)$$

where x is a variable, c is a constant and $f(t, \dots, t)$ is an application of a functional symbol on the terms.

In this article, the authors focus on the Heyting semantic tradition and intensional fragment of linear logic. The Heyting semantic tradition deals with the meanings of formulas ($\mathbf{1}$ for a linear sense and $\perp$ for a linear nonsense). The **intensional fragment of PLL** can be described using the following syntax:

$$A ::= a_n \mid \mathbf{1} \mid \perp \mid P(t, \dots, t) \mid A \otimes B \mid A \wp B \mid A \multimap B \mid A^\perp \quad (3)$$

Sequents, [20] the essential elements of the **linear deductive system** [21], can be described as expressions in the following form:

$$\Gamma \vdash \Delta \quad (4)$$

where $\Gamma = (A_1, \dots, A_n), \Delta = (B_1, \dots, B_m), m, n \in \mathbb{N}_0$, represents the end sequence of predicate linear logic formulas.

Sequent $\Gamma \vdash \Delta$ means that the sequence of formulas Γ , called antecedent, consists of a set of assumptions, from which the sequence of formulas Δ , called succedent,

is derivable. If we imagine the set Γ as an intensional conjunction of assumptions $A_1 \otimes \dots \otimes A_n$ and consider the set Δ to be the extensional disjunction of conclusions $B_1 \oplus \dots \oplus B_m$, a sequence entry has the following form:

$$A_1 \otimes \dots \otimes A_n \vdash B_1 \oplus \dots \oplus B_m \quad (5)$$

It means that if all sequence formulas on the left side are applicable, at least one formula from the right side has to be applicable.

For the purpose of this article, only a small part of the PLL sequent calculus has to be defined here. It contains the following rules:

$$\begin{array}{c} \frac{}{A \vdash A} (id) \quad \frac{\Gamma \vdash A, \Delta \quad \Sigma \vdash B, \Pi}{\Gamma, \Sigma \vdash A \otimes B, \Delta, \Pi} (\otimes_r) \quad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \multimap B, \Delta} (\multimap_r) \\ \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash A^\perp, \Delta} ((.)^\perp_r) \quad \frac{\Gamma \vdash A[t/x], \Delta}{\Gamma \vdash (\exists x)A, \Delta} (\exists_r) \quad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash (\forall x)A, \Delta} (\forall_r) \end{array} \quad (6)$$

4 Ludics Theory

Ludics [22] [23] is a PLL extension, while it includes a space-time substantiating calculus using Gentzen-style sequences. The author of this theory is Jean-Ives Girard, who called it Locus Solum [24]. The basic principle of this calculus is handling positions of linear logic formulas, while ignoring their contents.

In ludics, **time** is the change in polarity of the individual units (called clusters) within the proof tree. Polarization can be explained as categorization of logical connections to positive and negative linear logic connections. Logical connections, linear logic constants, exponentials and quantifiers can be classified as follows:

- Positive – $\otimes, \oplus, \mathbf{1}, \mathbf{0}, \exists, !$
- Negative – $\&, \wp, \top, \perp, \forall, ?$
- Special - $\multimap$ represents dependent polarity, $(.)^\perp$ causes the flipping of polarity

The properties of focalization and invertibility are used in ludics. Focalization allows closing several consecutive instances of proof, incurred by applying deriving rules that establish a positive intensional conjunction or existential quantifier as an instance of proof. We can call this instance of proof, a cluster of positive formula values. Invertibility allows closing several consecutive instances of proof indicating a negative intensional disjunction or universal quantifier as one instance of proof. We can call this instance of proof a cluster of negative formula values. An instance of the formula within the proof tree therefore represents an alternation of positive and negative clusters. Changing the polarity in a proof

instance – from positive to negative or from negative to positive – is an incrementation of logical time.

In ludics, **space** represents linear formulas as arguments without the cut rule and all the logical information. In ludics, neither the truth nor the content of a formula are essential in the proof tree. The only important factor is its location, known as ξ . The proof tree that contains only location data (it does not work with the formulas but with addresses) is a design. Immediate subformulas of the A formula are enumerable, while the number of immediate subformulas can be labelled as $B_i, B_{ij}, B_{ijk}, \dots$, where i, j, k are positive integers - biases and i, ij, ijk are concatenations of particulate biases i, j, k . Then the address (also called locus) is the final sequence of the biases. If all the data, except the formula addresses, are removed in the sequences used in argument, we get a pitchfork, in the following form:

$$\xi \vdash A \quad (7)$$

where ξ represents a single address (i.e. a locus, which can also be empty) and context Λ is a finite set of addresses. Locus ξ and context Λ are pairwise disjoint.

In **pitchfork calculus**, the following rules apply:

The daimon axiom:

$$\frac{}{\vdash \Lambda} (\star) \quad (8)$$

Positive and negative rules:

$$\frac{\dots, \xi * i \vdash \Lambda_i \dots}{\vdash \Lambda, \xi} (+, \vdash, \xi, I) \quad \frac{\dots \vdash \Lambda_i, \xi * I \dots}{\xi \vdash \Lambda} (-, \xi \vdash N) \quad (9)$$

5 Application of Ludics on Natural Language Sentences

The solution discussed herein consists of two parts – a linguistic one and a logical follow-up to it. Its result will be the creation of a space-time characteristics of the specific natural language sentence, analysed by the Montague grammar.

5.1 Linguistic Section - Translation of a Sentence into Formal Language

We decided to work with the English sentence "Every hero seeks a princess but some may find a dragon.", on which it is possible to demonstrate how Montague handles an intentional context. The sentence consists of lexical units from the following categories:

- *Seek* and *find* are from the category of transitive verbs TV

- *Hero*, *princess*, and *dragon* are from the category of common nouns CN^1

This sentence is correctly formed according to [15], so it has a correct syntax. It has multiple meanings, but it will be sufficient to choose the "de dicto" form, for which there will be a single syntax tree (Fig. 2) and a single derivation tree to illustrate the translation into the language of logic (Fig. 3).

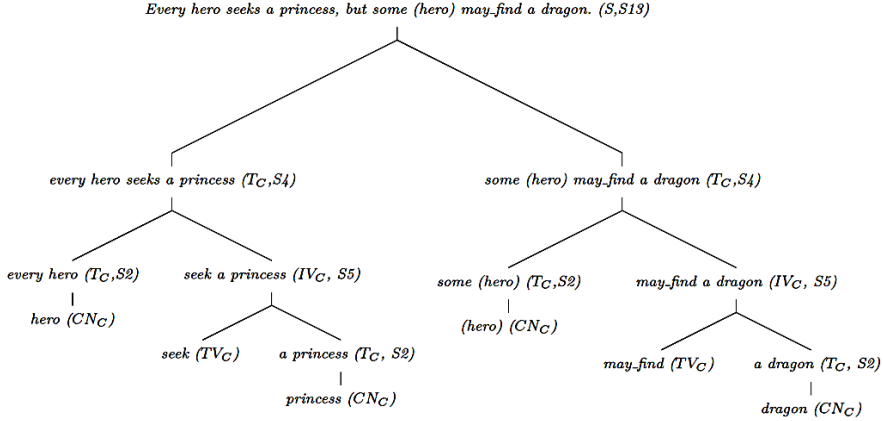


Figure 2

Syntactic derivation of a sentence

The syntactic rules used in the syntactic derivation are the following ($\times$ represents concatenation):

- $S_2: a/an, some/every \times CN \rightarrow T$
- $S_4: T \times IV \rightarrow t$, while the IV verb is replaced by its form in 3rd person singular
- $S_5: TV \times T \rightarrow IV$
- $S_{13}: T \times T \rightarrow T$

Semantic rules are used in the translation (in which the symbol $\mapsto$ represents "translate into", θ, ϑ are the translated expressions and θ', ϑ' are the results of the translation in logical language):

- $T_1: \theta \mapsto \theta'$, where θ' represents the translation of lexical units into logical language
- $T_{2\text{-}every}: \theta \mapsto \lambda P \forall u(\theta'(u) \multimap P(u))$
- $T_{2\text{-}some}: \theta \mapsto \lambda P \exists u(\theta'(u) \otimes P(u))$
- $T_4, T_5, T_{13}: (\theta, \vartheta) \mapsto \theta'(\wedge \vartheta')$

¹ In the syntax tree, the category of terms T and the category of intransitive verbs IV are used.

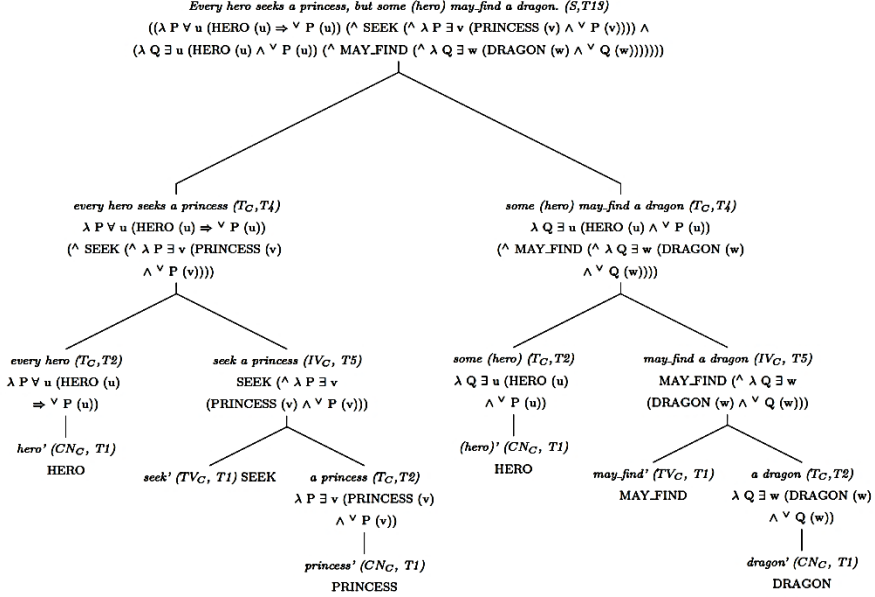


Figure 3

Translation of the sentence into the language of intensional logic

It is obvious that the logical expression that arose after the translation of the sentence into the language of intensional logic is too complicated and further work with it would be cumbersome. We can simplify the translated expression and modify it to make it more acceptable and suitable for further processing. The resulting logical representation of the aforementioned sentence is:

$$(\forall u(H(u) \rightarrow \exists v(P(v) \wedge S(u, v)))) \wedge (\exists u(H(u) \wedge \exists w(D(w) \wedge F(u, w)))) \quad (10)$$

The formula – the result of applying the original Montague principles – is a formula of FOPL that is not compatible with ludics. Therefore, we used the symbols of PLL by using its intensional fragment:

$$(\forall u(H(u) \multimap \exists v(P(v) \otimes S(u, v)))) \otimes (\exists u(H(u) \otimes \exists w(D(w) \otimes F(u, w)))) \text{ instead of } \\ (\forall u(H(u) \rightarrow \exists v(P(v) \wedge S(u, v)))) \wedge (\exists u(H(u) \wedge \exists w(D(w) \wedge F(u, w)))) \quad (11)$$

5.2 Logical Section - Translation of a Sentence into Formal Language

Fig. 4 shows a proof of formula $(\forall u(H(u) \multimap \exists v(P(v) \otimes S(u, v)))) \otimes (\exists u(H(u) \otimes \exists w(D(w) \otimes F(u, w))))$ in Gentzen sequent calculus of PLL. The following contexts are used in the proof:

$$\begin{aligned}
 \Gamma &= \{(H(u))^\perp, (P(v))^\perp, (S(u, v))^\perp, (H(u))^\perp \\
 &\quad (D(w))^\perp, (F(u, w))^\perp\} \\
 \Sigma &= \{(H(u))^\perp, (D(w))^\perp, (F(u, w))^\perp\} \\
 \Delta &= \{(H(u))^\perp, (P(v))^\perp, (S(u, v))^\perp\} \\
 \Theta &= \{(H(v))^\perp, (S(u, v))^\perp\} \\
 \Lambda &= \{(H(w))^\perp, (F(u, w))^\perp\}
 \end{aligned} \tag{12}$$

When applying ludics, first, the spots in the proof tree, where the polarity changes from positive to negative and vice versa are marked; thanks to that, a polarized tree emerges from the derivation tree. Red + and - signs in parentheses on the left of the line separating the proof tree steps (instantiations) indicate the polarity of the formula.

Next, we simplify the polarized tree by clustering formulas with the same polarity to form a reduced tree showing sequential time incrementation (Fig. 5).

The reduced tree uses the following:

- Substitution of formulas and subformulas:

$$\begin{aligned}
 H &= (\forall u(H(u) \multimap \exists v(P(v) \otimes S(u, v)))) \otimes (\exists u(H(u) \otimes \\
 &\quad \exists w(D(w) \otimes F(u, w)))) \\
 B &= P(v) \otimes S(u, v)
 \end{aligned} \tag{13}$$

- Contexts:

$$\begin{aligned}
 \Gamma &= \{(H(u))^\perp, (P(v))^\perp, (S(u, v))^\perp & \Gamma_4 &= \{(D(u))\} \\
 &\quad (H(u))^\perp, (D(w))^\perp, (F(u, w))^\perp\} & \Gamma_5 &= \{(F(u, w))\} \\
 \Gamma_1 &= \{(H(u))^\perp\} & \Gamma_6 &= \{(D(u))\} \\
 \Gamma_2 &= \{(P(v))^\perp, (S(u, v))^\perp\} & \Gamma_7 &= \{(D(u))\} \\
 \Gamma_3 &= \{(H(u))\} & \Gamma_8 &= \{(S(u, v))\}
 \end{aligned} \tag{14}$$

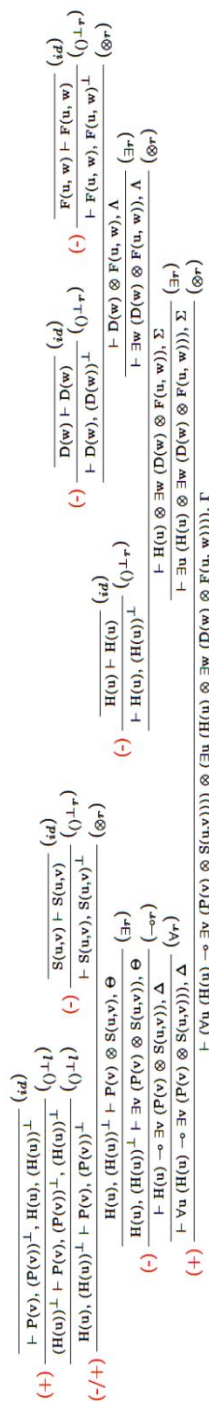


Figure 4
Prof tree of the predicate linear formula

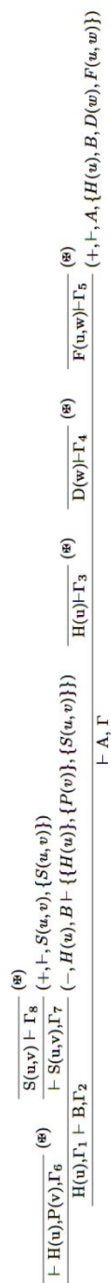


Figure 5
Reduced tree showing incrementation of logic time



Figure 6
Address tree showing logic space

The address tree represents the logical space, thanks to assigning addresses to formulas and subformulas (Fig. 6). Contexts (denoted by Δ) in Fig. 6 are substituted by appropriate loci (ξ):

$$\begin{array}{lll} \Delta = \xi & \Delta_3 = \xi_3 & \Delta_{11} = \xi_{11} \\ \Delta_1 = \xi_1 & \Delta_4 = \xi_4 & \Delta_{21} = \xi_{21} \\ \Delta_2 = \xi_2 & \Delta_5 = \xi_5 & \Delta_{211} = \xi_{211} \end{array} \quad (15)$$

Conclusions

The contribution of this article is a description of a nontraditional connection between the linguistic domain (the meaning of natural language sentences by MG) and the logical domain (ludics theory, which allows natural language sentences to be placed in logical space and time). PLL is used to describe limited resource issues, and usually not as a tool of LANL. Revealing suitable properties of PLL (intensional character, high expressiveness due to operating with two conjunctions and two disjunctions, constants not only for truth/false but also sense/nonsense, description of changing states of the world together with the possibility to capture consumption of resources and others) allows it to be used in this area. The presented linguistic-logic connection is the starting point for our research, in which we try to move from the abstract syntax of the language (the level of MG trees) through the semantic interpretation (the level of MG formulae) to interactions in dialogues (the level of Ludics design).

The future remains open, as there are several possibilities, as to where one can proceed with the knowledge gained in this work. Working with the Montague grammar, the Authors wish to extend the original fragment of English language PTQ in Montague's theory or to create a similar custom fragment of Slovak language (for which there would be custom categories, types and corresponding syntactic and semantic rules). Within the complex logical analysis of natural language, it is possible to compare Montague and Tichý's approaches, based on the fact that even if Tichý's transparent intensional logic is often overlooked, it is considered to be a more progressive one (due to the fact that Tichý considered the set of possible worlds as a grammatical category and he introduced the two-sorted type theory). Artificial intelligence may also be used with computational linguistics, which may be the future of automated natural language processing. It is also possible to modify PLL and MG into a single complex logical system, which would allow manipulation of natural language sentences directly, without the need for two-step processing. Focusing on ludics, the natural extension of this work would be modelling natural language dialogues, by using interactions in the form of players' turns in gaming spaces. In future research, we would like to investigate the possibility of designing a solution based on the outcomes of the research published herein, using specialized hardware based on data flow computation control, including operating memory hardware [25], where we will try to include granularity [26] in the design of a security automated solution [27].

Acknowledgement

This work has been supported by the following projects:

- Project KEGA 011TUKE-4/2020: "A development of the new semantic technologies in educating of young IT experts".
- Project of the Faculty of Electrical Engineering and Informatics at the Technical University of Košice under contract No. FEI-2020-70: "Behavioral model of component systems based on coalgebras and linear logic."

This support is very gratefully acknowledged.

References

- [1] M. Werning, W. Hinzen, and E. Machery, eds., "The Oxford Handbook of Compositionality", Oxford Handbooks in Linguistics. Oxford: Oxford University Press, 2012, p. 560
- [2] T. Degani, N. Tokowicz, "Semantic ambiguity within and across languages: An integrative review", *The Quarterly Journal of Experimental Psychology*, Vol. 63, No. 7, 2010, pp. 1266-1303
- [3] B. Partee, "Montague's "Linguistic" Word: Motivations, Trajectory, Attitudes", *Proceedings of Sinn Und Bedeutung*, Vol. 17, 2013, pp. 427-453
- [4] M. Stokhof, The development of Montague grammar, In: "History of the Language Sciences", eds. S. Auroux, E. F. K. Koerner, H.-J. Niederehe, and K. Versteegh, Berlin -New York: Walter de Gruyter, Vol. 3, 2006, pp. 2058-2073
- [5] T. M. V. Janssen, "Montague semantics", *The Stanford Encyclopedia of Philosophy* (Winter 2011 Edition), ed. E. N. Zalta. Stanford: Stanford University, 2011
- [6] M. Duží, B. Jaspersen, P. Materna, "Procedural Semantics for Hyperintensional Logic. Foundations and Applications of Transparent Intensional Logic", 1nd ed. Netherlands: Springer, 2010, p. 550
- [7] B. Jaspersen, "Structured lexical concepts, property modifiers, and Transparent Intensional Logic", *Philos Stud*, Vol. 172, 2015, pp. 321-345
- [8] M. Duží, B. Jaspersen, "In Memory of Pavel Tichý", *Organon F*, Vol. 172, Vol. 4, 2014, pp. 558-566
- [9] L. Dixon, A. Smaill, T. Tsang, Tracy, "Plans, Actions and Dialogues Using Linear Logic", *Journal of Logic, Language and Information*, Vol. 18, 2009, pp. 251-289

- [10] E. Demeterová, D. Mihályi, V. Novitzká, "A categorical model of predicate linear logic," *Journal of Applied Mathematics and Computational Mechanics*, Vol. 14, 2015, pp. 27-42
- [11] A. Lecomte, B. Troncon, "Ludics, Dialogue and Interaction", PRELUDE Project 2006-2009. Berlin: Springer, 2011, p. 221
- [12] M. Hamano, P. Scott, "On geometry of interaction for polarized linear logic," *Mathematical Structures in Computer Science*, Vol. 28, No. 10, 2018, pp. 1639-1694
- [13] P. B. Andrews, "An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof," 2nd ed., Berlin: Kluwer Academic Publishers, 2002, p. 304
- [14] S. Abramsky, "Computational interpretations of linear logic", *Theoretical Computer Science*, Vol. 111, No. 1-2, 1993, pp. 3-57
- [15] R. Montague, "English as a formal language", *Linguaggi nella Società e nella Tecnica*, Milan: Edizioni di Comunità, 1970, pp. 189-223
- [16] R. Montague, R., "The proper treatment of quantification in ordinary english", *Approaches to Natural Language (Synthese Library 49)*, 1973, pp. 221-242
- [17] R. Montague, "Universal grammar", *Theoria*, Vol. 30, 1970, pp. 373-398
- [18] L. Vokorokos, A. Pekár, P. Fecilák, IPFIX Mediation Framework of the SLAmeter Tool, *International Conference on Emerging eLearning Technologies and Applications*, 2013, pp. 311-314
- [19] V. Novitzká, D. Mihályi, "What about linear logic in computer science?", *Acta Polytechnica Hungarica*, Vol. 10, No. 4, 2013, pp. 147-160
- [20] S. Grys, W. Minkina, L. Vokorokos, "Automated characterisation of subsurface defects by active IR thermographic testing - Discussion of step heating duration and defect depth determination", *Infrared Physics & Technology*, Vol. 68, 2014, pp. 84-91
- [21] T. Brauner, "Introduction to linear logic", *BRICS - Basic Research in Computer Science*, 1996, p. 66
- [22] P. L. Curien, "Introduction to linear logic and ludics, Part I", *Advances in Mathematics (China)*, Vol. 34, No. 5, 2005, pp. 513-544
- [23] A. Lecomte, "Meaning, Logics and Ludics", *Imperial College Press*, 2011, p. 388
- [24] J. Y. Girard, "Locus Solum: From the rules of logic to the logic of rules", *Journal Mathematical Structures in Computer Science*, Vol. 11, No. 3, 2001, pp. 301-506

- [25] L. Vokorokos, B. Madoš, N. Ádám, A. Baláž, “Innovative operating memory architecture for computers using the data driven computation model”, *Acta Polytechnica Hungarica*, Vol. 10, No. 5, 2010, pp. 63-79
- [26] J. Juhár, L. Vokorokos, “Separation of Concerns and Concern Granularity in Source Code”, 2015 IEEE 13th International Scientific Conference on Informatics, 2015, 139-144
- [27] L. Vokorokos, A. Baláž, and B. Madoš, “Application Security through Sandbox Virtualization”, *Acta Polytechnica Hungarica*, Vol. 12, No. 1, 2015, pp. 83-101