

THESIS ON INFORMATICS AND SYSTEM ENGINEERING C51

Discovering Logical Constructs from Estonian Children Language

ERIKA MATSAK

TUT
PRESS

Faculty of Information Technology
Department of Computer Engineering
Chair of Digital Systems Design
TALLINN UNIVERSITY OF TECHNOLOGY

Dissertation was accepted for the defense of the degree of Doctor of Philosophy in Engineering on November 24, 2009.

Supervisor: Prof. Peeter Lorents. Research and Development Branch Chief, Cooperative Cyber Defence Centre of Excellence

Dots. Margus Kruus. Department of Computer Engineering, Tallinn University of Technology

Opponents: Dr. Gabriel Jakobson. Altusys Corporation, USA

Dr. Risto Vaarandi. Department of IT Security Analysis, SEB Estonia

Defense of the thesis: January 8, 2010

Declaration

Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology has not been submitted for any academic degree or examination at any other university.

Copyright: Erika Matsak, 2009

ISSN 1406-4731

ISBN 978-9985-59-961-7

INFORMAATIKA JA SÜSTEEMITEHNIKA C51

Loogiliste konstruktsioonide avastamine eesti laste keelest

ERIKA MATSAK

To my daughter Julia

FOREWORD

This thesis provides an overview about Erika Matsak's doctoral research, defining the field of study and the problems researched, explaining methods used to solve these problems and describing the results of the work. The field of study is narrowed down to the capability of intelligent systems to arrive to logically founded (correct!) conclusions based on knowledge that is present in natural language texts. In case of intelligent IT systems this requires the capability to transform the arguments in natural language texts into logic formulas, as well as the capability to extract logical inference steps from natural language reasoning. According to well known results from algorithm theory and mathematical logic (Church, 1936) it can be assumed that "fully automatic tools" cannot be relied upon. Therefore, a dialogue system was developed to solve the problem.

The core of this doctoral thesis is the development of a prototype dialogue system DST and its application to identify logical constructs used by children, who by definition are (natural) self-evolving intelligent systems. Identifying these constructs (including logic operations, formulas, inference rules and inferences) enables us to map the use of logical instruments by children of various ages. This, in turn, supports researchers who wish to study the development and perfecting mechanisms of logical instrument complexes in self-evolving intelligent systems.

The thesis consists of the introduction, conclusion and chapters providing an overview of:

- The nature of the field of study and related work
- The DST (Dialog System for Transforming Natural Language Texts) dialogue system and its improved version, as well as using DST for identifying logic constructs in natural language texts
- Results from implementing the DST dialogue system and its improved version to transform texts from children of various age groups, including the surprisingly diverse arsenal of logic instruments identified.

EESSÕNA

Käesolevas töös antakse ülevaade Erika Matsaki doktoritöö sooritamise vältel uuritud valdkonnast, selles käsitletud probleemidest, nende lahendamiseks kasutatud meetoditest ning saadud tulemustest. Nimetatud doktoritöö uurimisvaldkond seostub intelligentsete süsteemide suutlikkusega jõuda loogiliselt põhjendatud (õigete!) otsusteni lähtuvalt neist teadmistest, mida esitatakse loomuliku keele tekstide abil. Intelligentsete IT-süsteemide korral eeldab see võimekust transformeerida loomuliku keele tekstides esitatud väited loogikavalemiteks. Ja samuti veel võimekust ekstraheerida loomuliku keele tekstides esitatud põhjendustest loogilisi tuletussamme. Algoritmiteooria ning matemaatilise loogika hästi tuntud tulemustele (vt nt Church 1936) tuginedes võib eeldada, et nii valemite kui ka tuletussammude välja eraldamisel pole võimalik tugineda „täisautomaatsetele vahenditele“. Seetõttu tuleb paratamatult piirduda vastava dialoogsüsteemiga.

Antud doktoritöö tuumaks ongi sobiva dialoogsüsteemi DST prototüübi loomine ning selle rakendamine laste kui (looduslike) isearenevate intelligentsete süsteemide poolt kasutatavate loogiliste konstruktsioonide esiletoomiseks. Nimetatud konstruktsioonide (sh loogilised operatsioonid, valemid, tuletusreeglid ja tuletused) esiletoomine võimaldab luua ülevaate sellest, millistel arenguetappidel, milliseid loogilisi vahendeid (juba) omatakse ning rakendatakse. See omakorda toetab uurijaid, kes soovivad selgusele jõuda isearenevates intelligentsetes süsteemides vajalike loogiliste vahendite komplekside kujunemise ja täiustumise mehhanismides.

Käesolev doktoritöö koosneb sissejuhatusest, kokkuvõttest ning osadest, mis annavad ülevaate

- antud uurimisvaldkonna loomusest ning sellega seonduvatest töödest
- dialoogsüsteemi DST ja selle edasiarenduste ülesehitusest ning selle rakendamise loomuliku keele tekstides sisalduvate loogiliste konstruktsioonide väljatoomiseks
- dialoogsüsteemi DST ja selle edasiarenduste rakendamisel saadud tulemustest mitmetest vanuserühmadest pärit laste loodud tekstide transformeerimisel esile tulnud ootamatult mitmekesistest loogilistest instrumentidest.








ACKNOWLEDGEMENTS

The author would like to express her gratitude towards her teachers and advisors – CCD COE R&D Branch Chief and Chair of EBS IT Department, professor Peeter Lorents and Director of TUT Computer Engineering Institute, Dr Margus Kruus. In addition, the author is grateful for valuable advice and support from CCD COE Scientist and leading research scientist of TUT Cybernetics Institute, professor Enn Tõugu. For his assistance with the translation of this work, many thanks go to Rain Ottis from CCD COE.

TABLE OF CONTENTS

1. Introduction.....	1
2. A system for mapping natural language into logic language.....	7
3. Transforming texts. Formula level.....	11
3.1. The various roles of the word “siis“ (then).....	12
3.2. Assessments.....	13
3.3. Modality complexes.....	15
3.4. Ownership.....	15
3.5. On hidden equivalence.....	16
4. Transforming texts. Inference level.....	17
4.1. Identifying constructs from texts using the DST dialogue system.....	19
4.2. Problems with algorithmic insolvability.....	19
4.3. Improved DST dialogue system. Formula level.....	20
4.3.1. Training the dialogue system.....	21
4.3.2. Automatic sentence transformation.....	22
4.3.3. Full text transformation.....	23
4.4. Improved DST dialogue system. Inference level.....	25
4.4.1. Customizing the Lorents’ method for searching inference rules	26
4.4.2. Inference step identification module.....	27
4.5. The general layout of the program.....	29
5. Logic constructs identified with DST from children’s texts.....	31
5.1. Formula level.....	32
5.2. Inference level.....	37
Summary.....	40
References.....	42
APPENDIX A: Algorithms.....	45
APPENDIX B: Curriculum Vitae.....	53
CURRICULUM VITAE in English.....	55
CURRICULUM VITAE Eesti keeles.....	58
Research papers.....	61

LIST OF PUBLICATIONS

-  Matsak E. (2005). Dialogue system for extracting Logic constructions in natural language texts. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2005. Volume II p. 791 – 797. Las Vegas, Nevada, USA
-  Matsak E. (2006). Using Natural Language Dialog System DST for Discovery of Logical Constructions of Children's Speech. The 2006 International Conference on Artificial Intelligence IC-AI 2006. Volume I p. 325 – 331. Las Vegas, Nevada, USA
-  Matsak E. (2006). System DST for Transforming Natural Language Texts, Representing Estimates and Higher Order Predicates and Functionals. The 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2006. Volume III p. 79 – 84. Orlando, Florida, USA
-  Matsak E. (2007). The prototype of system for discovering of inference rules. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2007. Volume II p. 489 – 492. Las Vegas, Nevada, USA
-  Matsak E. (2008). Improved version of the natural language dialog system DST and its application for discovery of logical constructions in children's speech. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2008. Volume II p. 332 – 338. Las Vegas, Nevada, USA
-  Matsak E. (2009). Representing logical inference steps with digital circuits. Lecture Notes in Computer Science: HCI International 2009. 5618 Springer Berlin, p. 178 – 184.
-  Matsak E. (2009). On the logic module in intelligent systems. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2009. Volume II p. 594 – 599. Las Vegas, Nevada, USA

1. INTRODUCTION

The ability to apply logic is an important capability of intelligent systems. This is important in order to get correct conclusions from correct descriptions of situations. Correct conclusions, in turn, are necessary to make correct decisions, which are often required in a very short amount of time. For example, in case of a cyber attack the "survival" of a system often depends on very quickly choosing and applying countermeasures. Therefore, the decision making process (drawing correct conclusions from available information) must be as quick as possible. This is only reasonable, if the following two capabilities are present among others:

- representing descriptions of situations and the resulting conclusions with language based constructs or arguments that are clearly defined and
- applying only clearly defined transformations from one set of arguments to the next.

Using the terminology of mathematical logic, the preceding can be formulated as having the capability to

- construct logic formulas and
- apply only those (regular) inference steps that belong to a set of defined inference rules.

This brings us to an important realization: *in order to qualitatively assess the ability of an intelligent system to make correct decisions based on correct inferences of correctly described situations, one must have an overview of the system's capability to construct logic formulas and to apply inference steps.*

It is important, however, not to ignore the fact that humans and the technical systems created by humans use many *different* logic systems. For example, lawyers use the classical bi-valued logic that uses two truth values. In order to describe some micro-level physics phenomena J. von Neuman and G. Birkhoff proposed a tri-valent logic system that uses three truth values (Birkhoff, von Neumann, 1936). The intuitionistic logic system (Mints, Tyugu, 1982) is useful when handling computerised synthesis of programs using E. Tōugu's structural synthesis tools (Tyugu, 1988). It is known that the scale of truth values in intuitive logic is infinite (Dragalin, 1979).

Another important aspect of note is that the exact scientific form of various logic systems (which is a prerequisite for creating reliable technical applications) comes from two closely related sources:

- the area of study that requires the logic system (for example, quantum mechanics)
- the instruments that humans use to describe this area of study and to infer new knowledge about it (for example, structural synthesis formulas and rules).

Unfortunately, in most cases only the completed system of instruments is “visible”. How this system was developed typically remains hidden. One of the notable exceptions here is the work of G. Gentzen, especially his 1936 study on non-contradiction in arithmetic (Gentzen, 1936). In this study he analyzes the Euclidean theorem on the infinite amount of prime numbers. By replacing the textual parts in the theorem and the proof with logic symbols and logic formulas that they form, Gentzen creates a set of formulas that represent the theorem.

For example, the argument “z is a prime number that is greater than a” is replaced with the formula $\text{Prim } z \ \& \ z > a$.

Next, Gentzen explains how and why some arguments can be associated to others in a concrete fashion. This brings Gentzen to inference rules or relationships between formulas. An inference rule allows “well-founded” arguments to form a foundation, or a predicate, for the logically following argument that must therefore also be “well-founded”.

For example, an arbitrary formula $F(z)$, where z is a symbol representing some object, is associated to the formula $(\exists z)F(z)$. In such cases specific formulas are associated with a “general association.” More specifically, the formula $\text{Prim } z \ \& \ z > a$ is associated with the formula $(\exists z)(\text{Prim } z \ \& \ z > a)$, because the argument “there exists an z , where z is a prime number and z is greater than a ” follows from the already known argument that for some z “ z is a prime number and z is greater than a ”. Based on this fact, all formulas of the type $F(z)$ can be associated with the formula $(\exists z)F(z)$.

At the same time, Gentzen’s work does not explain how the text in a natural language is transformed into a formula. In other words, what are the steps and how should they be implemented in order to transform the text “there exists such an z that is a prime number and that is greater than a ” into the formula $(\exists z)(\text{Prim } z \ \& \ z > a)$.

$> a$)? The process of forming “general associations” or “general rules” from a set of specific formulas is also unclear. For example, how to form a general rule about getting one formula (Prim $z \ \& \ z > a$) from some other formulas (formula Prim z and formula $z > a$)?

Transforming text into formal language and understanding the text became important in the middle of the 20th century. This was mostly due to the creation of programs that were meant to answer questions posed by a human. The first program of this type was able to answer baseball-related questions (Green, Wolf, Chomsky, Laughery, 1961). The program consisted of two parts. The *linguistic part* read questions from punched cards, performed syntax analysis and ascertained information about the data in question. The *execution part* was tasked with selecting the necessary information and presenting the answers. Syntax analysis was used to identify verbs, nouns, preposition groups and adverbs. The result was not in the form of a logic (predicate calculus) formula, but information mirroring the grammatical structure of texts, which was later used to access the dictionary, add attributes to word types, and finally to find suitable answers.

A similar approach was used in the program “Student”, which could solve textual assignments based on elementary algebra (Bobrow, 1964).

At this point we should mention some Prolog-related works, specifically, the formalism created by Pereira and Warren. This formalism allowed constructing a grammatical tree of the sentences in a text. On the other hand, it allowed generating corresponding texts based on predicate calculus formulas (Pereira, Warre, 1980). With the help of the grammatical tree it was possible to identify information sets that are linked not only to individuals (concepts), but also to modalities, time, location, goals, reasons, results etc. Once again, it should be noted that (natural) texts were not transformed into logic (predicate calculus) formulas. However, texts in use conformed to quite strict restrictions, which matched the formalism (Osugi, Saeki, 1990).

Various tools have been created to operate with mathematical texts, including ones that applied certain algebraic constructs (for example see Atayan, 1986). The goal of these works was to develop and implement a formalism that relied on the „subordination relation” of some concepts (or a situation, where some concepts could be viewed as „subordinated” to other concepts). Describing and solving tasks was based on concepts and relations between concepts. Semantic webs were created to present situations for arguments from a specific text. A set $\{U_i, S_i\}$ of textual elements was created for an argument U_i that was present in a text S_i . The

relations between parts of texts were studied in situations where one text is a part of the other. The formal language used was actually quite close to natural language. In this language, only fixed types of arguments were used to form the language products with specific templates. In the work process the lexical parts were identified and „wrapped” into the necessary format. Syntax analysis was also used to form the necessary grammatical tree. However, like in previous examples, (natural language) texts were not transformed into logic (predicate calculus) formulas.

In order to identify logic constructs that are present in natural language texts we rely on the natural language text transformation procedure (Lorents, 2000) and the predicate mining procedure (see Lorents 1993, 2002) developed by Lorents.

Lorents’ procedures are mainly used in the context of this work in order to observe step by step how the formulas and the necessary rules for deriving formulas are created. This, in turn, allows us to understand how *reliable* (and usable, in a controlled manner) instruments are formed. These instruments can be applied in intelligent systems to describe situations and to generate correct decisions based on this description.

There are several other approaches to describing situations and making correct decisions. Keyword here is *reasoning*, which is well covered by an article of L. Chittaro and A. Montanari in 2000 (Chittaro, Montanari, 2000). To some extent, many of the well-known reasoning instruments can be handled as fragments of predicate calculus. Depending on the nature of the area of interest, these fragments use only language that is “essential” and has very specific meaning (including a relatively limited set of symbols and formulas). The same is also more or less true for the selection of axioms. It is important to note that in addition to the theoretical approaches to this problem, there are also several practical software applications in use in the control systems of robots and the production process (Levesque, Reiter, Lesperance, Lin, Scherl, 1997).

In addition to the “relatives of predicate calculus” a totally different approach is also available: algebraic systems. For example, Allen’s interval algebra IA, which is used in reasoning about temporal constraints (Allen, 1983).

A separate approach to note is the use of various graphs and graph-like constructs to describe situations and reasoning methods (Golombik, Shamir, 1993). Best known in this field are conceptual graphs and semantic networks, as well as neural networks. These approaches can be used to perform the relevant

calculations. Tōgu's computational models, designed to describe the structural synthesis of programs, are a good example. These models are basically bi-partite graphs where the first type of nodes represents relations, the second type represent variables and edges associate relations with variables. Over time, various (logic) calculations with their specific alphabets, formulas and inference rules have been developed to handle the structural synthesis of programs (Tyugu, 1970, 1972, 1988, 2007).

There is no question that many interesting approaches have been developed to research and create one of the most important capabilities of an intelligent system: to describe situations and make correct decisions. Unfortunately, the detailed description of this field would require substantial work that falls outside the limits of this study.

In the following paragraphs we will define the area of study and the problems that will be examined.

Let us start with the above mentioned fact that many of the instruments for describing situations and making decisions are “copied” from the human intellect. A substantial part of these instruments can be related to some form of logic. However, it has not been possible to “acquire”, study and apply the mechanisms that create or shape these capabilities. A special case - how human intellect works, is not well known at present. Does it evolve (or is it acquired) smoothly over time in small steps or are there significant jumps in capability? At what stage of the human development do various logic instruments appear (including logic operations for creating non-atomic formulas and inference steps or rules for constructing proofs)?

One way to *start* answering the questions above is to study the texts collected at fixed times in human development. Or more precisely:

- acquire texts that seem to contain logic constructs and for which the age of the author (at the time of recording) is known
- transform the texts in order to identify the logic constructs present in the text
- by relating the age of the text creators with logic constructs present in the text, we can explain, when various logic constructs emerge in the development of human intellect.

Knowing when various logic constructs emerge will enable much more detailed research in the field of human intellect development, to the level where it becomes possible to model the creation mechanisms of logic instruments and capabilities in humans. This allows self-learning and self-improving intelligent systems to be created and an *appropriate* logic system applied, instead of applying logic befitting to describe quantum effects to the field of automatic synthesis of new programs.

2. A SYSTEM FOR MAPPING NATURAL LANGUAGE INTO LOGIC LANGUAGE

In this chapter we consider a specific mapping system that allows the transformation of arguments from a natural language to logic formulas. The system is based on author's research on Estonian language text transformation, which directly influenced the choice of symbols and their meanings as described below. The texts in question are transcripts from the conversations of children of various age groups that were taped using a dictaphone. The transformation of these texts identified the need for the following symbols (Matsak, 2005):

- Individual symbols to represent (related or unrelated) individual objects. Let us use the symbols x_1, \dots, x_n and q_1, \dots, q_n
- Individual symbols to represent time. Let us use the symbols t_1, \dots, t_n
- Individual symbols to represent the value of assessments. Let us use the symbols $\gamma_1, \dots, \gamma_n$ and τ_1, \dots, τ_n
- Individual symbols to represent natural numbers
- Predicate symbols to represent first order predicates. Let us use the symbols P_1, \dots, P_n and A_1, \dots, A_n
- Second order predicate symbol “ \vDash ” to represent the correctness of formulas
- Second order functional symbol “Val” to assess predicates
- Logic operation symbols: “ \neg ” – negation, “ $\&$ ” – conjunction, “ \vee ” – disjunction, “ \supset ” – implication, “ \Leftrightarrow ” equivalence
- Quantifiers: “ \forall ” – universal quantifier, “ \exists ” – existential quantifier
- Modalities: “ \diamond ” – maybe and “ \square ” – definitely
- Various useful symbols for describing order etc. (brackets, comma, semi-colon etc.)

Note. In principle, the functional symbol “Val” can be replaced with as many second order predicate constants as we have symbols for assessment values. For example, if the assessments are “somewhat” and “very” and the first order single element predicate is “tall”, then instead of “Val(tall(boy))=somewhat” and “Val(tall(boy))=very” we could write “somewhat(tall(boy))” and “very(tall(boy))”.

Table 1. Logical roles with semantic sets

Type	Examples	Symbol(s)
1. Negation	Ei, pole, vale [no, not, false]	\neg
2. Conjunction	Ja, ning, ka, samuti [and, also, too, as well]	$\&$
3. Disjunction	Või, ehk [or]	\vee
4. Implication	Siis, seega, järelikult [then, thus, therefore, consequently]	\supset
5. Equivalence	Sama, samaväärne, ekvivalentne, samalaadne, ühesugune, seesama, toosama [same as, equal, equivalent, identical, the same]	\Leftrightarrow
6. Universal quantifier	Kõik, kogu, terve, igäüks, iga, igamees, viimseni, igäüks [all, whole, total, entire, every]	\forall
7. Existential quantifier	Leidub, on olemas, juhtub, sattub, on, esineb, eksisteerib, ette tuleb [there is, there exists, it happens, is]	\exists
8. Modality	Kindlasti, kahtlemata, raudselt ilmtingimata, tingimata, igatahes, surmkindlalt [for sure, absolutely, definitely, inevitably, by all means]	\square
	Võib-olla [maybe, perhaps, possibly]	\diamond
9. Belonging to a set	Minu, ema oma (sõnad semantikaga kuuluvus või omandamine) [mine, mother's (words with the semantics of ownership or belonging)]	\in
10. Time symbols	Eile, täna, homme, hiljem, pärast, enne, praegu jne. [yesterday, today, tomorrow, later, afterwards, before, now, etc.]	t_1, t_2, t_i, t_{i+n}
11. Assessments on the individual, predicate or amount of time	Natuke, liiga, palju, hästi palju [a little, too much/many, many, very much/many]	$\text{Val}(x)=\gamma,$ $\text{Val}(P)=\gamma,$ $\text{Val}(t)=\gamma,$
12. Assessments on the time interval	Nüüd, kaua, kohe, asj [at present, long time, now, at the moment]	$\text{Val}(t)=\tau$
13. Correctness	Ongi [it is so]	\vDash
14. Complexes of modality	Võib teha midagi, Saab teha midagi [may/can do something]	$\vDash [\diamond P(x1, x2)]$ $\vDash [\square \diamond P(x1, x2, t)]$

In the following table we present the mapping between the symbols described and the appropriate phrases from Estonian language. The mapping draws upon the

work of D. Lorents (Lorents D., 1992). This work defines the necessary foundation for identifying predicates and individual symbols in the grammatical form of a word, as well as maps the classical logic operators and quantifiers with their equivalents in Estonian language. From here we can continue to formulate new rules and further narrow down the roles of logic operators and quantifiers. This requires the use of opposites (by meaning) of a word and the base word forms (as the various language cases must not interfere with establishing the meaning of the word). However, in case of natural languages we must also deal with elements of non-classical logic. Therefore, we will use the results of author's prior research (Matsak, 2004) to decide what roles are definitely necessary in order to transform the sentences of a "young" human (6 year old child). These roles are assigned to semantic sets that will best represent all the synonyms within a group (Table1):

Table 2. Logical roles detectable by morphological roles

Predicates	Morphological symbols	Individual symbols	Morphological symbols
Adjective	<u>A</u>	Noun	<u>S</u>
Verb	<u>V</u>	Pronoun	<u>P</u>
Superlative	<u>U</u>	Real name	<u>H</u>
<i>Ole</i> (be) forms + adjective	ole+ ... <u>A</u>	Adverb	<u>D</u>
<i>Ole</i> (be) + superlative	ole+ ... <u>U</u>		
<i>Ole</i> (be) + verb, or many verbs, in case of having the same role of a predicate	ole+ ... <u>V</u>		
<i>Ole</i> (be) + noun in nominative case	ole+... <u>S</u> sg n ole+... <u>S</u> pl n		

Note. During the course of the work it became clear that a separate set needs to be formed from words (and the corresponding morphological roles) that do not have a logical role.

In order to determine a match for predicates we must specify the morphological forms and their syntactic roles (Table 2).

Having the *alphabet* of fixed symbols we can form *formulas*. Defining formulas begins as usual:

- first we assemble (atomic) formulas where terms (in our case only individual symbols) are in the place of predicates that have one or many elements

- new formulas are assembled by adding logic operations, modality and quantifier prefixes (while avoiding quantifier collisions) to the existing formulas
- by putting parentheses around an existing formula we (essentially) get the same formula

In this case we add another way of creating formulas:

- If P_i and A_i are predicate symbols while γ_i and τ_i are symbols for the (value of) assessments, then $\text{Val } W=\gamma_i$ and $\text{Val } W=\tau_i$ are also formulas
- If W is an existing formula, then $\text{Val } W$ and $\mathbb{F}W$ are also formulas.

Formulas correspond to arguments in natural language texts.

Finally we observe *inference rules*. In logic, an inference rule is some $k+1$ -element relation T between formulas. If formulas $A_0, A_1, \dots, A_k, A_{k+1}$ are in a relation T , then we have an inference step that is based on T . The formula A_{k+1} is in this case a direct result (conclusion) of applying the inference step. Formulas A_0, A_1, \dots, A_k are the direct prerequisites (premise).

The inference step corresponds to a part in natural language text, where some argument (corresponding to formula A_{k+1}) is “derived” or “reached” from other arguments (corresponding to formulas A_0, A_1, \dots, A_k).

This chapter presented the necessary basis to get from the grammatical (morphological) role of a natural language word to its logical role. This is essential for creating a system that can extract logic constructs from natural language texts, including logic operations, quantifiers, modalities, formulas, inference steps (inference rules) and derivations.

3. TRANSFORMING TEXTS. FORMULA LEVEL

In this chapter we review the theoretical aspects of the natural language text transformation procedure. We also handle various problems related to implication, assessments, modalities, ownership and so-called hidden equivalence. The corresponding results of the author have been presented and published in several international conferences (Matsak, 2005, 2006, 2008)

By transforming texts we understand a step-by-step modification of the original text into logic formulas. The procedure by P. Lorents (see introduction) consists of applying the steps listed below, while the order and the amount of use of the steps is not important. It follows that in some cases it is sensible to use the same step many times in many parts of the text. The steps are:

- *complementing* or adding necessary parts to the text.
- *withdrawing* or removing unnecessary parts from the text.
- *repositioning* or changing the relative positions of arguments within the text.
- *replacing* or substituting some parts of the text with some other (equivalent) texts.
- *identifying symbols* or finding parts of the text that can be represented as *individual symbols* (fully representing individual objects), *predicate symbols* (fully representing properties of objects or relationships between objects), *logic operation symbols* (negation, conjunction, disjunction, implication or equivalence), *functional symbols* (fully representing functional relationships, including logic operations), *quantifiers* (fully representing some part or all objects under observation) or *modality symbols* (characterizing the “validity” of some argument about an object, for example *definitely* or *possibly*).
- *categorizing symbols* or determining whether a symbol belongs to individual, predicate, functional, logic operation, quantifier or modality category.
- *positioning symbols* or reshuffling the symbols according to the rules of creating formulas.

Note 1. The part of text that is in the role of a logic symbol may not be in “one chunk”. Instead, it may be scattered. Typical examples are parts of text like *...if...then...; ...from...follows...* , which have the role of implication.

Note 2. Two steps (of the seven listed) may be applied at the same time, if the applier is sufficiently experienced to notice, which role a part of the text plays. Separating steps may be useful if the logic role is not clear. For example, the word “or” that can have the role of disjunction or conjunction, depending on the situation (Consider the possible interpretations of the text: *Which bus goes to the city centre? Bus 2, or bus 5, or bus 9*).

Next we will describe some semantic problems in transforming sentences. Of special note are the cases on assessments, modality and their complexes, hidden equivalence and ownership (Matsak, 2006).

3.1. The various roles of the word “siis“ (then)

The traditional role for the word “siis” [then] is implication.

Examples:

Kui tahad raha tagasi saada, siis saad (If you want to get money back, then you will)

$P1(x1, x2) \supset P2(x1, x2)$

Kui see dinosaur tuleb, siis ta pistab sind kõhtu (If this dinosaur comes, then it will eat you) $P1(x1) \supset P2(x1, x2, x3)$

However, while transforming texts, many other roles emerged for the word “siis”. First, the role of determining time, both the time of occurrence of a single event and establishing the order of occurrence of multiple events.

Examples:

Siis, kui mitte keegi ei taha kuulata (Then, when no one wants to listen) $\neg(\exists x)P(x, t)$

Ja siis me läksime lehma lauta rattaga ja siis andsime neile süüa, ja siis tegime pai neile. (And then we went to the barn with a bike and then we fed them and then we stroked their fur) $P1(x1, x2, x3, t1) \& P2(x1, x4, x5, t2) \& P3(x1, x4, x6, t3)$

The third role of the word “siis” is in performing as logic conjunction or disjunction.

Example:

Meeldib mängida ja siis magada, hästi mõnus on magada (Like to play and then sleep, sleeping is enjoyable) $P1(x1, x2) \& P1(x1, x3) \& P2(x3) \& [Val(x3) = \varepsilon]$

3.2. Assessments

The transforming of texts identified several operational constructs that may use predicates as operands. Let us analyze sentences like “*Ta ei saa üldse käia*” (*She can’t walk at all*), “*Ma tahan natukene lennata*” (*I want to fly a little*). It is clear that finding the truth value in these texts does not depend on just the predicates {*käia* [*to walk*], *tahan lennata* [*want to fly*]}, individual symbols {*ta* [*she*], *ma* [*I*]} and negation. It is important to understand what role do the parts of the text like {*üldse(käia)* [*at all(to walk)*], *natuke(lennata)* [*a little(to fly)*]}, which basically represent one-element “predicates applied on predicates”. In such cases it makes sense to introduce a suitable second order functional symbol that in essence represents giving an assessment: $Val(P) = \gamma$. These symbols are also necessary if we wish to assess or measure individuals or time.

An interesting aspect of assessments is related to viewing assessments as elements in a somehow ordered set. Namely, the assessments belong to a partially ordered structure where it is not always possible to tell, which one of *any two* assessments is “greater” (Figure 1, 2).

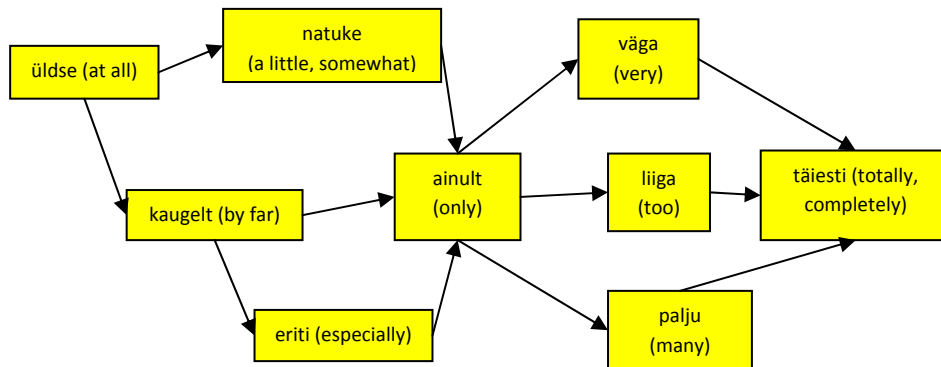


Figure 1. Assessments in partial order

Example:

Mul äsja oli sünnipäev [I recently had a birthday] → There exists a time when I had a birthday and the assessment (value) for this time is "recently".
 $(\exists t)P(x1, x2, t) \& [Val(t) = \tau]$

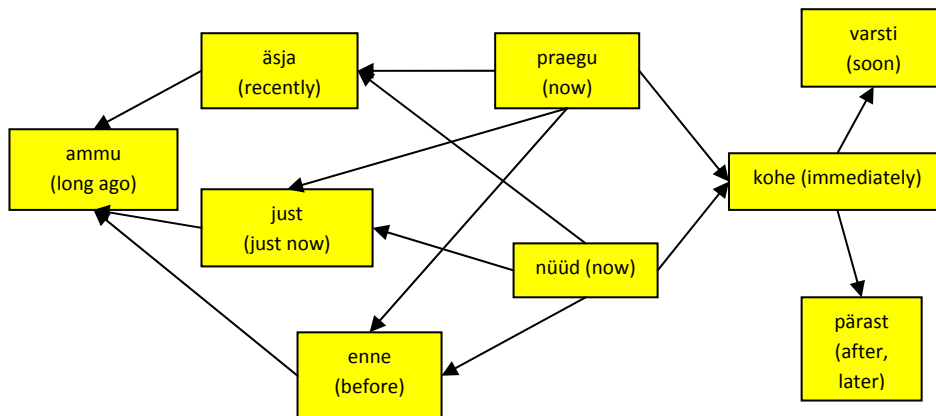


Figure 2. Example of time assessments in partial order

3.3. Modality complexes

While transforming children's texts, the "traditional modality" emerged with the words *võibolla* [maybe/possibly] and *vist* [maybe/guess]. For example, in sentences like "*Võib-olla need autod parkisid ette*" [Maybe these cars parked in front] $\Diamond P(x_1, x_2)$, "*Aga see on sinu oma, vist*" [But this is yours, maybe] $\Diamond P(x)$.

The words "võin", "saan" [I can] have a more difficult context. The first meaning is "*võib-olla on võimalik teha midagi ning see on reaalne võimalus*" [maybe it is possible to do something and that is the real chance]. The second meaning is the fact that the promised event will definitely occur, if someone asks for it. In many languages there are specific words to indicate this. For example, in English there are the words "may" and "can".

Examples:

Ma võin alla kukkuda, (I may fall down) $\models [\Diamond P(x_1, x_2)]$

Aga mina võin ainult silmadega lugeda, (But I can read with only eyes)
 $\models [\Box \Diamond P(x_1, x_2, t)]$

3.4. Ownership

There are at least two possible approaches to transform the words meaning "omada" [to own] into the language of logic. First approach involves using a predicate, since a predicate in essence represents a subset of elements that possess the relevant property. The second approach involves using the concept of ownership. For example, we can transform the sentence "These are doctor's things" into "These things belong to the doctor", which in turn leads to the formula $(\exists x)(\exists H)(x \in H)$ where "x" marks things and "H" marks the set of doctor's things.

3.5. On hidden equivalence

While analyzing the results of transforming children's texts, the texts that contained the word "muidu" (or else) revealed the use of hidden equivalence (Matsak, 2008).

Example:

Siia sa ei tohi tulla, muidu see onu hakkab pahandama [You can't come here, or else that man gets upset] → Kui sa tuled siia, siis onu hakkab pahandama ja kui sa ei tule siia, siis toimub sündmus X [If you come here, then that man gets upset and if you do not come here, then event X happens] (in many cases event X equates to "see onu ei hakka pahandama" [that man will not get upset]). → $(A \supset B) \& (\neg A \supset \neg B)$. It is easy to ascertain that this formula represents equivalence $A \Leftrightarrow B$.

In this chapter we have explained how it is possible to get from logical roles of the components of natural language text to the corresponding logic formulas. This process is the foundation for the creation and development of the dialogue system described in the following chapters.

4. TRANSFORMING TEXTS. INFERENCE LEVEL

In this chapter we describe how the logic formulas present in natural language text can lead to the logical inference steps and inference rules in the text. The relevant results of the author have been presented and published in international conferences (Matsak, 2007, 2008).

If a text is transformed into a formal shape and each argument is turned into a predicate calculus formula, then it is possible to investigate whether there are inference steps in those formulas or not.

Let us take a look at some well known inference steps, which are the basis of inference rules [table 3]:

Table 3. Examples of well-known inference rules

1. $\frac{A \supset B \quad B \supset C}{A \supset C}$ (Getzen's cutting rule)	4. $\frac{A \supset D}{(BVA) \supset (BVD)}$ (The rule of a sum from the Heyting system of arithmetic rules)
2. $\frac{\forall x[I(x) \supset E(x)] \quad \forall x[O(x) \supset I(x)]}{\forall x[O(x) \supset E(x)]}$ (Aristotelian syllogism)	5. $\frac{A(..b..)}{\forall xA(...x..)}$ (GN rule from the classical first order predicate calculus system of inference rules)
3. $\frac{A(...0...) \quad A(...x...) \supset A(...x+1...)}{A(...x..)}$ (Rule of complete induction)	

Note: According to the definition, “an inference rule is any set of inference steps, which consists of all pairwise similar inference steps. Every inference step belongs to some inference rule”. (Lorents, 2000)

Let us consider that each inference step must contain at least one atomic formula, which is present both in the premise and in the conclusion. Such an atomic formula corresponds to a one or multiple variable predicate with one or multiple individual symbols, respectively. In natural language the same word or phrase can correspond to several different individuals or predicates. Therefore, in order to find an inference step from a sentence, a single word must be used in the roles of individuals and predicates that represent the same object or concept. This

word must also cover all synonyms and pronouns. Analogous to text transformation, some parts in identifying inference steps may be hidden. In this case, the text needs to be complemented.

Example:

Andres on võtnud endale koera. Kui talle poleks koduloomad meeldinud, siis ta poleks endale koera võtnud. Andresele meeldivad koduloomad. [Andres likes pets. If he would not have liked pets, then he would not have taken a dog.]

The hidden argument in this text is “Andres has taken a dog”. A synonym in this case is “he”. By complementing the text and replacing the synonyms, we get the following line of reasoning:

Andres has taken a dog. If he would not have liked pets, then he would not have taken the dog. Andres likes pets.

$$\frac{V(a,d) \neg M(a,p) \supset \neg V(a,d)}{M(a,p)}$$

Applying the procedure also reveals other interesting aspects and questions. Namely, can some part of reasoning be considered an “element” or are some inference steps “hidden” between premises and conclusions? Secondly, is the inference step in question correct (in case of every interpretation, if the truth value of the premise formulas is 1, then the resulting formula’s truth value is also 1)?

The whole procedure for identifying inference steps involves the following phases:

- Complementing or including hidden arguments
- Repositioning or rearranging the parts of the text
- Withdrawing or removing an unnecessary part of text
- Replacing all synonyms and antonyms with a single word
- Transforming the text into formulas (see text transformation procedure above)
- Searching for matching formulas, displaying results

- Consecutive analysis of formulas. (It is often not enough to consider just a pair of sentences, but also “closely” preceding ones, because the premise may consist of multiple arguments)

Note: Unfortunately, the need for complementing, repositioning, withdrawing or replacing may emerge after the formula is created. But *this* formula does not suit us. For example, the meaning of the text may be lost in the process. This means that it must be possible to repeat some part of the transformation process. The first four transformations listed above can be repeated as many times as is needed. NB! The remaining three (transformation into formulas, searching formulas and analyzing them), however, must take place in exactly the order that they are listed.

In this chapter we described a method for finding logical (but not always correct) inference steps and implementations of inference rules from reasoning in natural language texts.

4.1. Identifying constructs from texts using the DST dialogue system

In this chapter we begin with the question: is the *transformation procedure* an algorithmic procedure (according to algorithm theory) or not. Then we explain the main principles and structure of the DST dialogue system. Finally we will discuss problems with DST development and ways to overcome these problems. The chapter identifies the problems and the corresponding solutions that emerged during the creation and development of the self-learning and –improving dialogue system DST. This system communicates from a distance with the morfalyzer software. The system uses novel methods, developed by the author, to extract logical inference steps (and rules) from natural language texts. The relevant results have been presented and published at international conferences (Matsak, 2005, 2006, 2007, 2008).

4.2. Problems with algorithmic insolvability

The author has continued her research based on the dialogue system that she developed for her Master’s thesis (Matsak, 2004). The system transforms natural

language texts into logic formulas. Next we will explain why this **has to** take the form of a dialogue (which **does not exclude** that sometimes the role of the user **may** be limited to inserting the text and receiving the formula from the computer).

Natural language texts are expressions (words, phrases etc.) that consist of symbols. Adding mathematical logic symbols to natural language symbols does not change this. We still have an expression that is written down using a specific alphabet. However, we can handle text transformation as some kind of a replacement system (for example some Thue system (Мальцев А., 1965)). At the same time it is known that even with “very modest” replacement systems the problem of word equivalence becomes algorithmically unsolvable. One requirement in the text transformation was that the output text had the same meaning as the input text (certain equivalence!). Based on this, it is easy to see that in some cases the transformation is impossible with a *single* algorithm. The algorithmic solvability of some problem, however, means that a single algorithm must guarantee that all solutions to that particular problem can be found. Therefore, the possibility of finding a single algorithm for text transformation is “close to zero”. As a result, an “oracle” is required to do transformation (this term is also used by Rogers in his book “Theory of recursive functions and effective computability”), or in other words a user who interacts with the program in a dialogue, providing “advice” to the program. (Rogers, 1967), (Успенский, Семенов, 1987)

4.3. Improved DST dialogue system. Formula level.

The first version of the DST dialogue system was completed as part of the author’s Master’s thesis (Matsak, 2004). In order to determine the logical role of words (according to the list in chapter 2) it was necessary to start with the sets of words and their grammatical roles. A web based morphologic analyzer developed by FILOSOFT [www.filosoft.ee] was used to create a system for identifying word categories. In the improved version DST turns to the analyzer to categorize the words in logic groups. During the process of transformation the user must confirm the morphological and logical roles that DST offers. One problem that has emerged from this process is that withdrawing, repositioning and replacing text requires almost constant “manual labour” from the user.

It turns out that if two sentences have the same structure (in terms of logic components and morphological forms having the same position in the sentences), then the sentences correspond to the same formula. This relationship allowed us to train the program to use the existing transformation scheme to identify similar sentences.

4.3.1. Training the dialogue system

The principle of training the dialogue system has remained the same during the DST development process. According to this principle, when a (structurally) new sentence is inserted into the system, the preliminary morphological scheme and the transformed sentence morphological scheme are stored.

Table 4. The meaning of morphological symbols

Symbol	Meaning
P+sg+#n#	Pronoun, e.g. <i>it</i> singular: nominative
V+#n#	verb, e.g. <i>to read</i> declarative, present, 1st person, active, positive - e.g. <i>I am reading</i>
A+sg+g#	adjective, positive, both declinable and indeclinable, e.g. <i>dear</i> singular: genitive
S+sg+kom#	noun (substantive), e.g. <i>thing</i> singular: comitative
S+sg+#n#	noun (substantive), e.g. <i>thing</i> singular: nominative
V+#b#!	verb, e.g. <i>to read</i> declarative, present, 3rd person, singular, active positive the relevant form of the verb “olema” [to be]
A+sg+#n#	adjective, positive, both declinable and indeclinable, e.g. <i>dear</i> singular: nominative

In addition, information about how the order of words has changed is stored. If a new word is introduced during the transformation process, then this is marked in the file that contains information about the order.

Example: [Matsak 2004]:

Let us train the program (to transform) with the sentence “Mina mängin ilusa nukuga” [I play with a pretty doll]. As a result of the training process we get the *scheme of morphological symbols*:

= $_P_+sg\+#n\# _V_+\#n\# _A_+sg\+g\# _S_+sg\+kom\# = _P_+sg\+#n\# _V_+\#n\#$
 $_S_+sg\+kom\# \& _S_+sg\+#n\# _V_+\#b\#! _A_+sg\+#n\# =$

(Morphological symbols are explained in Table 4)

and the order scheme: $0\ 1\ 2\ 3 = 0\ 1\ 3\ uus\&\ 3\ uus\ _V_+\#b\#!\ 2 =$

In the original sentence the words are numbered 0 (Mina [I]), 1 (mängin [play]), 2 (ilusa [pretty]), 3 (nukuga [with doll]). In the transformed sentence the order of words has changed. For example, the word in the third position must be number 3 (nukuga), not 2 (pretty). In addition, two new words have appeared: *uus&* (ja) [and] and *uus_V_+b#!* (on) [is].

The comparison between words is done based on the original words and the order scheme. The form of the word in the original sentence and the transformed sentence is not important. In the end, the formula $P_1(x_1)\&P_2(x_1)$ is stored.

Categorizing word roles by logic roles raises several problems. The feedback from the morphologic analyzer is often not unique. For example, the word “tee” [road, tea, do] can be categorized as a noun or a verb. As a result, the logical role would be an individual or a predicate. In case of nouns it is also important to determine the nominative case of the word, because it is necessary for forming a predicate according to the scheme “on+nimisõna nimetavas käänes“ [is+noun in the relevant case]. In order to solve this problem, a step-by-step morphological role confirmation took place for every word in the original system. The improved version uses by default the first available variant. Once the analysis is complete, a summary is presented, allowing the user to manually change morphological roles and restart the formula creation. The second problem was with categorizing words as logic operators and quantifiers. The system employs prepared word sets (in files) for purpose. These sets may need to be updated during the transformation process. This problem was solved by creating special software modules that provided the required capability.

4.3.2. Automatic sentence transformation

How does the automatic sentence transformation find the formulas that match original sentences? The system does not consider *semantics* and works as follows. Let us assume that the computer is trained with the sentence “Mina mängin ilusa nukuga” [I play with the pretty doll]. Upon entering the new sentence “Mina jalutan vallatu koeraga” [I walk with a cheerful dog] the dialogue-system recognizes a matching morphological pattern. The “original words” are identified

by the same morphanalyzer: *mina jaluta vallatu koer* [I walk cheerful dog]. We also know which order these words must have in the transformed sentence: *mina jaluta koer uus& koer uus_V_+#b#! vallatu*. By discovering a logic operator symbol in one of the new words, it is replaced with the corresponding “ja” [and]. Since the symbol “!” is tied to the word “olema” [to be] in the program, we can return the correct meaning. In addition, the required morphological form of the words in the new sentence is sent to the language synthesizer (FILOSOFT), which returns the words in their correct form. Thus, the transformed sentence “Mina jalutan koeraga ja koer on vallatu” [I walk with a dog and the dog is cheerful] is formed. The formula for this sentence is the same as the formula for the original training sentence: $P_1(x_1)\&P_2(x_1)$.

Note: As a result of the transformation process, empty fields often appear for words that were not present in the original sentence. The user is asked to provide the (new) words that fit in the empty fields. In principle, if there is no reason to fear misunderstandings of the created formula, then representation of the new formula in natural language is not required. The 2009 version of DST, however, works by finding the morphological scheme, generating a formula based on the training and asking whether or not the sentence should also be transformed in natural language.

4.3.3. Full text transformation

Transforming the full text (Matsak, 2008) is associated with several problems. While transforming in this mode we get formulas where predicates and individuals get the same symbols. In order to transform texts so that repeating words get the same symbol we developed some extra modules:

- A module that searches for repeating words (with lemmas of words) in the whole text, and assigns corresponding symbols to them. Symbols x_1, \dots, x_n are assigned to repeating individuals and A_1, \dots, A_n are assigned to repeating predicates (see chapter 2).
- A module, which first tries to locate a match to the logic role in the file that stores individuals and predicates. If that fails, the module searches a database for a matching sentence structure (for automatic transformation). If that also fails, the module proceeds with the logic role identification process.

Unfortunately for the text transformation, every natural language also possesses pronouns and synonyms. In order to transform a text, it must first be “cleaned up”. In other words, all pronouns and synonyms must be replaced with a single (base)

word. In order to minimize glitches and manual dialogue correction during transformation, these replacements should be carried out before starting the formula creation.

In the current version of DST the replacement of synonyms is handled manually. Future development may include a module, which will make the text clean-up easier by using the FILOSOFY synonym database in addition to the dialogue with the user. (Appendix A: figure 1).

In this module the inserted text is segmented into words and the cycles will be started based on the number of words. The text is processed by words. If the word is a pronoun, then the user must replace it with a concrete noun (individual). Words that share the same meaning (except pronouns) are replaced with just one synonym. In order to achieve this, a database of synonyms is used and every word's synonyms are compared to the words that have already been processed.

While doing full text transformation we meet with already familiar difficulties: it is often necessary to complement some parts of the sentence or to replace some words, regardless of the fact that the pronouns and synonyms have already been replaced in the previous step. If the module for replacing synonyms in Figure 3 is implemented, then for every added or replaced word we must generate synonyms and check whether one of them already exists in the symbol-meaning relationship file. If a match is found, then it is used in the place of the added or replaced word. Otherwise a new symbol is assigned. The user must make sure that the vocabulary remains suitable.

The synonyms, however, are only a part of the problem. During the transformation process, for example, a transformation (including replacements) of the third sentence may generate a new word, which happens to be a synonym for a word in the first sentence. The latter word already has an assigned symbol and the formula for the first sentence is determined. Therefore, a situation may develop where synonyms of the same word correspond to different symbols in the formulas. This, in turn, may unnecessarily produce different formulas for texts that in reality have the same meaning. In order to cope with this problem, we added a module to DST that, upon discovering this situation, generates a new symbol x_i vöi A_i (let us remember that repeating individuals and predicates are assigned with symbols x_1, \dots, x_n and A_1, \dots, A_n , while non-repeating ones are assigned with symbols q_1, \dots, q_n and P_1, \dots, P_n) and also makes the necessary replacements in the first sentence. This is implemented by adding a line in the corresponding file, which contains words,

symbols and the formula for each sentence. Once the full text is transformed, the file is emptied.

4.4. Improved DST dialogue system. Inference level.

In this section we view aspects of creating a software solution for identifying inference steps in natural language texts.

Firstly, let us reiterate again and again that one of the most important capabilities of an intelligent system is the capability to apply instruments of logic. At the *formula level* this means the ability to construct formulas (using logic symbols like predicates and operators in a certain way). The next important level is the *inference level*, which deals with the capability to infer the conclusions of one formula based on other formulas (by using inference rules or relations between formulas in a certain way). In the human intellect, the formula level is manifested in our ability to formulate clear arguments, whereas the inference level can be seen in our ability to logically justify our arguments.

A large fraction of the systems that are researched and implemented today have been “mined” from texts representing human thought. This includes the concept of forming formulas and the rules for inferring new formulas. On the other hand, two important aspects have remained in the background:

How and with what “technology” natural language texts should be processed, in order to identify the logic constructs within

How and with what “technology” these logic constructs have (been) developed (including inference in various logic systems, which differ amongst themselves in terms of inference rules and their application).

A natural question is how to implement these concepts in information technology. Therefore, we will continue by exploring the aspects related to inference rules and how they can be extracted from natural language texts. We will start with a short description of the customized Lorents’ method (Lorents, 1993, 2002) (also see introduction chapter), which helps to explain in a theoretical level some approaches of the author for creating a practical software implementation.

4.4.1. Customizing the Lorents' method for searching inference rules

Let us begin with the fact that inference rules represent multiple figure relations between formulas. k -tuple relations between the elements of some set H are, however, k -th Cartesian power subsets of $H^{(k)}=H \times \dots \times H$. We can rely on Lorents' procedure (Lorents, 1993, 2002) for forming k -tuples by using two meta-predicates: R the meta-predicate of relatedness and S the meta-predicate of similarity.

The meta-predicate of relatedness expresses that some objects in a tuple are all related to each other in the same way. For example, the numbers in the triples $\langle 0,0,0 \rangle, \dots, \langle 2,2,4 \rangle, \dots, \langle 4,2,6 \rangle, \dots, \langle 3,5,8 \rangle, \dots, \langle 5,5,10 \rangle, \dots$ are clearly related to each other. It is also true for the numbers in the following triples: $\langle 0,0,0 \rangle, \dots, \langle 2,2,4 \rangle, \dots, \langle 4,2,8 \rangle, \dots, \langle 3,5,15 \rangle, \dots, \langle 5,5,25 \rangle, \dots$.

The meta-predicate of similarity expresses that some same-level tuples are similar in a certain way. For example, the triples $\langle 0,0,0 \rangle, \dots, \langle 2,2,4 \rangle, \dots, \langle 4,2,6 \rangle, \dots, \langle 3,5,8 \rangle, \dots, \langle 5,5,10 \rangle, \dots$ are similar in some way. The same is true for the triples $\langle 0,0,0 \rangle, \dots, \langle 2,2,4 \rangle, \dots, \langle 4,2,8 \rangle, \dots, \langle 3,5,15 \rangle, \dots, \langle 5,5,25 \rangle, \dots$. On the other hand, we can agree that the triples $\langle 4,2,6 \rangle$ and $\langle 3,5,15 \rangle$ are definitely not similar in "the same way".

Lorents proved (Lorents 1993) that the following procedure will identify all relations that can be characterised by meta-predicates R and S that are "acting upon" some set H . Two cycles need to be started. One cycle finds only tuples where the objects in the tuple are related with each other. For every tuple identified by the first cycle the other cycle collects all tuples that are similar to it and each other. Every such "collection" represents a specific relation between the elements of H .

If, for example, H is a set of formulas extracted (by transformation) from a text, then the procedure will identify any inference rules that connect them.

By applying this procedure on natural language texts created by children of a certain age (age recorded at the time of creating the text), for example, it is possible to find out:

- Which inference rules are *actually* used (Is it just Aristotelian syllogism, Gentzen’s sequential computing, Heyting’s arithmetic etc. or are there more rules?)
- At what stage of development certain inference rules “appear”.

4.4.2. Inference step identification module

The first version of DST, capable of identifying inference steps, was created in 2008 (Matsak, 2008). In order to achieve this, a full text transformation capability had to be developed. In chapter 4 we reviewed the theory behind the procedure of identifying inference steps and observed the need to prepare the text with complementing, repositioning, withdrawing, as well as replacing synonyms and antonyms.

Example:

Let us assume that we must transform the following sentences: “*If the sun is shining, then it is warm. If the sun is not shining, then it is cold.*” The user must replace the word “cold” with the phrase “not warm”.

In the current version of DST these steps must be (initially manually) completed by the user before the logic formula creation process is started. In the future, a separate module can be used to query a database of synonyms and antonyms to assist the user in this task. All replacements must be made in order to be able to do full text transformation (see 5.2.3).

Let us recall that when transforming text as a “single full entity” (in the corresponding mode) we get formulas where repeating predicates and individuals are marked with the same symbols.

Next, an array is formed out of these formulas and all elements in the array are then compared in order to find common characteristics. For example, repeating individuals and predicates, as well as repeating atomic formulas that they form.

To some extent we can say that we are applying the meta-predicate R, since we are searching for formulas that are related to the same inference rule. Then we form doubles, triples, etc. out of the formulas identified during transformation, while making sure that each inference rule must contain at least one premise and one conclusion. By forming various doubles and triples (or higher level tuples) we can identify, which of these possess common parts of formulas, or in this case – atomic formulas (Matsak, 2007). This is somewhat similar to the application of the meta-

predicate S , with the distinction that similarity is sought on the basis of the presence of the same atomic formulas. As a result we get pairs that contain equivalent components and are at the same time related to each other. Such collections can represent inference steps or even inference rules with one or more premises. Unfortunately, pairs and triples that “look” similar, but are not logically correct may also surface. Or some necessary formulas may be absent in the identified pairs, which should exclude these pairs from logical inference steps. Often the premises and conclusions may be represented in mixed order. Despite these “interfering circumstances” the application of the algorithm in question enables us to perform the search for inference steps much quicker, more convenient and accurate than a human doing the same “manually” (the human may not notice necessary parts of the texts or the similarity between parts of text).

There exists an algorithm that can be used to perform the above described tasks. It works partly due to the fact that we have assigned different symbols to repeating and non-repeating words. In other words, we must only analyze the formulas that contain the symbols x_1, \dots, x_n and A_1, \dots, A_n (from now on “A and x-type” symbols).

The identified formulas and the corresponding sentences are formed into new matrices (Appendix A: figure 1a). According to the number of selected formulas the data is prepared for identifying inference steps (Appendix A: figure 2b). Each formula is split by spaces in the written text. The result is the matrix *@cons*, which corresponds to the formula. Every component of the matrix is then processed. First, the parentheses are removed. Second, if the component contains the symbols x or A , then it will be added to the table where each row corresponds to a formula and the slots in a row contain the parts of the formula that include the symbols x or A . Then the longest row is found. According to the maximum number of rows and slots the cycles are started and every slot is compared to every following slot (Appendix A: figure 2c). By default the existence of a predicate or an individual is *false*. However, if a slot is found that matches the contents of a following slot, then the “full formulas” of the corresponding rows are noted as a pair of formulas. If it turns out that a pair already existed, then it is not stored in memory. In addition, the slot in question is examined to see if it contains an individual (x) or a predicate (A). If at the same row, both a slot with an individual and a slot with a predicate is found, then the corresponding “full formula” pair is output as a potential (single predicate) inference step.

This approach simplifies the search by several times (Matsak, 2008).

Example:

If we have identified a pair of formulas $\langle P_3(x_2) \& A_4(x_1), A_4(x_1) \rangle$, then we see that the formulas in the positions of premise and conclusion contain “A and x-type” symbols. We see that the atomic formula $A_4(x_1)$ is repeating in both the premise and the conclusion. Based on this we can handle the formula as resulting from an inference rule (that means, an inference step, which can be represented as $\frac{P_3(x_2) \& A_4(x_1)}{A_4(x_1)}$).

The module that searches for pairs can be easily modified to search for triples. As before, the similarity can be ascertained by searching for common atomic formulas (although, several iterations may be necessary).

Note: The tuples or presumably “well-formed” inference steps received in this fashion may not always be the “true” inference steps. For example, the identified steps could be logically incorrect. Another example is when a necessary argument is missing in the text, because the author of the text has not explained the reasoning in sufficient detail.

The identified steps are stored in a database and a sheet with sentence information is created for each step. These sheets can be reviewed in order to see how many times a certain step has been used. In case of the records in the database, it is important that DST finds equivalent formulas like $\frac{P_1(x_1) \& A_{10}(x_4)}{A_{10}(x_4)}$ and

$\frac{P_3(x_2) \& A_4(x_1)}{A_4(x_1)}$ equal.

More detailed examples of inference step identification can be found in the author’s relevant publication (Matsak 2008).

4.5. The general layout of the program

In this section we review shortly the latest version of DST dialogue system. The algorithm for the first version of DST was presented in 2005 (Matsak, 2005). The first update to this version followed in 2006 (Matsak, 2006). Over the following years several improvements were made to the system by adding new modules and by modifying the existing system.

The latest version of DST is oriented towards transforming “simpler” sentences, which do not result in higher order formulas where an atomic formula is in the place of an individual in a predicate. An example of a “not simple” sentence is *Ta ütles, et me läheme külla [He says that we are going to visit (someone)]*, which can be represented by the formula $P_1(q_1, P_2(q_2, q_3))$. When transforming “complex” sentences the user must provide more complements and corrections, possibly at every step of the transformation process, since DST is a dialogue system, not a fully automatic transformer. In order to correct formulas, a special editing feature has been included that makes adding missing symbols much easier.

The development process of the dialogue system, including the changes and additions to the modules, is covered by the author’s publications between 2005 and 2008. The figure 3 (Appendix A) represents the layout of the latest version of DST.

5. LOGIC CONSTRUCTS IDENTIFIED WITH DST FROM CHILDREN'S TEXTS.

In this section we provide an overview of the logic constructs that emerged from children's texts after the transformation with DST, whereby we consider two different levels: formulas and inference steps.

The aim of this transformation was to find out at what age certain logic constructs *may* be present in humans. In other words, at how early age *may* children display capability to operate with certain logic constructs? Specifically, we are interested in both the "first appearance", as well as "intensity" of these constructs.

The age groups under study are:

First age group consisted of children between the ages of 1,2 and 2,8 years. The data came from two sources:

- recordings of Estonian children talking
- CHILDES database (see Child Language Data Exchange System).

The files in the CHILDES database are recordings of children from many countries and the corresponding texts. For the Estonian recordings a group of children of appropriate age were selected and interviewed twice a month over a period of several months.

The youngest children provided a few single word answers. Over time (months) it is possible to see how the answers change into sentences. The results of transforming these texts show, which logic constructs appear at what age. This thesis *does not* draw statistical generalizations for all children, as the focus of the work is on developing DST and researching ways to use it. The children's texts serve as a very useful material, because they are not too complex for DST in this phase of development. In addition, they provide an initial overview of the set of logic constructs used by self-evolving intelligent systems and "traces" of the development of the capability to use them.

5.1. Formula level

The results of transforming the children's texts (figures 3,4) brought out some unexpected aspects (Matsak 2009):

- (A) The negation operator was observed already at an age of 1,2 years
- (B) Most of the rest of the logic operators, quantifiers, as well as some modalities are present in two year olds. The values in the figure below have been calculated based on the use of an operator or quantifier compared to the total amount of children observed.
- (C) Zero values are not indicated on the figure. Lack of plot points before 1,7 years means that the corresponding logic operators or quantifiers did not emerge in the transformation process. Lack of plot points for older children is due to lack of recordings, as this age group was not studied. It is important to note that the presence of logic operators at a very early age is not only a factor of age, but the general development of the child's ability to express her thoughts. Identifying negation in such an early age (1,2 years) was due to a single child who could talk much better than other children of that age. Earliest use of implication was detected in 2,1 year olds.

The development of such a logical arsenal over time and in that order can be explained as follows: when an intellectual system (in this case the logical tool set of a human) forms, a knowledge base must be created and truth values must be assigned to the atomic formulas that correspond to knowledge (let us remember that knowledge is an ordered pair $\langle X, Y \rangle$, where $X \uparrow Y$, or X and Y are related to each other with the notation-denotation relation \uparrow) (Lorents, 2001). Knowledge $\langle X, Y \rangle$ is considered correct, if the corresponding formula $X \uparrow Y$ is correct. It is "technically easiest" to recognize correct and incorrect knowledge by using the negation operator. This illustrates the need to operate with non-atomic formulas. Even more complex non-atomic formulas are necessary in order to get from existing correct pieces of knowledge to a new correct knowledge item. Primarily, formulas containing one or more conjunctions (and negations) are needed. The reason for this is that aside from knowledge about objects that have a certain property (for example, the cake is sweet) we also need knowledge about objects with multiple different properties (the cake is sweet and the cake is big). Next,

quantifiers are needed to indicate belonging to a set (these are all my toys; some of these toys are mine).

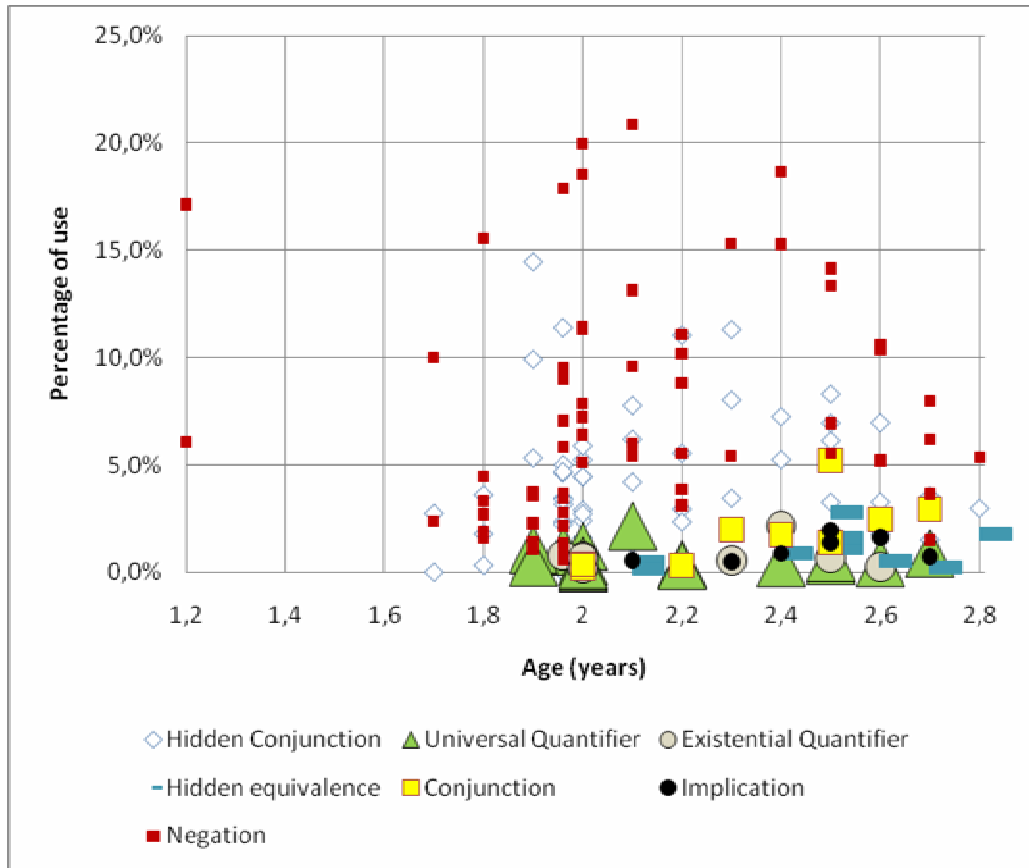


Figure 3. Classical logic operators and quantifiers at age 1,2 to 2,8

Without these operators we are unable to draw even the most general conclusions. In order to replace ever more complex knowledge with equivalent, but possibly simpler formulas, we need to use logical equivalence. Even in hidden form, equivalence prepares a system for substitutions. For example, substituting a group of simultaneously applicable properties with a new property that is basically a conjunction of the individual properties. With the necessary components present in the form of formulas, we can start to use implication, which takes us from simple if-then statements to inference steps and inference rules (for example, if you take my candy, then you are naughty; you took my candy : you are naughty).

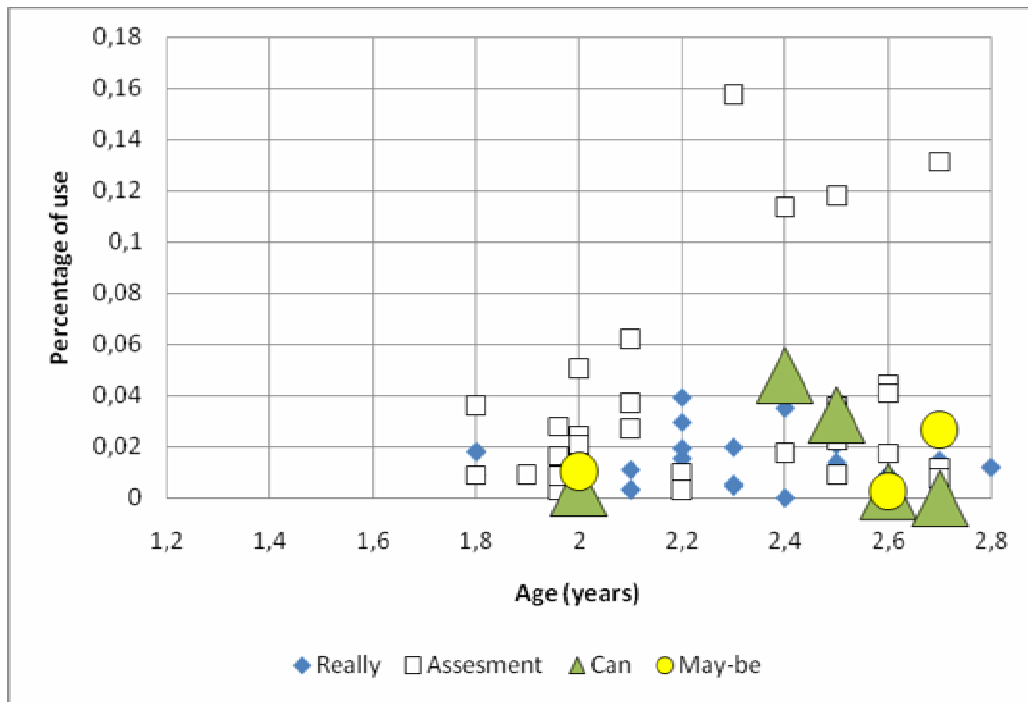


Figure 4. Non-classical operators and complexes at age 1,2 to 2,8

Next we explain the use of non-classical modalities (and sets of modalities) in the early stages of the development of a child’s intellect. The modality “ongi” [*is so*] is used to verify, check and defend the existing knowledge in arguments. Modalities can also be used as a support tool for ascertaining the correctness of knowledge in situations where it is not yet clear or may never become clear whether the knowledge is correct or incorrect. Modalities “maybe” and “can” allow the assignment of truth values depending on the situation. Sometimes, however, they provide alternatives that in essence are equivalent to disjunction. For example, the statement “may-be yes and may-be no” is essentially very similar to the statement “exists or not”. The lack of classical disjunction in a “young” intellectual system may be explained with real and complex processes at the neuron level of the human brain. Assessments are also important for verifying some knowledge as correct or incorrect. In later stages of development, modalities, including assessments, are used in probabilistic models and risk assessment. The childrens text studied by the author confirm the existence of a logical arsenal containing quantifiers and modalities as early as (NB!) two and a half years of age, which

were followed by first tries of applying inference rules (at first, including incorrect rules) (table 5).

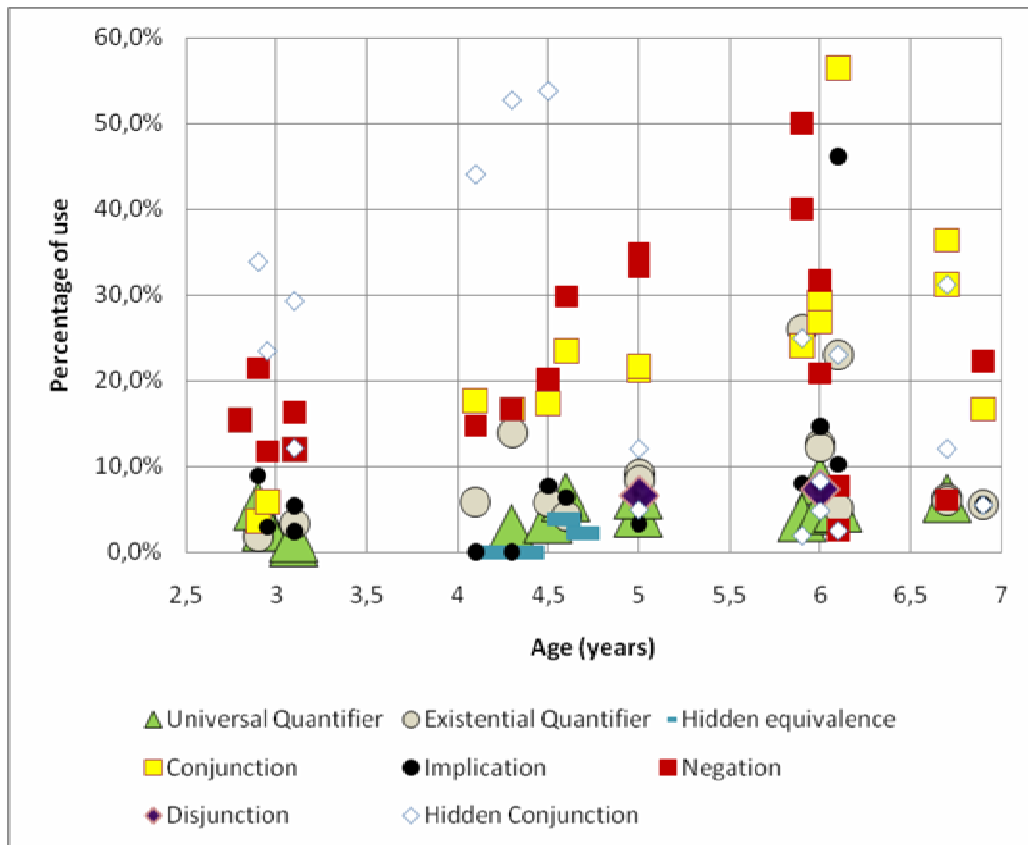


Figure 5. Classical logic operators and quantifiers at age 2,8 to 6,8

The second age group consisted of children between the ages of 2,7 and 6,8. The recordings for this group were a part of this doctoral research and were collected with the help of primary school pedagogy university students and kindergarten teachers.

The figures (5, 6) show the increase of use of logic operators and quantifiers over time. It is interesting that disjunction was very rare in both CHILDES and Estonian texts (Only two children, ages 5 and 6, used this operator). The results also show that the use of conjunction and negation grows much faster compared to other operators. Assessment and modality “can” are also often used, while

quantifiers and implication are not as common. It should be noted that implication is required for using inference steps in justifying arguments.

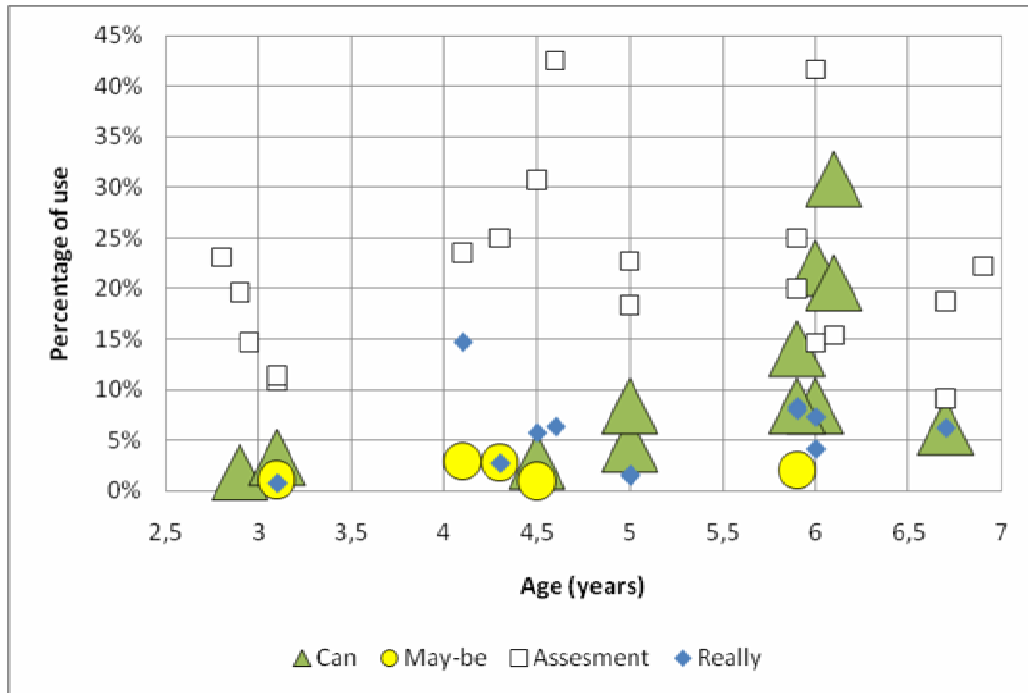


Figure 6. Non-classical operators and complexes at age 2,8 to 6,8

In 2007 the texts of 10 and 11 year old children’s essays was analyzed (Matsak, 2007). These essays were collected by K. Pata for the project “Seostatud kontseptuaalse arusaamise kujunemine keskkonnaalastest probleemteemadest” [The development of related conceptual understanding of environmental problem topics], which consists of 387 sentences from 175 children. The analysis was not aimed at transforming the text into formulas (the sentences were often complex, requiring additional modules for DST), but on categorizing the words in the sentences as logic operators and quantifiers. The texts were also analyzed to find inference steps in the justification of arguments.

The following figure (7) represents the average frequency of logic operators, quantifiers and assessments *per sentence*. If you compare the data to the age groups in the previous figures, then the first notable change is the increase of the use of conjunction (specifically, explicit conjunction) and the second one is the jump in the use of disjunction.

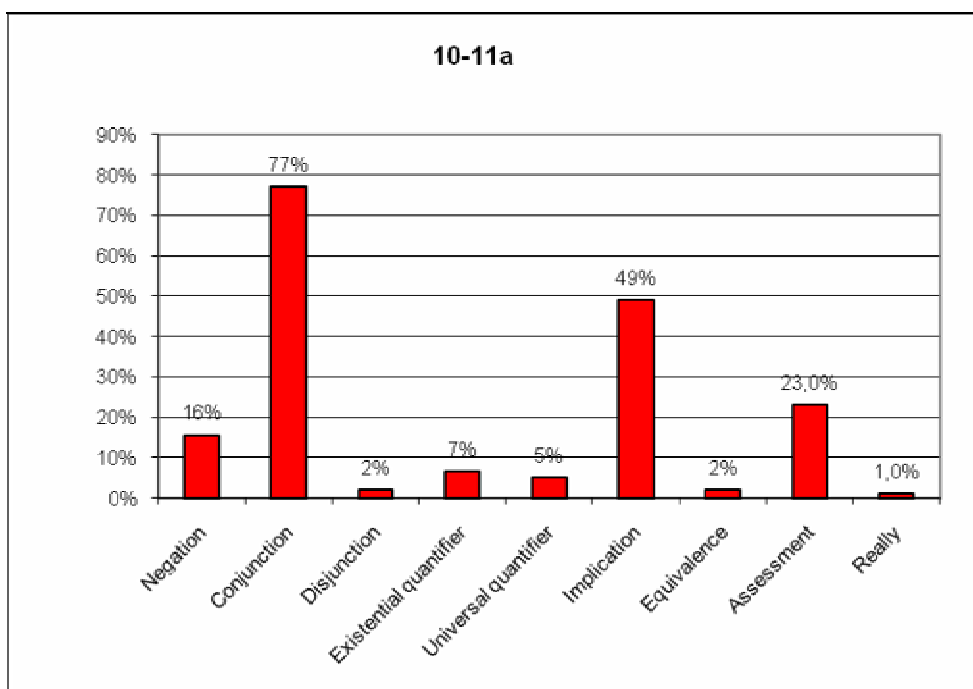


Figure 7. Logic operators and quantifiers at age 10 to 11

An interesting fact is that for some reason no modality was detected in the texts. This does not mean, however, that modality should not or cannot occur. There may be a very simple reason for this situation: children answered previously fixed questions, which possibly did not require modality in the answer.

5.2. Inference level

The development of the DST prototype has revealed a relatively diverse arsenal of inference instruments that is used by very young children to justify their arguments. Like many adults, the children also try to construct necessary justifications for their arguments with logically correct, as well as seemingly correct (but actually incorrect!) inference steps. (Matsak, 2009) Table 5 represents the inference steps that the author identified in the analyzed texts.

Table 5. Inference steps identified in children's texts

1. $\frac{A}{A}$ (2y 9m)	5. $\frac{A \& B}{A}$ (3y 1m)	9. $\frac{A \supset \neg B}{B \supset \neg A}$ (5y 11m)
2. $\frac{A \supset \neg B}{\neg A \supset B}$ (2y 6m)	6. $\frac{(\exists t)A(x, t)}{A(x, t)}$ (3y 1m)	10. $\frac{A \quad A \supset B}{B}$ (10-11)
3. $\frac{A \supset B}{\neg A \supset \neg B}$ (2y 6m)	7. (5-6) $S(x) \supset (\forall x)A(x)$ $\frac{\neg S(q)}{\neg A(q)}$	11. (10-11) $\frac{\neg K \supset X \quad \neg X}{K}$
4. (2y 6m) $\frac{(\exists \alpha A(\alpha) \& \exists \beta A(\beta))}{\forall x A(x)}$	8. $\frac{\neg(\exists t A(t, x))}{(\forall t)\neg A(x)}$ (5y 9m)	12. (10-11) $\frac{\neg K \supset \neg S}{S \supset K}$

The texts recorded from younger age groups were not directly related to justification of arguments. Therefore, it was not possible to gain a reliable overview on what types of transitions, inference steps and inference rules children use. However, some interesting logical transitions were discovered in the justification process. Some of them were logically correct while some were incorrect, but “looking” very similar to correct transitions. In here, logically correct means that if the formulas in the premise of the transition are correct, then the resulting conclusion formula must also be correct (see Lorents 2000, “Keel ja loogika“, §10).

At this point we will review the structure of the rules of contra-position, Modus Ponens and syllogism. Table 5 shows that these rules represent the most frequent correct and incorrect inference rules that surfaced in the studies. First we show that in contra-position, the premise and conclusion is basically the same formula.

$$A \supset B \leftrightarrow \neg A \vee B$$

$$\neg B \supset \neg A \leftrightarrow \neg A \vee B$$

This gives cause to assume that the mechanism that „technically” executes both implications is actually the same in both cases. The mechanism lets you read the

formula in the „other way”, if the truth value of the conclusion is the opposite of the truth value of the premise.

If we view the implication as a digital circuit, then it is easy to understand the simple human mistakes in applying the rule. In order to get the correct result, the signals that have the inverse values of the premise signals must be sent to „inverted” inputs (premise signal one goes to port two and vice versa). If the inversion is not implemented we will get a circuit that corresponds to an incorrect inference rule, such as $\frac{A \supset B}{\neg A \supset \neg B}$.

In the case of Modus Ponens a human must operate with an extra formula in the premise, compared to the contra-position. The verification of suitable inputs is more complicated: one signal (formula) of the two must be input twice (as an atomic formula and as a signal for the implication part). The result is the signal (formula) that was input once. In the classical case the truth value for the double signal must be the same and it must be the signal of the first input (in relation to the implication inputs). From the texts of 10-11 year olds we could also find examples of the use of Modus Tollens (which is basically a combination of Modus Ponens and contra-position), where the second input signal of the above example was used. This may refer to children „experimenting” with different inputs and outputs to the inference rules. It seems very probable that a human would not be able to use Modus Ponens if she could not use implication and conjunction. The use of Modus Tollens, in turn, seems to require the skill to use contra-position. At ages 5-6 an incorrect transition emerged that was similar to a syllogism:

$$\frac{S(x) \supset (\forall x)A(x)}{\frac{\neg S(q)}{\neg A(q)}}$$

In classical Aristotelian syllogisms the third statement should be derived from the two implications in the premise. In here, however, the child has used something between the contra-position rule and the syllogism rule, which may be one possible sign of the development of the ability to use logic rules. Table 5 shows attempts to use quantifiers and related incorrect (at first) inference steps, which at the same time look very similar to the inference steps of correct inference rules.

SUMMARY

When building intelligent systems it is crucial to ensure that they are able to form correct decisions based on the described situation.

This, in turn, requires the capability to operate with logic constructs. In a special case, when the systems are designed to be self-evolving (learning), we must also consider the nature of the mechanisms that drives the learning process. One way to research this is to study existing self-evolving systems – humans (in their development phase), in order to explain the development and adoption of logic instruments. More precisely, at what age various logic instruments first develop in children.

One of the best ways to study this is to analyze the natural language texts, finding “explicit” and “hidden” logic constructs (for example, arguments represent logic formulas, while reasoning represents inference). A necessary prerequisite for such an analysis is the transformation of natural language texts into the corresponding logic constructs.

The main results of Erika Matsak’s PhD thesis are as follows:

- Based on the author’s theoretical studies, research results and software, novel methods have been developed for creating a dialogue system for extracting logic constructs (including formulas, inference steps and inference rules) from natural language texts
- The development of a working prototype of the DST dialogue system and its improvements that performs the transformation process at the logic formula level, which allows the extraction of individuals, predicates, quantifiers, modalities, logic operations and logic formulas that are represented (for example, as words) in natural language texts
- The development of the next level of the DST prototype, which operates at the logical inference level to identify logical inference steps and rules from natural language texts
- The dialogue system DST has been used to study children of different age groups using various logic constructs, which confirm a surprisingly diverse arsenal of logic instruments at a very early age.

Actually knowing the nature of logic instruments available for children of a certain age enables us to better analyse, plan and execute future research in the field of the development of the human thought process.

Identifying logic constructs from natural language texts also creates other possibilities aside from understanding and supporting children's development. For example, identified "personalised" inference steps (inference rules) can be used to "game" through potential actions and decisions of an intelligent system (e.g. person or group) based on the descriptions of certain situations (for example, see "modelling Bismarck"). This enables us to predict future actions, allowing systems that are "aware" of this to more effectively plan their own actions.

The system, which is theoretically justified and developed as a software prototype by the author, is unique in Estonia as well as the rest of the world. The technological solutions presented here provide new opportunities to study the development of children as natural self-evolving intelligent systems, including the study of the development of logical tools in humans. This, in turn, presents new opportunities as well as new and interesting problems (in software and hardware (see Matsak 2009)) for researching the human made decision support systems and (self-evolving) intelligent systems that use natural language and logical tools.

REFERENCES

- Allen J.F. (1983). Maintaining knowledge about temporal intervals, *Communications of the ACM* 26(11) 832–843.
- Birkhoff G., von Neumann J. (1936). The logic of quantum mechanics. *Ann. Math.* 37, 823–842.
- Chittaro L., Montanari A. (2000). Temporal representation and reasoning in artificial intelligence: Issues and approaches, *Annals of Mathematics and Artificial Intelligence* 28, 47–106
- Church A. (1936). A note on the Entscheidungs-problem, *J. Symbolic Logic* 1. 40-41 (Correction, *ibid.*, pp. 101-102
- D. G. Bobrow. (1964) Natural Language Input for a Computer Problem Solving System. Ph.D. Thesis, Mathematics Department, M.I.T., Cambridge, Mass.
- Dragalin A. G. (1979). Математический интуитсионизм. Введение в теорию доказательств. [Mathematical intuitionism. Introduction to proof theory] “Наука”. Москва.
- Gentzen G. (1936). Die Widerspruchsfreiheit der reinen Zahlentheorie. [The consistency proof of pure number theory] *Math. Ann.* 112, No 4, 493 – 565
- Golumbik M.C, Shamir R. (1993). Complexity and algorithms for reasoning about time: a graphtheoretic approach, *Journal of ACM* 40(5) 1108–1133.
- Green B. F., Wolf A.K , Chomsky C, Laughery K. (1961) Baseball: an automatic question-answerer. Western joint IRE-AIEE-ACM computer conference, May 09-11, 1961, Los Angeles, California
- Levesque H.J, Reiter R., Lesperance Y., Lin F., Scherl R. (1997). GOLOG: a Logic Programming Language for Dynamic Domains, in [155] pp. 59–83.
- Lorents D. (1992). Loogilised konstruktsioonid eesti keele tekstides. Valemite tase. [Logic constructs in Estonian language texts. Formula level] Lõputöö, Tallinn 1994
- Lorents P. (1993). Algebraic Framework for Knowledge Acquisition. Royal Institute of Technology. Departement of Teleinformatics. ISRN KTH/IT/R – 93/10-SE.

Lorents P. (2000). Keel ja loogika. [Language and logic] Tallinn: Estonian Business School.

Lorents P. (2001). Formalization of data and knowledge based on the fundamental notation-denotation relation". Proceedings of the International Conference on Artificial Intelligence. IC – AI' 2001. Volume III. p. 1297 – 1301

Lorents P. (2002). A system mining framework. The 6-th World Multiconference on Systemics, Cybernetics and Informatics. SCI – 2002. Proceedings. Volume I. p. 195 – 200.

Matsak E. (2004). Eestikeelsetes tekstides sisalduvate loogiliste konstruktsioonide väljaeraldamise süsteem. Valemite tase. Magistritöö [A system for identifying logic constructs in Estonian Language texts. Formula level. Master's thesis.] TPÜ, Tallinn

Matsak E. (2005). Dialogue system for extracting Logic constructions in natural language texts. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2005. Volume II p. 791 – 797. Las Vegas, Nevada, USA

Matsak E. (2006). System DST for Transforming Natural Language Texts, Representing Estimates and Higher Order Predicates and Functionals. The 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2006. Orlando, Florida, USA.

Matsak E. (2006). Using Natural Language Dialog System DST for Discovery of Logical Constructions of Children's Speech. The 2006 International Conference on Artificial Intelligence IC-AI 2006. Las Vegas, Nevada, USA

Matsak E. (2007). The prototype of system for discovering of inference rules. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2007. Volume II p. 489 – 492. Las Vegas, Nevada, USA

Matsak E. (2008). Improved version of the natural language dialog system DST and its application for discovery of logical constructions in children's speech. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2008. Volume II p. 332 – 338. Las Vegas, Nevada, USA

Matsak E. (2009a). Representing logical inference steps with digital circuits. Lecture Notes in Artificial Intelligence: HCI International 2009. Springer [submitted]

Matsak E. (2009b). On the logic module in intelligent systems. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2009 Las Vegas, Nevada, USA [submitted]

Mints G., Tyugu E. (1982). Justification of the Structural Synthesis of Programs. Science of Computer Programming. v. 2. No. 3. 1982. 215 - 240

Pereira F.C.N., Warren D.H.D (1980). Definite Clause Grammer for Language Analysis, Artif. Intell., 13, 3, pp. 231-278

Rogers H. (1967). Theory of recursive functions and effective computability. McGraw-Hill, New York. [Теория рекурсивных функций и эффективная вычислимость. Издательство «МИР» Москва 1972]

S. Osugi U. Saeki (1990). Приобретение знаний / под редакцией С. Осуги, Ю. Саэки ; перевод с японского Ю.Н. Чернышова . Москва : Мир. [Knowledge acquisition: translation from Japanese, Mir, Moscow]

Tyugu E. (1970). Решение задач на вычислительных моделях. [Solving problems with computation models] ЖВМ и МФ, т. 11, Nr 5

Tyugu E. (1972). A data base and problem solver for computer-aided design. Information Processing, 71. pp 1046 – 1049. North-Holland. Amsterdam.

Tyugu E. (1988). Knowledge - Based Programming. Addison-Wesley

Tyugu E. (2007). Algorithms and Architectures of Artificial Intelligence. IOS Press. Amsterdam. Berlin. Oxford. Tokyo. Washington, DC.

Атаян В.В. (1986). Использование алгебраических конструкций в алгоритмах анализа математического текста. Сборник научных трудов, Методы алгоритмизации и реализации процессов решения интеллектуальных задач. Академия наук Украинской ССР. Киев.

Мальцев А. (1965). Алгоритмы и рекурсивные функции. [Algorithms and recursive functions] Издательство «Наука» Москва

Успенский В. А, Семенов А. Л. (1987). Теория алгоритмов: основные открытия и приложения. [Theory of algorithms: main discoveries and applications] Москва, Наука

APENDIX A: ALGORITHMS

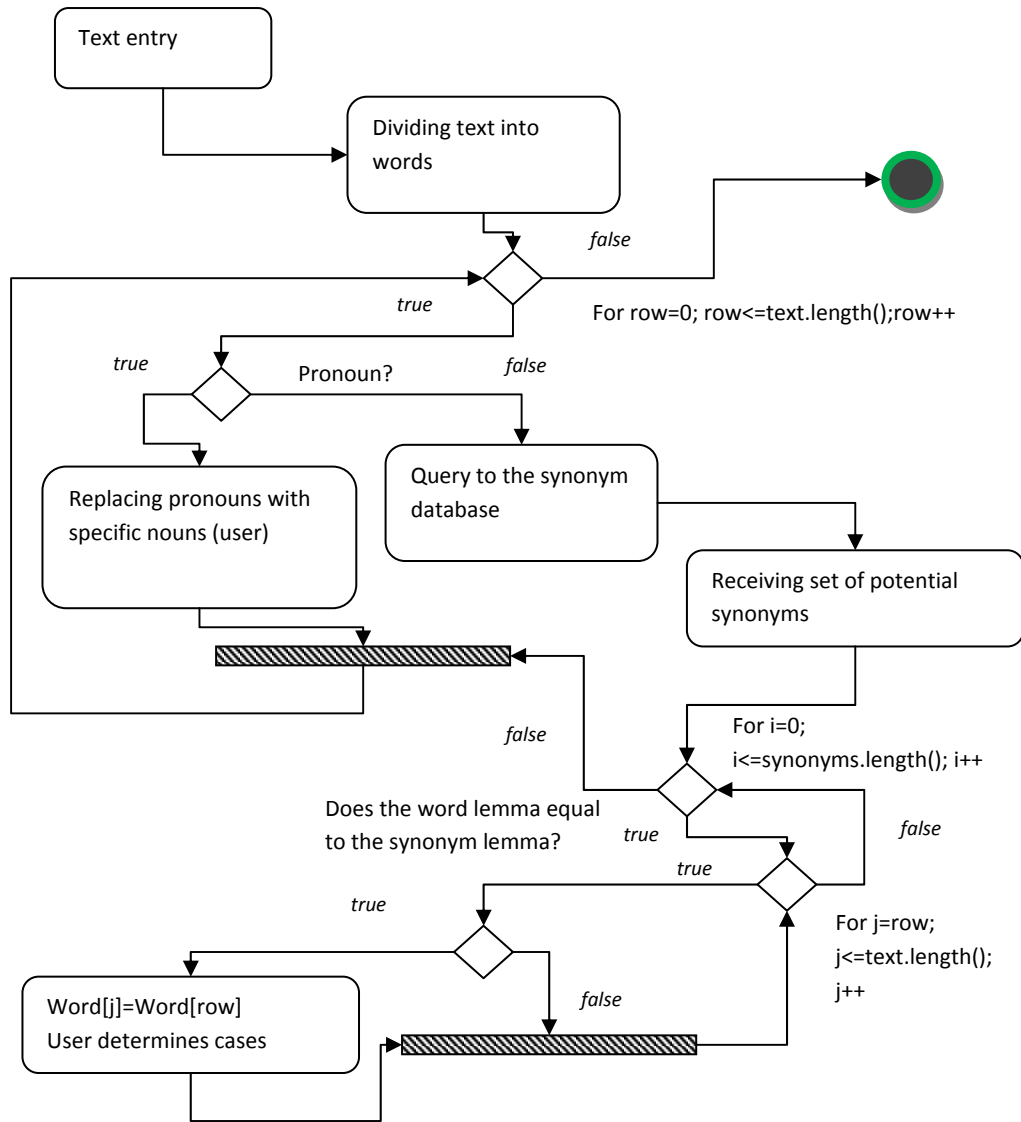


Figure 1. Replacing synonyms in text

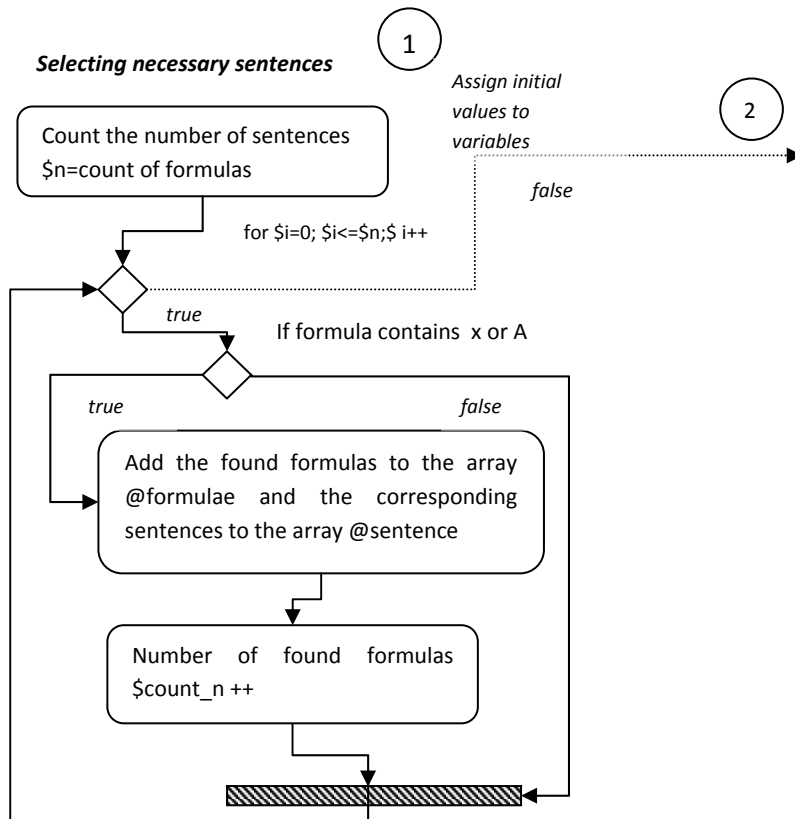


Figure 2a. Algorithm for identifying inference steps. Part I, Selecting necessary sentences

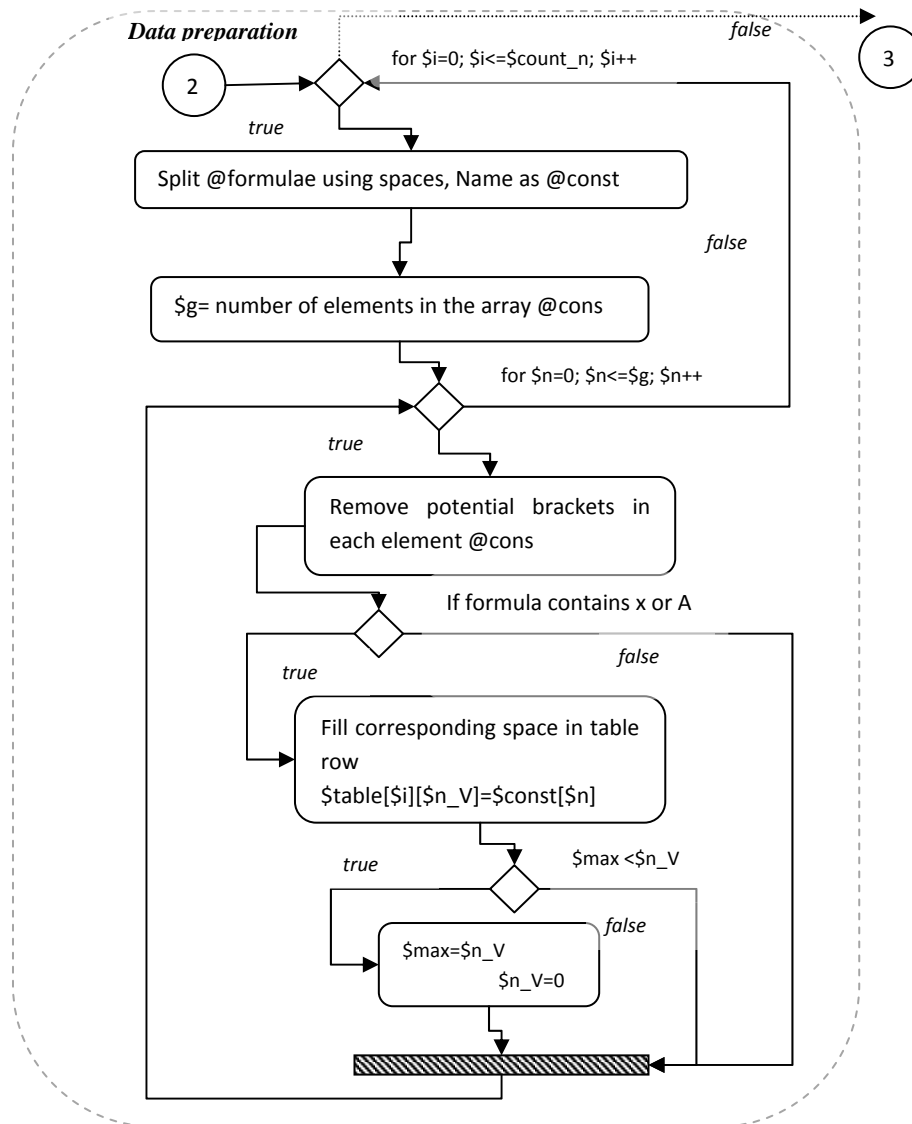


Figure 2b. Algorithm for identifying inference steps. Part II, Data preparation

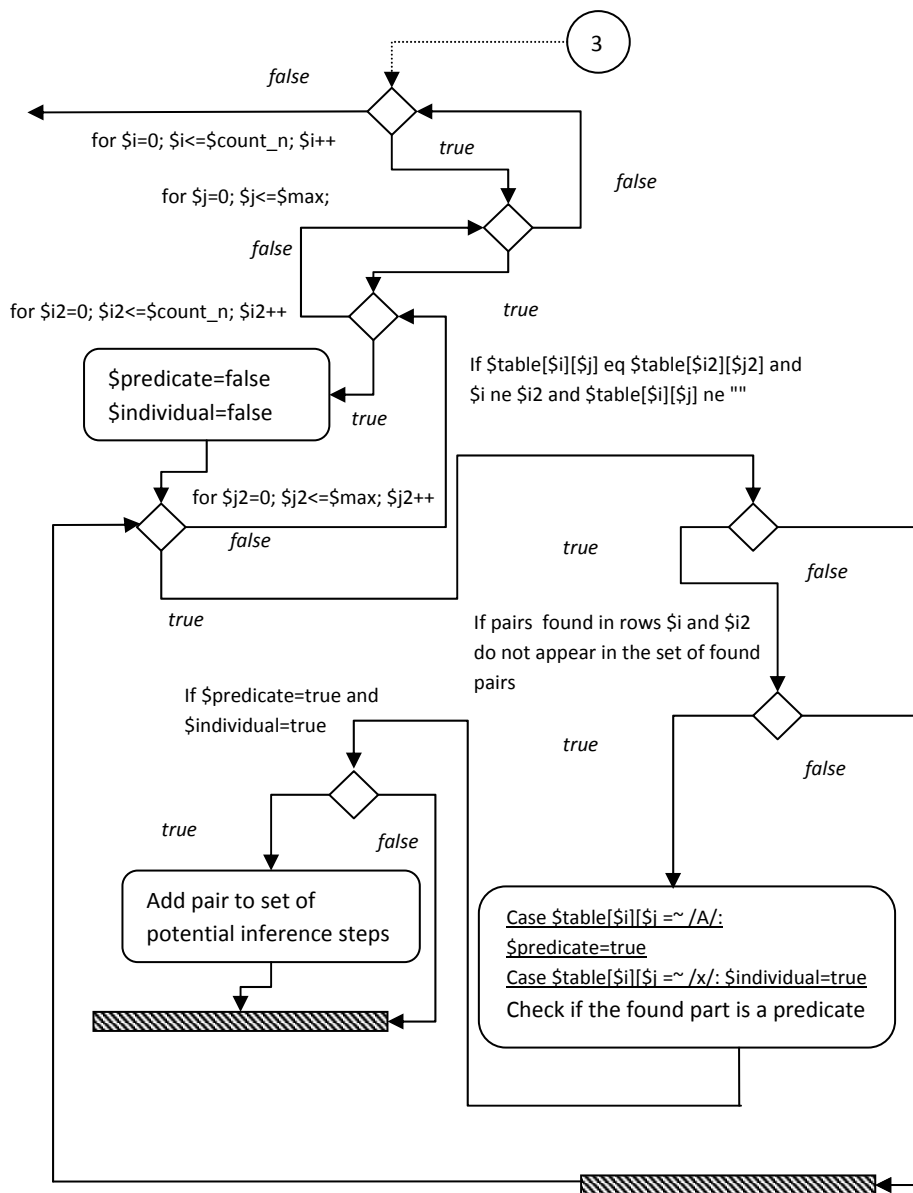


Figure 2c. Algorithm for identifying inference steps. Part III, inference steps

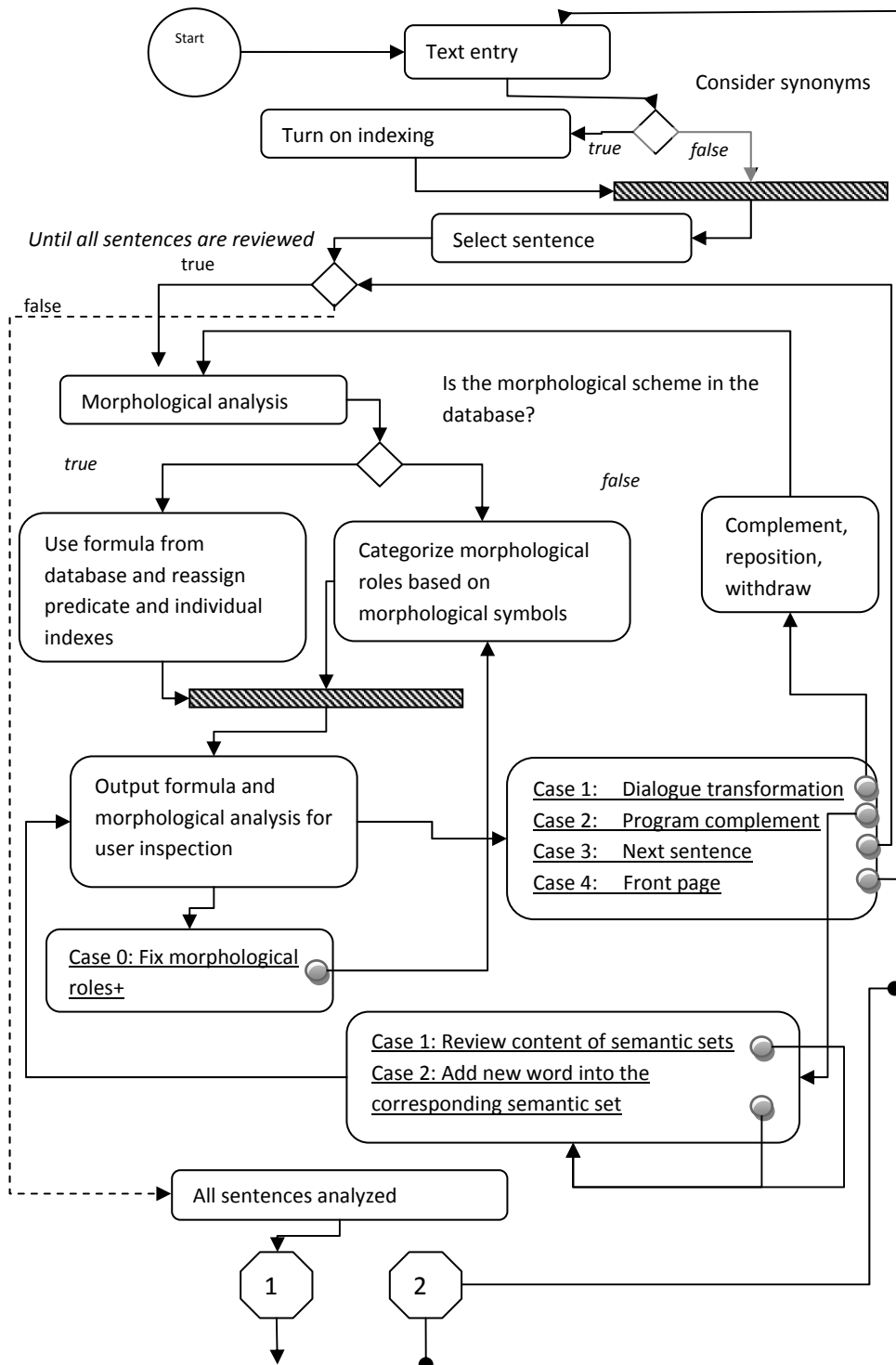
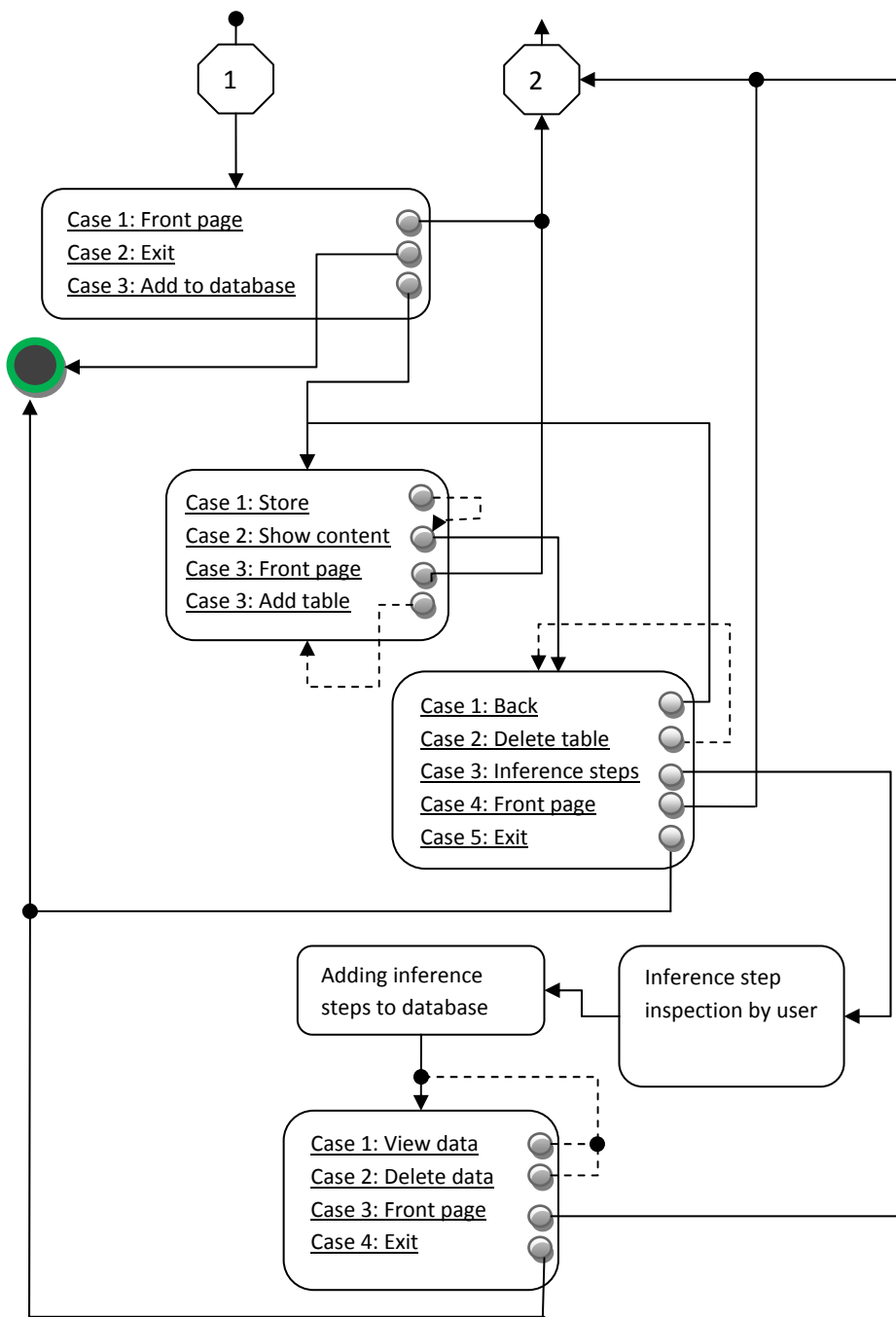


Figure 3. Layout of the latest version of DST



Continue of Figure 3. Layout of the latest version of DST

APPENDIX B: CURRICULUM VITAE

CURRICULUM VITAE **in English**

1. Personal data

Name: **Erika Matsak**

Date and place of birth: **19.04.72, Sankt-Peterburg, Russia**

2. Contact information

Address: **Vormsi 6-45, Tallinn, 13913**

Phone: **+372 56656537**

E-mail: **erika.matsak@tlu.ee**

3. Education

Educational institution	Graduation year	Education (field of study/degree)
St. Petersburg Technology Institute (Technical University) Санкт-Петербургский государственный технологический институт (технический университет)	1997	Automation Engineer (инженер по автоматизации)
Tallinn Pedagogical University	2004	MS in Multimedia and learning systems

4. Language competence/skills (fluent; average, basic skills)

Language	Level
Russian	Native
Estonian	Fluent
English	Fluent
French	Basic skills

5. Special Courses

Period	Educational or other organisation
2000	Economic Accounting (Tallinn Higher School of Commerce)
2000	NT administration (Microsoft Certified Technical Education Center)
2000	"Teachers' Functions" (Merlecons), "Tutoring, Cooperation with Colleagues, pupils and parents", "Teacher in Lead of Learning Process"
2002	Teacher Management in Changing Environment

6. Professional Employment

Period	Organisation	Position
1996–1998	Computer company ERIMEX (Master Distributor Acer, CLR, etc) St. Peterburg.	Advertising, marketing
1998–2002	Kuressaare Gymnasium	Teacher of Informatics. Head of ICT
1998–2002	Training Center OSILIA	Teacher of Informatics
2002–2004	Kuressaare Trade School	Teacher of Informatics
2002–2004	TPÜ Open University In- service Training	Teaching „Subjects in the Computer Class“
2002–2005	Co-ed.Gymnasium of Saaremaa	Teacher of Informatics
2005–2008	Tallinn University	Research worker
2006–...	Tallinn University of Technology	External lecturer
2008–...	Tallinn University	Lecturer

7. Defended theses

Пневмоимпульсное дозирование сыпучих материалов. [Pneumo-impulse dosing of scatter-materials] St. Peterburg Technology Institute

A system for identifying logic constructs in Estonian Language texts. Formula level. Master's thesis, Tallinn Pedagogical University, Tallinn.

8. Main areas of scientific work/Current research topics:

Artificial intelligence, Language formalization, Language technology

9. Other research projects

2005–2007 Concept mapping

2008–2010 Development of language software and language technology resources for the Estonian interlanguage corpus

CURRICULUM VITAE

Eesti keeles

1. Isikuandmed

Ees- ja perekonnanimi: **Erika Matsak**

Sünniaeg ja -koht: **19.04.72, Sankt-Peterburg, Venemaa**

Kodakondsus: **Eesti**

2. Kontaktandmed

Address: **Vormsi 6-45, Tallinn, 13913**

Telefon: **+372 56656537**

E-posti address: **erika.matsak@tlu.ee**

3. Hariduskäik

Õppeasutus (nimetus lõpetamise ajal)	Lõpetamise aeg	Haridus (eriala/kraad)
Sankt-Peterburgi Tehnoloogia Instituut (Tehnika Ülikool) Санкт-Петербургский государственный технологический институт (технический университет)	1997	Automaatika insener (инженер по автоматизации)
Tallinna Pedagoogikaülikool	2004	Magister. Informaatika (multimeedium ja õpisüsteemid)

4. Keelteoskus (alg-, kesk- või kõrgtase)

Keel	Tase
Vene keel	Emakeel
Eesti keel	Vaba
Inglise keel	Vaba
Pransuse keel	Algtase

5. Täiendusõpe

Õppimise aeg	Täiendusõppe läbiviija nimetus
2000	majandusarvestus (Tallinna Kõrgem Kommertskool)
2000	NT administreerimine (Microsoft Certified Technical Education Center)
2000	"Õpetaja põhifunktsioonid" (Merlecons), "Klassijuhataja koostöö kolleegide, õpilaste ja lapsevanematega", "Õpetaja kui õppeprotsessi juht"
2002	"Toimetulek õpetaja rolliga pidevalt muutuv keskkonnas"

6. Teenistuskäik

Töötamise aeg	Tööandja nimetus	Ametikoht
1996–1998	Arvutifirma ERIMEX (Master Distributor Acer, CLR, etc) St. Peterburg.	Reklaam, marketing.
1998–2002	Kuressaare Gümnaasium	Informaatika õpetaja. IT suuna juht
1998–2002	Koolituskeskus OSILIA	Informaatika õpetaja
2002–2004	Kuressaare Ametikool	Informaatika õpetaja
2002–2004	TPÜ Avatud Ülikooli Täiendõppekeskus.	Informaatika õpetaja "Ainetunnid arvutiklassis"
2002–2005	Saaremaa Ühisgümnaasium	Informaatika õpetaja
2005–2008	Tallinna Ülikool	Teadur
2006–...	Tallinna Tehnikaülikool	Välisõppejõud
2008–...	Tallinna Ülikool	Lektor

7. Kaitstud lõputööd

Пневмоимпульсное дозирование сыпучих материалов. Санкт-Петербург
Tehnoloogia Instituut

Eestikeelsetes tekstides sisalduvate loogiliste konstruktsioonide väljaeraldamise
süsteem. Valemite tase. Magistritöö, TPÜ, Tallinn

8. Teadustöö põhisuunad

Tehisintellekt, Keele formaliseerimine, Keele tehnoloogia

9. Teised uurimisprojektid

2005–2007 Concept mapping

2008–2010 VAKO – Eesti vahekeele korpuse keeletarkvara ja
keeletehnilise ressursi arendamine

RESEARCH PAPERS

**DIALOGUE SYSTEM FOR EXTRACTING LOGIC
CONSTRUCTIONS IN NATURAL LANGUAGE TEXTS**

Erika Matsak

Reprinted with permission, from Proceedings of the International Conference on
Artificial Intelligence. IC – AI'2005. Volume II p. 791 – 797 Las Vegas, Nevada, USA

Dialogue System For Extracting Logic Constructions In Natural Language Texts

Erika Matsak
Department of Computer Science
Tallinn University
25 Narva Road, 10120 Tallinn, Estonia
E-mail: erika.matsak@tu.ee

Abstract: *Different areas on intellectual activity and methodology require different logic. One way to extract logical constructions in natural language is to process the presented texts correspondingly. It is possible to apply special transformation procedure of texts, which enables to transform a statement in the natural language source text into a logic formula. We can find justification why such a problem cannot be generally solved by using algorithms (from the aspect of algorithm theory). A corresponding and functioning dialogue system for transforming texts in the Estonian language may be created.*

Keywords. Natural language texts. Logical constructions. Terms and formulas. Transformation procedure of texts. Description of algorithms.

1. Introduction

Research work in human and artificial intellect and its application has brought along the knowledge that different logic is applied in corresponding fields and in certain task categories. For example, intuitive logic is appropriate for the structural synthesis of programs (see [Tyugu 1988], [Mints, Tyugu 1982]), the synthesis of programs with sub-tasks is related to the modal logic system S4. For describing and analyzing the time-limited interactive process systems we find the classical – the so-called weak secondary predicative calculation appropriate (see [Lorents, Motus, Tekko 1986] and [Lorents, Motus 1986]).

The choice of the logic means in the above described examples are basically based on one and the same phase, during which we agreed what kind of natural language texts to use and after that we found the formal equivalents to

these texts (and as it turned out later) within one or some other kind of logic.

The aim of this paper is to research this phase, during which the natural language texts obtain a strict presentation within one or some other type of logic frame. Hereinafter we agree to confide ourselves to the so-called formula level and not to consider the deduction device/contrivance/apparatus.

2. Lorents's Procedure Of Texts Transformation

Next we would view the procedure, first described and applied in the book by P. Lorents (see Lorents 2000)]. The procedure is divided into seven phases, which application sequence and number is unregulated. The basic requirement for performing each phase is to guarantee the same meaning or thought of the source text and the final text. During the mentioned phases the following processes practically take place in the parts of the texts:

- *relocating*
- *supplementing* (or adding new parts)
- *reducing* (or removing the parts)
- *replacing* (or replacing some part by a new text)
- *extracting symbols* (or extracting the parts of texts, which have a role of some logic alphabet symbol)
- *categorizing the symbols* (specifying the role of extracted parts of the text e.g. if we deal with a symbol identifying an object, the object's characteristic feature or the relationship between objects or even a symbol identifying a logic operation, etc.)
- *positioning symbols* (placing symbols in their appropriate places in logical contractions).

Examples:

- “Are together.” → Supplementing →
“We are together” → $P_1(q_1)$,
where P_1 – are together, q_1 – we.
- “Yes, the cat eats fish.” → Reducing →
“The cat eats fish” →
 $P_1(q_1, q_2)$, where P_1 – eating, q_1 – cat, q_2 – fish.
- “Finally the kitten has found a friend” →
Supplementing → “Before the kitten did not have a friend, but now the kitten has finally found a friend.” → Replacing →
“Before the kitten did not have a friend, but now the kitten has found a friend.” →
 $\neg A_1(x_1, x_2, t_0) \& A_1(x_1, x_2, t_1)$, where A_1 – have, x_1 – kitten, x_2 – friend, t_0 – at first time moment (before), t_1 – at final time moment (now).

As mentioned above, in every phase it is required that the meaning of the initial text and final texts at the end of the phase had the same meaning. And it is the transformers decision to decide if it is so or not. Consequently we may state that transformation procedure cannot be fully/completely automatic which at the same time does not exclude the *possibility of corresponding dialogue system* in itself. The necessity of dialogue is relevant also because of certain algorithmic-theoretical aspects, which we would like to characterize briefly in the following part.

Namely, natural language texts are formations consisting of symbols (words, expressions, etc.). If we supplement the natural language texts with mathematical-logical symbols, we actually do not change the situation. We still have a certain construction, written down in some alphabet. We may view the transformation phases of texts as changes in some *replacement system* as for example in the so-called Thue system (see e.g. [Maltsev 1986], chapter VI, §14.1). We certainly know that even in case of a “very modest” replacement system the problem of replacing words with equal equivalents has no algorithmical solution. So we need an “oracle” (see e.g. [Rogers H. 1976]) or a user working with a program in the dialogue mode.

3. General Characterization Of The Dialogue System

The observed dialogue system was possible thanks to the software created by the company FILOSOFT “HTML morph analyzer of the Estonian language” and “The synthesizer of the Estonian language”, located at www.filosoft.ee and enable to get morphological signs of words and vica versa – formation of words in required morphological form using the basic form.

At this point we may point out that basically the above-mentioned dialogue system algorithm can be applied also in other languages, which have corresponding linguistic programs, enabling to perform automatic inquiry and getting feedback about the morphological types and programs, which enable synthesis. Besides it is necessary to carry out a research to establish the connections between the logical roles and morphological classes in a certain language.

In the discussed dialogue system we have two modes *to transform* the sentences. The first mode so to say is teaching the program. The user transforms the sentence phase after phase and writes the intermediate variants of the text into the text fields. Each sentence is analyzed morphologically and the outcome is a saved *scheme of morphological signs*. In each sentence the change of word order and when adding new words the morphological form of the sign (the order of the scheme) is memorized.

Example:

We teach transformation to the program with the following sentence “I play with a beautiful doll”. In the learning process the following *scheme of morphological signs* is created:

= $_PP\#\#sg\#\#I\#_V_\#\#pr\#\#\#s\#__Pre$
 $_A\#\#i\#_Adj\#\#sin\#_N\#\#sin\#_$
 $_PP\#\#sg\#\#\#I\#_V_\#\#pr\#\#\#s\#__Pre_A\#\#i\#_$
 $_N\#\#sin\#_new(\&)_new(_A\#\#d\#\#_N\#\#sin\#_$
 $new(_V_\#\#pr\#\#\#s\#\!)_Adj\#\#sin\#_ =$

(Morphological signs are explained in the table below)

Table 1

Morphological signs	Explanation
<u>PP</u> + <u>#sg</u> + <u>#I</u> #	Pronoun, single, first form
<u>V</u> + <u>#pr</u> #+ <u>#s</u> #	Verb, present, single
<u>Pre</u>	Preposition
<u>Pre</u> <u>A</u> + <u>#i</u> #	Article, individual
<u>Adj</u> + <u>#sin</u> #	Adjective, single
<u>N</u> + <u>#sin</u> #	Noun, single
<u>new</u> (<u>&</u>)	New word - and
<u>new</u> (<u>A</u> + <u>#d</u> #)	New word: article - the
<u>new</u> (<u>_V</u> + <u>#pr</u> #+ <u>#s</u> #!)	New word: verb, present, single, "are" verb form - is

and the scheme of the order:

0 1 2 3 4 5 =0 1 2 3 5 new(&) new (_A+ #i#) 5
new(_V+#pr#+#s#!) 4

In the initial sentence the words are enumerated as follows: 0 (I), 1 (play), 2 (with), 3 (a), 4 (beautiful), 5 (doll). In the transformed sentence the order of words is changed, e.g. in the fifth position there should be a word with number 5 (doll), not 4 (beautiful). In addition three new words are created:

new & (and), new _A+#d# (the) and new _V+#pr#+#s#! (is).

The comparison of words is organized with the help of initial words and via the way of creating the *ordering scheme* and it is not important, which is the form of the word in the first sentence and which is the form in the transformed sentence.

The second mode is the automatic transformation according to the corresponding *morphological scheme* found prior. The mode is based on principle that the morphological schemes are equal and we obtain the results in the formal image of equal logical constructions.

Examples:

"I play with a beautiful doll" → "I play with a doll and the doll is beautiful" → $P_1(q_1, x_1) \& P_2(x_1)$.

"I talk with a new friend" → "I talk with a friend and the friend is new" → $P_1(q_1, x_1) \& P_2(x_1)$.

"At my home I have an aquarium, where fish swim" → "The home is mine and at home I have an aquarium and fish swim in the aquarium" → $P_1(x_1) \& P_2(x_1, x_2) \& P_3(x_1, q_1)$.

"At my school I have an auditorium, where teachers give presentation" → "The school is mine and at school I have an auditorium and teachers give presentation in the auditorium." → $P_1(x_1) \& P_2(x_1, x_2) \& P_3(x_1, q_1)$.

4. Description Of The Algorithm in general

index.html. Authorization (users need special codes)

index.pl. Text typing. Sentences must be typed grammatically correctly and at the end of sentences there must be a period.

e1.pl. Errors controls. Sentences and words separated from the text. Truth-value existence check in sentences (if the sentence including the verb is not in the form of imperative or question, the software defines the truth-value existence).

If the user disagree of the software choice, he can change it to the opposite. All sentences, which truth-value existence is not present, are extruded from the next analyze.

e2.pl. Sentence analyze, which includes morphological analyze and logical operators searching.

All necessary synonyms for each group of operators or symbols are kept in corresponding files, written and saved in accordance with special research.

Detecting time symbols is divided into two main blocks. First (tense keyword search) is implemented in e2.pl, second (tense form analysis of the verb) in e3.pl.

The user must confirm software-processing choices. As result script creates morphological scheme of sentence.

The software detects the next operators and symbols (table 2):

Table 2

denotate	Some words examples	sign
Negotiation	no, not, false, wrong, incorrect, improper	\neg
Conjunction	and	$\&$
Disjunction	or	\vee
Implication	then	\supset
Equivalence	same, equivalent	\Leftrightarrow
Universal quantifier	all, any, anyone, anybody, every, everything, everybody	\forall
Existential quantifier	exist, existent, existing	\exists
Modals	always, certainly, surely	\square
	maybe, possible, possibly, probable	\diamond
Time symbols	tomorrow, after, afterward, later, now, current, earlier, sooner, previously, yesterday, previous	t_1, t_2, t_i, t_{i+n}

e3.pl. Sentence part extracting (an according to logical operators existence). Logical roles definition. Formula creation.

Individuals (in Estonian language) automatically separated by the software are: nouns, pronouns, proper nouns, adverbs. Predicates are described in table 3 below.

Table 3. The Estonian language predicates automatically separated by the software

1. adjective	5. two or more verbs as one (example: are going)
2. verb	
3. superlative	6. verb "are" forms+ noun in nominative (example: is teacher)
4. verb "are" forms+ adjective (example: are blue)	7. verb "are" forms+ superlative (example: will best)

All of individuals and predicates may be presented in the sentence for a few times. A special module separates such words and the index correctly in the formula.

Individual are marked as q_1, q_2, q_i, q_{i+1} , and in case of any repetition of some individual they are marked as $-x_1, x_2, x_i, x_{i+n}$.

Drawing 1. Formula creation.

Eestikeelsetes tekstides sisalduvate loogiliste konstruktsioonide väljaerdamise süsteem *Valemite tase.*

Lause nr. 1. Kokku lauseid: 1

Võib-olla homme on ilus ilm

Sinu poolt valitud lause skeem:

$\diamond P1(q1 t0)$

Edasi Esilehele Transformeeri Välju programmist

Üles

Lauseosa nr 1

Võib-olla

homme

Sündmus toimub ajal **t0** Jah Ei

Kas oled nõus, et **on ilus** on selles lauses predikaadi rollis? Jah Ei

Kas oled nõus, et **ilm** on selles lauses indiviidi rollis? Jah Ei

Alla

Predicates are marked as P_1, P_2, P_i, P_{i+n} and in case of any repetition of some predicate then – A_1, A_2, A_i, A_{i+n}

The script also analyzes verb tense forms for time symbols excluding. If the sentence consists of two or more parts and each part includes verbs or verb combination, the software compares those forms. If they are not equal (the meaning of the tense), then time symbols indexation.

At the screenshot (drawing 1), the software creates the formula for the sentence: “*Maybe the weather will be beautiful tomorrow*”.

In the previous script it was defined that *maybe* is modal, *tomorrow* is time symbol.

Here the user must confirm, that:

- This event will happened at moment t_0
- *Will be beautiful* is a predicate
- *Weather* is an individual

If the software suggestion is not right, the user can choose the logical role manually.

If sentence needs transformation, the users may use two kinds of modes.

transform.pl. This script checks the saved schemes and advice of the transformation mode.

If the scheme is found, the dialog-system shows to user the initial sentence, which has been used for software teaching. It is also possible to ignore the suggestion of transformation mode and teach step by step.

study.pl. This mode teaches the dialog-system step by step.

The user types each transformation step as a sentence and the script executes the morphologic analysis. The software saves all steps in necessary files.

Studying is protected by password. It is very important that the person, who teaches software, knows enough of sentence transformation.

v1.pl. Reports of transformation and necessary schemes creation (*morphological scheme* and *order scheme*).

Drawing 2. Automatic Transformation.

Initial sentence: “I drive by a new car”. After transformation we have: “I drive by car and the car is new”

Eestikeelsetes tekstides sisalduvate loogiliste konstruktsioonide
väljaeraldamise süsteem *Valemite tase.*

Lause nr. 1. Kokku lauseid on 45

Mina sõidan uue autoga.

Automaatne transformeerimine:

Edasi Välju programmist

Ülesse

Transformeerimise samm nr 1:
Moodustatud lause:
mina sõidan autoga ja auto on uus

ma
 mina
 mina sõidan autoga

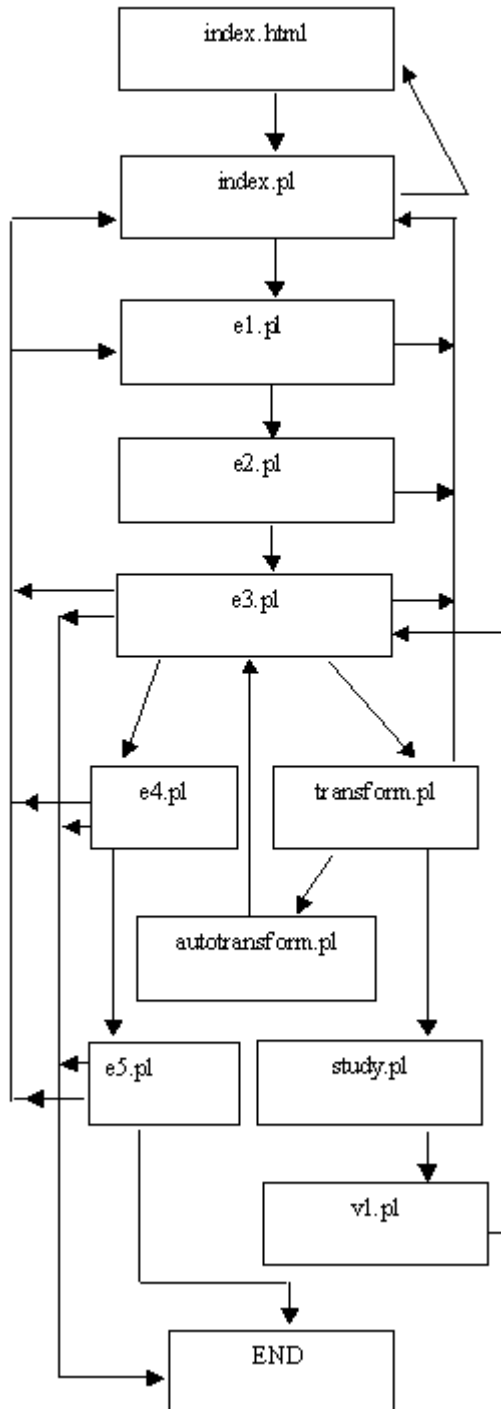
ja auto on uus

Alla

Next mode gives an opportunity to use transformation automatically.

autotransform.pl This script uses “The synthesizer of the Estonian language” and gives words in requested forms according with *morphological scheme* and *order scheme* saved before.

Drawing 3



The dialog-system only shows the initial and final sentence, all other steps (if exist) the script executes automatically.

The user must confirm some words in the transformation result sentence (synthesizer gives a few output for any word).

Each word is displayed in the textbox and the user, if necessary, can correct the words. As a matter of fact, testing has proved that such necessity occurs very seldom.

e4.pl. Text and formula addition to the user database.

It is possible to select the date of the sentence transformation and use two databases: personal and research.

The database may be copied as a table in the Microsoft Office and other programs (for example MS Excel, SPSS) for the next analysis or for creating a report.

e5.pl. Shows all sentences typed before, in index.pl. User may select one of them and go back to index.pl. It is also possible to delete sentences or finish work.

Software interruption as an exit button is added to most of scripts (except for *study.pl*, *v1.pl*), but formula creation finishes in *e3.pl*.

The general scheme of connections between scripts is shown in drawing 3.

About the software

The program is realized via *perl* and *javascript*. If you want to use the program the computer must have the installed Microsoft Internet Explorer beginning with version 6 and with Java support.

As far as we open the program in a new pop-up window, the installed firewall must allow it.

For using logical signs/markers you need the support of *charset=iso-10646-1*.

About the hardware

The computer has to be connected to the Internet. The monitor resolution – at least 800X600, processor – at least 400MHz, memory – at least 64MB, speed of the network (or the modem) – at least 56KB/sek.

The software may be used also with lower parametric computer, but in this case the process will be slower and the results may be displeasing.

Pluses/Advantages

The described dialog-system is created first and foremost for logical research, but it may be applied in higher education, if students study the human language transformation into logical formula.

- Without software, transformation is going to use many time recourses.
- Without software, people use in transformation procedure its imagination and formula result may be different for the same sentence.

Using the dialog-system makes the transformation process faster (the user must only confirm all necessities steps). If the sentence is serious or abstruse, then the high-knowledge users teach the software and hereafter automatically solutions are suggested.

5. Summary

Logical thinking is one of the attributes of human intellect. It plays an important role in the formation of people and education processes. It is necessary to point out that teaching different constructions to an inappropriate age group will probably prove to be ineffective.

The most important sphere of life where the logic manifests itself is the usage of human language.

Therefore this dialog-system, which transforms the Estonian language sentences into logic constructions, gives us the possibility to

analyze human thinking and adjust literature for educational purposes.

There are very many languages in the world and the language groups differ a lot from each other, but we can observe certain structural similarities in them. Thus, at some point of time we may be able to find a universal formal language, which would present logic constructions from every language.

Reference

1. Lorents P., Motus L., Tekko J. 1986. *A language and calculus for distributed computer control systems description and analysis*. 4th IFAC/IFIP Symposium on Software for Computer Control, Graz, Austria May 20 – 23.
2. Lorents, P. , Motus, L. 1986. *Logical tools to describe and analyze the interactive system of processes with prescribed time limiters*. IV conference of the USSR “Application of methods of mathematical logic.” Theses of the presentations. Institute of Cybernetics of Estonian Academy of Science. Tallinn.
3. Lorents P. 2000. *The language and the logic*. EBS-PRINT. Tallinn.
4. Maltsev A. I. 1986. *Algorithms and recursive functions*. “Science”. Moscow.
5. Mints G., Tyugu E. 1982. *Justification of the structural synthesis of programs*. Science of computer programming 2(3), 215-240
6. Mints G., Tyugu E. 1987. *The programming system PRIZ*. Journal of Symbolic Computation, (4).
7. Rogers H. 1967. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill Book Company. New York. St Louis. San Francisco. Toronto. London. Sydney.
8. Tyugu E. 1988. *Knowledge - Based Programming*. Addison-Wesley.

**USING NATURAL LANGUAGE DIALOG SYSTEM DST
FOR DISCOVERY OF LOGICAL CONSTRUCTIONS OF
CHILDREN'S SPEECH**

Erika Matsak

Reprinted with permission, from Proceedings of the International Conference on
Artificial Intelligence. IC-AI 2006. Volume I p. 325 – 331 Las Vegas, Nevada, USA

USING NATURAL LANGUAGE DIALOG SYSTEM DST FOR DISCOVERY OF LOGICAL CONSTRUCTIONS OF CHILDREN'S SPEECH

Erika Matsak

Faculty of Information Technology

Tallinn University of Technology

15 Raja, 12617 Tallinn, Estonia

E-mail: erika.matsak@tlu.ee

Abstract: Logical constructions in speech of 4-6 years old children are investigated by using the dialog system DST for transformation of the natural language texts. The research has brought up the point, that children are able to operate both on the first-level (predicate) calculus, and higher-level predicate calculus – that means formulas creation. It is remarkable how the constructs of making conclusions are used in texts: they occur in representation of inference, ordering of time moments, and in the role of conjunction and disjunction operator.

Keywords: dialog system for extracting logical constructions from natural language text, high-level logic constructions, modalities, gradation for assessment as structures in the 4-6 years old children texts.

Introduction:

The present paper describes shortly the results of implementation of the dialog system for extracting the logical constructions from natural language text, created by author.

[Matsak 2005]

The aim of system creation and development is infotechnological assistance for researches, which helps to understand the formation of logical means and implementation in children's evolution.

The similar research has been planned by prof. Lorents at about 15 years ago with scientists of Estonian Academy of

Science, Cybernetic Institute, Technical University of Tallinn and University of Tartu. Unfortunately, the project was stopped because of financial problems. And now, years later, the mentioned research is continued by Tallinn University, Technical University of Tallinn and Estonian Business School.

The one of the main phase is the collection of children text and transformation in a form, which is able to bring up the logical constructions: primarily logical formulas and inference rules.

The dictaphone recording of children speech has been done thanks to student's help, who had a practise in pre-school institutions. The speech has been typed into computer as text and transformed by DST system into logical formulas.

Because of organisational reasons, it has been decided, at first to collect and analyse the 4-6 year old children speech-texts. The initial results are described in this paper.

The analysis of children's texts indicates the existence of quite complicated logical constructions. On the formulas level it is as follow:

- Children use all components of first-level predicate calculus, including:
 - sorts of individuals
 - time moment and time related symbols
 - unary and n-ary predicates
 - all logical operators (\neg , $\&$, \vee , \supset)
 - quantifiers (\exists , \forall)
 - modalities (\diamond , \square)

Remark: operator of equivalence \Leftrightarrow did not appear in texts.

- Children use linguistic constructions for implication for the following events:

- for logical implication
- for ordering of time moments

-in the role of conjunction and disjunction

- Children use constructions related to higher-level (at least 2-level) logic
- Children operate with many different gradations, applying the second-level predicates and functionals.

All findings mentioned above brought up the necessity to improve the DST dialog system by adding new modules.

1. The dialog system for extracting logical constructions in natural language text.

1.1 Lorents's Procedure of Text Transformation

The procedure of text transformation was first described and applied in the book by P. Lorents (see Lorents 2000)]. It is divided into seven phases, where the order of usage and number of steps is inessential.

The main requirement for performing each phase is to save the meaning of the source text in the final text. The steps of the procedure are as follows:

- *relocating*
- *supplementing* (or adding new parts)
- *reducing* (or removing the parts)
- *replacing* (or replacing some part by a new text)
- *extracting symbols* (or extracting the parts of texts, which have a role of some logic alphabet symbol)
- *categorizing the symbols* (specifying the role of extracted parts of the text e.g. if we deal with a symbol identifying an object, the object's characteristic feature or the relationship between objects or even a symbol identifying a logic operation, etc.)
- *positioning symbols* (placing symbols in their appropriate places in logical constructions).

Examples:

Maybe we play something interesting. → *supplementing* → Somebody may play

something interesting. → *replacing* → Exist a person, who may play something interesting. → *supplementing* → Exist a person, who may play something and this something is interesting. → *positioning* → Exist a person and exist something, the person may play and this something is interesting. → *replacing* → Exist a person and exist a thing, the person may play a thing and the thing is interesting. → *categorizing the symbols* →

$(\exists x_1) (\exists x_2) P_1(x_1 x_2) \& P_2(x_2)$

1.2. The DST dialog system

The dialog system for extracting the logical constructions from Estonian text uses the HTML morph analyzer and the synthesizer of the Estonian language, located at <http://www.filosoft.ee>. Dialog system automatically executes the query to Filosoft company software for receiving necessary morphological signs and visa versa for creation of words in required morphological forms using the basic form. The discussed dialogue system has two modes *to transform* the sentences. *The first mode* is teaching the program. The user transforms a sentence phase after phase and writes the intermediate variants of the text into the text fields. Each sentence is analyzed morphologically and the outcome is a saved *scheme of morphological signs*. The change of word order and adding new words and the morphological form of the sign (the order of the scheme) in each sentence is memorized. *The second mode* is the automatic transformation according to the corresponding *morphological scheme* found prior. The mode is based on principle that the equal morphological schemes give the equal logical constructions.

2. Detected logical constructions

2.1 Meanings and roles of "then"

The traditional role of "then" is logical implication. Whereby, one of the "pure" roles is the possibility to represent "if ... then" sentence as "from ... follows that".

Examples *:

If you want to get money back, then you get it.

(*Kui tahad raha tagasi saada, siis saad*)

$P_1(x_1, x_2) \supset P_2(x, x_2)$

If this dinosaur comes, then it eats you.

(*Kui see dinosaur tuleb, siis ta pistab sind kõhtu*)

$P_1(x_1) \supset P_2(x_1, x_2, x_3)$

All examples in this paper are created from children's text. The age group of children is 4-6 year old.

If he does not come to visit me, then I can't marry him.

(*Kui ta ei tule mulle külla, siis ma ei saagi temaga abielluda*)

$\neg P1(x1, x2, x2) \supset \neg P2(x1, x2)$

The second role of “then” is a time instance allocation, which may be present as a simple moment and as several moments, which are ordered in more complicate sentences. If the premiss does not have a predicate, the role and meaning of “then” is a time moment of the action. In a situation, when a sentence with “then” includes more than two parts with own predicates, there will be several time instances, that are ordered according to sentence grammatical scheme. If the number of predicates is more than the number of time moments, it means, that some parts of sentence describe the situations, that have happened at the same moment.

Examples:

When nobody wants to listen.

(*Siis, kui mitte keegi ei taha kuulata*)

$\neg(\exists x)P(x, t)$

And then we went to cow-house by bicycle and then gave them to eat and then caressed them.

(*Ja siis me läksime lehma lauta rattaga ja siis andsime neile süüa, ja siis tegime pai neile.*)

$P1(x1, x2, x3, t1) \& P2(x1, x4, x5, t2) \& P3(x1, x4, x6, t3)$

Third role of “then” could be defined by logical operator, which is positioned before, it may be present as conjunction or disjunction.

Example:

I like to play and then to sleep, sleeping is very nice.*

* We changed the verbs, which answer to the question “what is doing” by combination of nouns with verb “commit”. For example, “I like to sleep” is replaced by “I like commit to the sleeping”.

(*Meeldib mängida ja siis magada, hästi mõnus on magada*)

$P1(x1, x2) \& P1(x1, x3) \& P2(x3) \& [Val(x3) = \epsilon]$

2.2 The second level predicate calculation in children's text

The children's text transformation experience showed the appearance of the second level predicate calculation. The situation when predicates may be used as variables, belong to the second level predicate calculation.

In particular, children in the age group of 4-6 year use many kinds of assessment for individuals, predicates and time amount.

For example, in the sentence “*The sleeping is very pleasant*”, the text transformation gives: *The sleeping is pleasant and the value of pleasant is “very”*. $\rightarrow P(x) \& [Val(P(x)) = \epsilon]$. Here we use the symbol of binary relation \int for categorization the text as notation and denotation presented by P. Lorents in 2001. We read $A \int B$ as the meaning of A is B, or A is a name or a symbol for B.

$x \int$ sleeping

$P \int$ is pleasant

$Val(P(x)) = \epsilon \int$ value of pleasant is “very” (see also p.2.2)

Remark. Probably here would be conventional to use “very pleasant” instead of “very” as the assessment. In this case we need to consider such things as, for example “a little bit sweet”, “quite salty”, “too hot” etc. It is more simple to use as assessment “A little bit”, “quite”, “too” etc., which is possible to implement for predicates and formulas. For instance, the “sweet is good” ($Val(P) = \epsilon$) or “it is good, that cake is sweet” ($Val(P(x)) = \epsilon$)

The logical constructions, which children use include the assessments of predicates.

For example in the sentences “He can not walk at all (*Ta ei saa üldse käia.*)” or “I want to fly a little (*Ma tahan natukene lennata.*)” it is clear, that the truth-value calculation depends not only on construction with predicates {“walk”, “fly”}, individuals {“ he”, “I”} and logical operator “ \neg ”, which belong to first level calculation alphabet. It is also necessary to understand what roles play such elements like:

{“at all (of walk), “a little (of want)”}. It is brought out the predicate type's relation, where the variable is another predicate.

It may be necessary to use a second-order predicate, which is denoted by Val and which means the assessments process. [See also 3.]

The sentences, which include such part as “really do, really have etc”, have special semantics. It means that some argument is controlled and the result is “true”. A second-level predicate may be used with sign \vDash , which means, that statement interpretation is true.

$\vDash(P(x)) \int \varphi(P(x))=1$, where the φ is the formula interpretation.

Examples:

It is really my star. (*See ongi minu täht.*)
 $\vDash(P(x))$

I have not found a name for them yet, really. (*Ma ei olegi neile veel nime välja mõelnud*)
 $\neg [\vDash(P(x1, x2))]$

2.3 Modalities in the children text

Modality symbols point at states, whose probability is “definitely” or “may-be”.

Examples:

Everything is red, maybe. (*Kõik on punane võib-olla.*) $\diamond[(\forall x)P(x)]$

Maybe these cars parked at front. (*Võib-olla need autod parkisid ette*) $\diamond P(x1, x2)$

But this is yours, perhaps. (*Aga see on sinu oma, vist*) $\diamond P(x)$

More difficult logical constructions are associated with semantic of words “can” and “may”.

Here is one of possibility to interpret the sentences like “I can do something” and “I may do something”:

- The statement that somebody can do something means that may-be is the possibility to do something and it is real:
 $\vDash[\diamond P(x1, x2)]$

Example: I may fall down (*Ma võin alla kukkuda*)

- The statement that somebody can do something means that if somebody asks to do something then this really will happen, if somebody really performs this.

$\vDash[\diamond P(x1, x2, t)]$

Example: But I can read only with eyes (*Aga mina võin ainult silmadega lugeda*)

2.4. About having and belonging

There are different ways for understanding the words, that mean “be possessed by”. For example the sentence “This is doctor's article” could be transformed to “This article belongs to doctor”. In the first case, the formula can be written as $P(x1, x2)$, where

$P \int$ be doctor's thing

$x1 \int$ this

$x2 \int$ article

The second case could be $(\exists x)(\exists H)(x \in H)$, where

$x \int$ article

$H \int$ the set of doctor's things

It is difficult to create the formulas for such sentences as “I take your dog away (*Ma võtan sinu koera ära*)” . \rightarrow At the first moment the dog belongs to you and in the next moment I commit the getting process and in the third moment the dog does not belong to you:

$(\exists x1)(\exists H)[(x \in H, t1) \& P(x1, x2) \& \neg(x \in H, t2)]$

3. The assessment gradation

3. Structures for assessments

The structure $\langle H;G \rangle$ is the gradation of assessment if it has been agreed to call the elements of set H as *assessments* and it is decided to use the *relations* of G between the assessments and theirs properties.

[P. Lorents 2005]

Assessment is a process, where the aim is to assign a value to a “thing”.

3.2 Values of property, quantity, activity and time amount

The values of property, quantity, activity and time amount can be graphically represented as a partially-ordered structure, where some element cannot be ordered in

principal without additional parameters. For example, it is impossible to decide whether “a little” is more than “not much” if it is not defined earlier (drawing 1).

The illustrated graph uses the words extracted from the children texts, and all synonyms are not shown.

The dialog system denotes the values of property, quantity, activity and time amount as ϵ and the assessment process as $Val(P(x))=\epsilon$ in predicate case, $Val(x)=\epsilon$ in quantity and $Val(t)=\epsilon$ in time amount case.

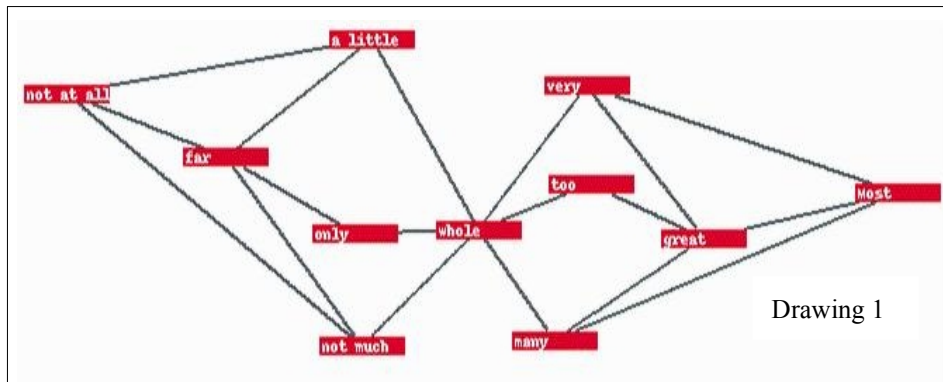
by concrete value. Whereat the structure of time values, like the structure of values of assessments is partially-ordered (drawing 2). That’s why it is impossible always to say, whether an event happened before or later.

The dialog system denotes such values as τ and the process, when somebody defines this value, as $Val(t)=\tau$.

Example:

I had lately the birthday. (*Mul äsja oli sünnipäev*)→ Exist a time moment, when I had birthday and the value of time moment is lately.

$(\exists t)P(x1, x2, t) \& [Val(t)=\tau]$, where



$x1 \int I$
 $x2 \int \text{birthday}$
 $P \int \text{had}$
 $t \int \text{the time moment}$
 $\tau \int \text{lately}$
 $Val(t)=\tau \int \text{the value of time moment is lately.}$

3.3 Values of time

The time symbols are used in the following events:

- If the sentence consists of two or more parts, where verbs are in different time forms.
- If time-related words are used, like tomorrow, now, later etc.

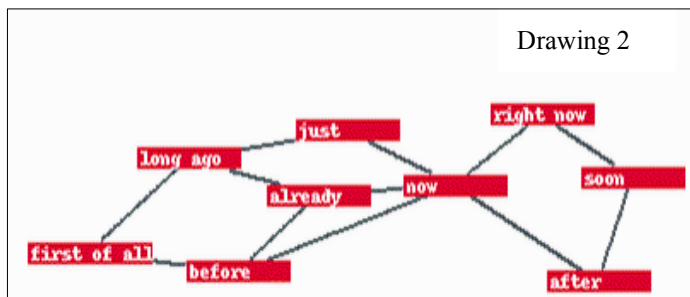
Drawing 3 shows how the DST system detects the morphological construction of this sentence. User must conform all suggested forms. Red font draws the attention to the partially-ordered time symbol in this example.

4. New developed modules of DST dialog system

Module e2.pl. Sentence analysis, that includes morphological analysis and searching of logical operators.

In accordance with the main scheme, the user must confirm software-processing choices and as the result, the script creates morphological scheme of sentence.

A new module has been created for extracting values, which uses saved files with synonyms for *measure values*, which includes the words, collected from children texts. There are two groups of values. The first includes the *values of property, quality, activity and time*



intervals. The second presents the *values of time amount.* If an existence of a value is detected, user has to choose also the type of variable: individual, predicate or time amount.

- If the sentence includes the partially-ordered time moments.

When somebody specifies the time moment by time-related word, it may be interpreted as time moment determined

Eestikeelsetes tekstides sisalduvate loogiliste konstruktsioonide väljaeraldamise süsteem *Valemite tase.*

Lause nr: 1. Kokku lauseid: 1

Mul äsja oli sünnipäev.

Sinu poolt valitud lause skeem:

`_P_+sg+ad#H(t)=e_V_+#s#!_S_+sg+#n#`

Edasi

Esilehele

Välju programmist

Programmi täiendus

Üles

Sõna nr 1: Mul

Morf. analüüs:

* **asesõna (pronomeni), nt see**

ainsus: adessiiv alalütlev

Sõna nr 2: äsja **Kas oled nõus, et äsja on selles lauses osalise järjestusega aja sümboli rollis?**

Jah

Morf. analüüs:

* **määrsõna (adverb), nt kõrvuti**

Drawing 3

Alla

Values of property, quantity, activity and time amount	Values of time intervals
Most, a little, a bit, a few, not much, not many, very, ever, so, only, whole, many, too, great	Anymore, now, first of all, first off, never, ever, sometime, immediately, after, yet, just, before, not ever, long ago, right now, currently, presently, soon, soon enough, early, at first, in the beginning, already

The module extracting time symbols identified items only in ordered gradation in the previous version of dialog system. New release includes the possibilities to use the partially ordered gradation with values of time intervals (Drawing 3).

In new version of DST the present time moments are saved in a separate file. It gives a better possibility to order the moments.

Implication module is complemented with time symbol choice and possibility to ignore it, if it is consolidated with disjunction or conjunction.

If morphological analyser has detected the verbs with “really” semantics (this construction in Estonian language is a bit different), the DSP dialog system suggest the usage of symbol \vdash in logical formulas.

The special module has been added for processing the words “can” and “may”. At first e2.pl requests the conformation of words and then keeps them in memory. Next script (e3.pl), which creates the logical formulas according to previous script results, adds necessary logical symbols, such as \neg , \diamond , \vdash , parenthesis and time moments.

A new module *seaded.pl* has been added, that gives an opportunity to insert new words in files, which makes available conjunctions, disjunctions, negotiations, universal quantifiers, existential quantifiers, modals, implications, assessments, time symbols in past, present and future (Drawing 4). User may also see the file contents, if it is

Eestikeelsetes tekstides sisalduvate loogiliste konstruktsioonide väljaeraldamise süsteem *Valemite tase.*

Sisesta uus sõna ja vali kuuluvus rühma:

Vaata sisu Sisesta

Analüüsi juurde tagasi Kustuta Va

- Konjunktsioon
- Disjunktsioon
- Eitus
- Üldsus kvantor
- Olemasolu kvantor
- Modaalsus kindlasti(lihtne vorm)
- Modaalsus võib-olla
- Implikatsioon
- Hinnang
- Osalise järjestusega aeg
- Aja sümbolid minevikus
- Aja sümbolid olevikus
- Aja sümbolid tulevikus

Drawing 4

necessary. Program does not add the same words twice in the file.

5. Remark (about next stage).

In order to extract from the natural language those constructions which follow the applications of some derivation rules, the system should operate not only with some single sentences but also with the passage consisting of several sentences. For that purpose an additional choice should be realized in the first script "Analyze totally" and with it repeated individuals, predicates etc. will be indexed on their first appearance on the basis of the given index. To extracting formulas, a new function will be applied, which will offer logical signs to the parts of the text being transformed depending on the fact whether those words appeared earlier or not.

6. Conclusions

Logical constructions, which appeared in texts of 4-6 years old children has been investigated in the present paper. Here it is limited to level of formulas (inference rules have not been studied). Logical constructions, which appeared in texts have been extracted by DST dialog system, which is created by author in 2005.

It has been demonstrated that children use quite complicated logical instruments, including multisorts object symbols, modalities, higher-order

predicates and functionals. Also the application of many different gradations of assessment (as partially-ordered structures) has been demonstrated.

6. References

Curry H. B. 1963. *Foundation of Mathematical Logic*. McGraw-Hill Book Company. New-York, San Francisco, Toronto, London.

Feys R. 1965. *Modal Logics*. Gauthier-Villars. Paris.

Frege G. 1892. *Über Sinn und Bedeutung*. Zeitschrift für Philosophie und philosophische Kritik. Nr 100. p. 25 -50.

Lorents P. 2001. *Formalization of data and knowledge based on the fundamental notation-denotation relation*. Proceedings of the International Conference on Artificial Intelligence. IC – AI' 2001. Volume III. p. 1297 – 1301.

Lorents P. 2004. *Knowledge and understanding*. Proceedings of the International Conference on Artificial Intelligence. IC – AI' 2004. Volume I p. 333 – 337.

Lorents P. 2001a. *Süsteemse käsitluse alused*. (in Estonian) EBS Print. Tallinn.

Matsak E. 2005. *Dialogue system for extracting Logic constructions in natural language texts*. Proceedings of the International Conference on Artificial Intelligence. IC – AI' 2005. Volume II p. 791 – 797.

**SYSTEM DST FOR TRANSFORMING NATURAL
LANGUAGE TEXTS, REPRESENTING ESTIMATES AND
HIGHER ORDER PREDICATES AND FUNCTIONALS**

Erika Matsak

Reprinted with permission, from Proceedings of the 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2006. Volume III p. 79 – 84. Orlando, Florida, USA

System DST for Transforming Natural Language Texts, Representing Estimates and Higher Order Predicates and Functionals.

Erika MATSAK

Department of Mathematics and Natural Sciences
Tallinn University
25 Narva Road, 10120 Tallinn, Estonia
E-mail: erika.matsak@tlu.ee

ABSTRACT

Logical constructions, which is appeared in natural language texts is extracted by DST dialog system, which is created by the author in 2005. When applying DST in detecting logical constructions in speech of children of age 4 – 6, we discovered a need for extension of the system - first of all, to be able to transform the texts that express modalities, estimates (systems of evaluations) and higher order predicates and functionals.

Keywords: The DST dialogue system for transforming natural language texts. Values of property, quality, activity and time intervals, values of time amount.

1. INTRODUCTION

One can say that working in different problem domains applies different logic. For example, in domains related to law, one uses basically the means known from Aristotle's time. In structural synthesis of programs, one needs intuitionistic logic (Tõugu 1988; Tõugu, Mints 1982, 1987). The logics may vary by alphabets (for instance, they may have or have no predicate variables, modalities etc.), by truth-values, interpretations, inference rules, axioms etc. For example, a system of logical instruments developed for systems of interactive processes with time-constraints (Lorents, Mõtus, Tekko 1986) differs from the classical first order predicate calculus, first of all, in the following: it operates with variables for finite sequences of arbitrary length, including sequences of arbitrary length of data, of time moments or of time intervals.

In order to understand which logic or which system of logical instruments is being used in one or another application domain, one must transform natural language texts occurring in the application domain into the language of mathematical logic. An excellent example here can be a part of paper written by G. Gentzen (Gentzen 1936) where a rather conventional (although a mathematical) text was converted step-by-step into formulae of first-order predicate calculus. This work can obviously be never completely automated. At the same time, suitable dialogue systems can be developed and applied. The DST system (Matsak 2005) developed by the author is one of the systems of this kind. It is based on the seven-step conversion procedure described by Lorents (Lorents 2000, Matsak 2005).

When applying DST in detecting logical constructions in speech of children of age 4 – 6, we discovered a need of extension of the system. First of all, to be able to transform the texts that express

- Modalities
- Estimates (systems of evaluations)
- Higher order predicates and functionals.

In the present work we describe the respective extensions and possibilities of their application.

2. SHORT CHARACTERIZATION OF THE DST DIALOGUE SYSTEM

Sentence transformation to the logic formula based on the grammatical analyze, which outlines e.g. the nouns and suggest them as individuals, verbs as predicates etc. This grammatical-logical relation is completely described in the last year publication. [Matsak, 2005]. For grammatical analyze the special software is needed, which may communicate with software for logical formula creation. The DST dialog system uses the FILOSOFT "HTML morph analyzer of Estonian language", located at <http://www.filosoft.ee/>.

It is possible (e.g. for other language purposes) to use the analogous morph analyzers for other language. It needs some modifying of the DSP system, but the main idea and algorithm will still be the same. The table 1 gives some examples for other language software.

The DSP dialogue system has two modes to transform the sentences. The first mode is so to say "teaching" the program. Every sentence, which users type for formula creation, as mentioned before, has a morphological scheme, where the elements signs with special symbols. Software saves these schemes in the special "memory" file. The transformation process may include the next steps [Lorents 2000]: - relocating

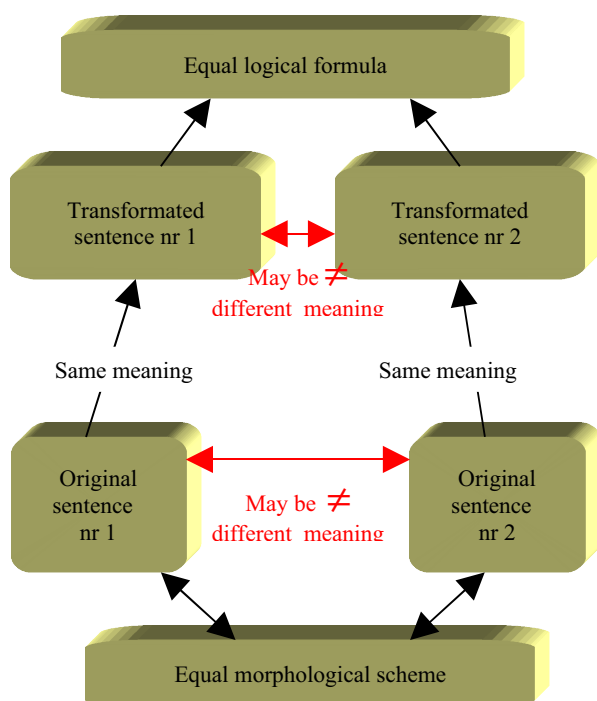
- supplementing (or adding new parts)
- reducing (or removing the parts)
- replacing (or replacing some part by a new text)
- extracting symbols
- categorizing the symbols
- positioning symbols

Every step may bring the morphological scheme difference.

Table 1

Morphological and Orthographic Tools for English	http://www.informatics.susx.ac.uk/research/nlp/carroll/morph.html
French: Morphological Analysis Using the INFL Analyzer	http://humanities.uchicago.edu/orgs/ARTFL/forms_unrest/analyze.query.html
Spanish Morphological Analyzer	http://www.mat.upm.es/~aries/description.html
Morphological analysis of Bulgarian sentence	http://www.uni-plovdiv.bg/dcs/morphe.htm
Morphological Analyses for Inflected Greek Words	http://www.perseus.tufts.edu/cgi-bin/morphindex
Hindi Morphological Tagger	http://ccat.sas.upenn.edu/plc/tamilweb/hindi.html

That's why it is important to save each step's scheme. The words positions may be changed too, and every step moving is saved to necessary "memory" file.



Drawing 1

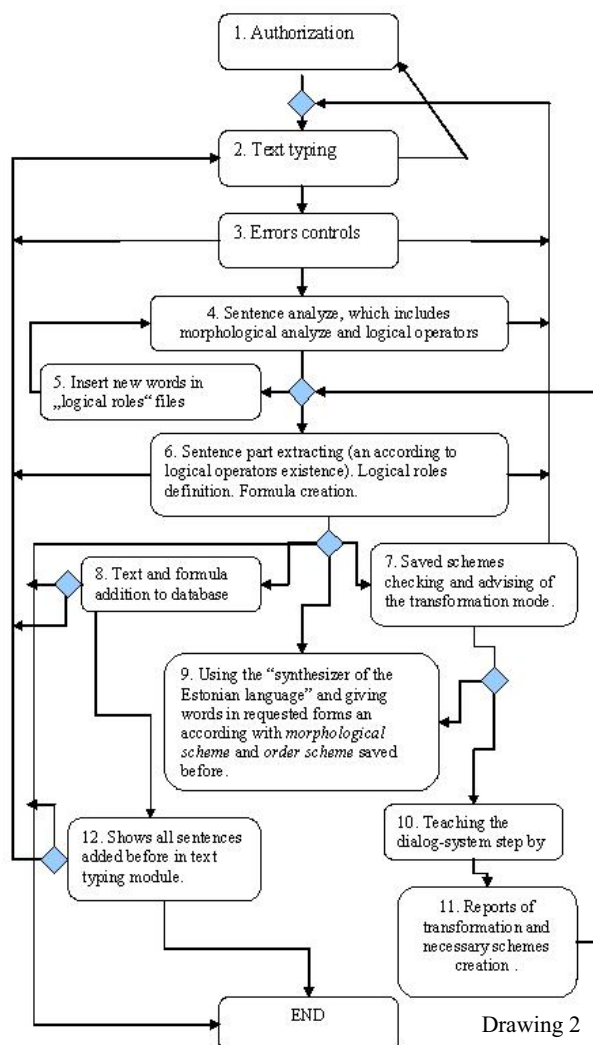
The second mode is the automatic transformation according to the corresponding morphological scheme found prior. This mode is based to the principal, that if we have the equal morphological schemes of sentence, they logical formulas are equal too. Drawing 1 describes the example, how two different (or same) sentences may have an equal morphological scheme, their transformed sentences must save the initial meanings and the result formula for both is the same.

3. GENERAL DESCRIPTION OF THE ALGORITHM

For authorization users need special codes. (Drawing 2)

The sentences must be written grammatically correctly in the typing block and at the end of sentences there must be a period.

During error testing, the software executes also the steps, where the DSP system extracts words and sentences from the text, checks truth-value existence (if the sentence including the verb is not in a form of imperative or question, the software defines the truth-value existence). Always, than user does not agree with software choice, he has an opportunity to change it to opposite.



Drawing 2

All sentences, which truth-value existence is not present, are extruded from the next analysis.

When the user finds the word, which should be in logical operator or symbol role, but software would not detect it, it is possible to go to the software study block (5) and insert the new word in the "logical role" file. This block is protected by password and only experts have access to this. [see also p.4]

Table 2

Denotate	Some words examples	Sign
Negotiation	no, not, false, wrong, incorrect, improper	\neg
Conjunction	and	$\&$
Disjunction	or	\vee
Implication	then	\supset
Equivalence	same, equivalent	\Leftrightarrow
Universal quantifier	all, any, anyone, anybody, every, everything, everybody	\forall
Existential quantifier	exist, existent, existing	\exists
Modals	always, certainly, surely	\square
	maybe, possible, possibly, probable	\diamond
Time symbols	tomorrow, after, afterward, later, now,	t_1, t_2, t_i, t_{i+n}
Values of property, quantity, activity and time amount	Most, a little, a bit, a few, not much, not many, very, ever, so, only, whole, many, too, great	$Val(x)=\gamma$ or $Val(P)=\gamma$ or $Val(t)=\gamma$
Values of time intervals	Anymore, now, first of all, first off, never, ever, sometime, immediately, after, yet, just,	$Val(t)=\tau$
Belongs to set	Mother's, doctor's, etc (word, which meaning is belonging or having	\in
Correctness	really	\vDash

The software detects the next operators and symbols (table 2).

Sentence part extracting is processing in according with logical operator existence, detected in previous block (4). In this article we define the sentence part as such text, which

- is a part of sentence
- represent the atomary formula in according to the logic rules.

Such operators, which divide sentence to the separate parts, are e.g. the conjunction, disjunction, implication. All parts have a personal predicate with related individuals.

Individuals (in Estonian language) automatically separated by the software are: nouns, pronouns, proper nouns, adverbs. Predicates are described in table 3 below.

Table 3

1. adjective	5. two or more verbs as one (example: are going)
2. verb	
3. superlative	6. verb .are. forms+ noun in nominative (example: is teacher)
4. verb .are. forms+ adjective (example: are blue)	7. verb .are. forms+ superlative (example: will best)

In sentences with two or more parts, the same predicates or/and individuals may present few times in different parts. For example in the sentence "*I play with a doll and I love it*" the individual "*I*" is present twice.

In formula creation block individuals are marked as q1, q2, qi, qi+1, and in case of any repetition of some individual they are marked as x1, x2, xi, xi+n. Predicates are marked as P1, P2, Pi, Pi+n and in case of any repetition of some predicate then A1, A2, Ai, Ai+n

Time symbols are available if sentence have two or more parts or have a time moment or time amount assessments. [see also p.4] In the simple way, each part includes verbs or verb combination, the software compares those forms. If they are not equal (the meaning of the tense), then DSP system indicates time symbols.

For all logical operators and symbols, the user can choose the logical role manually, if the DSP suggestion is not right.

Usually, in this block it will appear that the sentence is complicate and needs transformation. For example, people use the form like "*Birds fly and sing*", but for formula creation we need the transformation to "*Birds fly and birds sing.*"

DSP system has a two way for transformation: manually, step by step and automatically mode. [See also p.2] The way for transformation can be advised by special block, which checks the saved schemes and advice of the transformation mode. If the scheme is found, the dialog system shows to user the initial sentence, which has been used for software teaching (10). It is also possible to ignore the suggestion of transformation mode and teach step- by- step.

First transformation mode (10) teaches the dialog system step-by-step. The user types each transformation step as a sentence and the script executes the morphologic analysis. The software saves all steps in necessary files. Studying is protected by password. It is very important that the person, who teaches software, knows enough of sentence transformation.

Reports of transformation and necessary schemes creation give the morphological scheme and order scheme. (Drawing 3).

Eestikeelsetes tekstides sisalduvate loogiliste konstruktsioonide väljaeraldamise süsteem *Valemite tase.*

Lause on transformeeritud korrektsele kujule.

Kokkuvõte programmi õpetamisest:

Mitte keegi praegu ei istu
 mitte kee to ei ist
 D _P_ +sg+#n# to ¬ _V_ +#o#

Ei leidu kedagi, kes praegu istub
 ei leidub keegi kes to istu
 ¬ ∃ _P_ +sg+p# _P_ +sg+#n# to _V_ +#b# Edasi

Initial sentence morphological scheme

After transformation of sentence

Drawing 3

The sentence transformation is:

Nobody seat here now

Not exist somebody, who seats here now

The second transformation mode gives an opportunity to use the DSP system automatic transformation. This block uses “The synthesizer of the Estonian language” and gives words in requested forms according with morphological scheme and order scheme saved before. The dialog system only shows the initial and final sentence, the script executes all other steps (if they exist) automatically. Each word is displayed in the textbox and the user, if necessary, can correct the words or if it is necessary change to other. As a matter of fact, testing has proved that such necessity occurs very seldom. But sometimes the user must confirm words in the transformation result sentence, when synthesizer gives a few outputs for any word.

DSP system gets an opportunity to save the results to the database. For default, there are two databases: personal and research. The database may be copied as a table in the Microsoft Office and other programs (for example MS Excel, SPSS) for the next analysis or for creating a report.

As far as in the typing block (2) user may paste in more than one sentence and at the same time transform only the one of them, all other sentences are kept in special file. Next block (12) gives an



Drawing 4

opportunity to choose sentences typed before and continue the work. It is also possible to delete sentences or finish work. Software interruption as an exit button is added to most of blocks, but formula creation will finish in formula creation block (Drawing 2)

4. NEW DEVELOPED MODULES OF DST DIALOG SYSTEM

The most important changes are provided in block “Sentence analysis, which includes morphological analysis and searching of logical operators.”

Fist of them is related to time symbols extraction. Now it is possible to use time symbols in the simple sentences too (with one sentence part). A new module has been also created for extracting values, which uses saved files with synonyms for values of time moments, time amounts, which includes the words, collected from children's texts. These texts outlined many sentences, in which time moments are partially ordered. It means that there are situation, when we can't decide exactly which process has happened before and which after. (Drawing 4). As far as we sometimes need to analyze not only separate sentences, but part or full text, it is necessary to use in the formulas the assessments of values.

Example:

Ann: *I am watching the TV now, I have already finished the lunch.*

Tom: *Me too, I have just finished the lunch.*

Ann: [P1(x1, x2, t1)& Val(t1)= τ₁] & [P2(x1, x3, t2)& Val(t2)= τ₂]

Tom: [P2(x4, x3, t3)& Val(t3)= τ₃]

Where:
 P1 ∫ am watching
 x1 ∫ Ann

x2 | TV
 Val(t1)= τ₁ | the value of time moment is “now”

P2 | have finished
 x3 | lunch
 Val(t1)= τ₂ | the value of time moment is “already”

x4 | Tom
 Val(t3)= τ₃ | the value of time moment is “just”

(Here is difficult to say which one has been finished first, but we can indicate these moments).

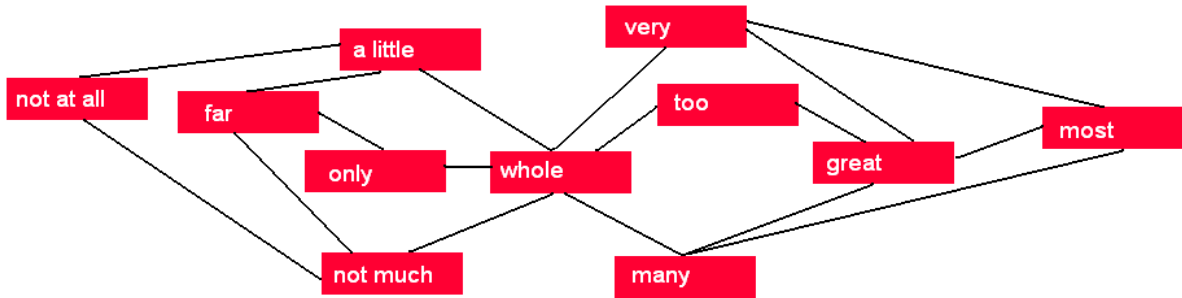
Here we use the symbol of binary relation | for categorization the text as notation and denotation presented by P. Lorents in 2001. We read A|B as the meaning of A is B, or A is a name or a symbol for B.

Other changes are the related to opinion, like the sentences, which include such part as “really do, really have etc”, have special semantics. It means that some argument is controlled and the result is “true”. A second-level predicate may be used with sign ⊢, which means, that truth-value result of statement interpretation is 1 (true).

Example:

“You are really building the house”. (in Estonian language. “Sa teedki maja”.)
 [⊢ (P(x1, x2))],

where
 ⊢ | really
 P | build
 x1 | you
 x2 | house



Drawing 5

Not only time moments may have the partially ordered structure. It is present also for the values of property, quality, and activity. Here the simple examples of the children's text, which the DSP extract with assessment of values:

“I wish to fly a little”. (in Estonian language: “Ma tahan natukene lennata”)
 P(x1)&Val(P)= γ),

where
 P | wish to fly
 x1 | I
 Val(P)=γ | value of predicate a little

The dwarf is a little bigger than candy. (in Estonian language: Päkapiikk on kommist natuke suurem). P(x1, x2) & Val(P)= γ),

where
 P | is bigger
 x1 | dwarf
 x2 | candy
 Val(P)=γ | value of predicate a little

In module for modalities is added the possible way form extraction of such word, like “can” and “may be”. There is the semantic difference between these phrases.

The statement that somebody **may do** something means that a real possibility exists to do something (but it is not compulsory):
 ⊢ [∃P(x1, x2)]

But, the statement that somebody **can do** something means that if somebody asks to do something then this really will happen, if somebody really performs this.
 ⊢ [□∃P(x1, x2, t)]

The other new module makes available to detect belongs (∈) to sets. This situation goes to appear then the text includes the words, which semantic mean are belonging or having. The morphological analyzer easily detects such forms.

A new module (5) has been added, that gives an opportunity to insert new words in files, which makes available conjunctions, disjunctions, negotiations, universal quantifiers, existential quantifiers, modalities, implications, assessments, time symbols in past, present and future.

5. CONCLUSIONS

The new steps of development of the DSP system described in this paper give us the following possibilities:

- to transform the texts to the logical formulas, which includes:
 - o complicate modalities
 - o time symbols and their assessments of values and amount
 - o assessments of individuals and predicates
 - o belongs to set
- to add the new words to the “logical operator and symbol detection module”(4) by module 5.

For the next development stage of the DSP system the full text analysis (right now, all sentences are processing separately) is planned. This approach gives many possibilities for different kind of text analysis, assessment, derivation steps extracting, etc.

5. REFERENCES

- [1]. Gentzen G. 1936. **Die Widerspruchsfreiheit der reinen Zahlentheorie.** *Math. Ann.* 112, Nr 4 (1936), pp 493 – 565.
- [2]. Lorents P., Motus L., Tekko J. 1986. **A language and calculus for distributed computer control systems description and analysis.** 4th IFAC/IFIP Symposium on Software for Computer Control, Graz, Austria May 20 . 23.
- [3]. Lorents, P. , Motus, L. 1986. **Logical tools to describe and analyze the interactive system of processes with prescribed time limiters.** IV conference of the USSR. Application of methods of mathematical logic.. Theses of the presentations. Institute of Cybernetics of Estonian Academy of Science. Tallinn.
- [4]. Lorents P. 2000. **The language and the logic.** EBS-PRINT. Tallinn.
- [5]. Matsak E. 2005. **Dialogue system for extracting Logic constructions in natural language texts.** Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2005. Volume II p. 791 – 797.
- [6]. Mints G., Tyugu E. 1982. **Justification of the structural synthesis of programs.** Science of computer programming 2(3),215-240
- [7]. Mints G., Tyugu E. 1987. **The programming system PRIZ.** Journal of Symbolic Computation, (4).
- [8]. Tyugu E. 1988. **Knowledge-Based Programming.** Addison-Wesley.

THE PROTOTYPE OF SYSTEM FOR DISCOVERING OF INFERENCE RULES

Erika Matsak

Reprinted with permission, from Proceedings of the International Conference on Artificial Intelligence. IC-AI 2007. Volume II p. 489 – 492. Las Vegas, Nevada, USA

THE PROTOTYPE OF SYSTEM FOR DISCOVERING OF INFERENCE RULES.

Erika Matsak

Tallinn University
Department of Computer Science
25 Narva Road, 10120 Tallinn, Estonia

Tallinn University of Technology
Department of Computer Engineering
Raja 15, 12618 Tallinn, Estonia

e-mail: erika.matsak@tlu.ee

Keywords. Natural language texts, Logical constructions, Transformation procedure of texts, Derivation steps and Description of algorithms

Abstract. The derivation steps hidden in natural language texts has been discovered. This discovery provides a prototype system for the extraction of derivation steps. The usage of this system provides an overview of the logical rules used by children between the ages of 10-11. The aim of such research is to understand logical thinking formation.

1. Introduction

The large amount of different logics (for example classic, intuitionistic, modal, fuzzy logics etc.) brings about a question: how do the before mentioned logics occur in intelligence systems that use them? Is logic (A) imputed from outside the system, (B) systems choose logics themselves and input inside, or (C) does a system create its own logic in accordance with necessity?

Logic formation mechanism's main aim is to *create a device, which makes it possible for artificial systems to synthesize necessary logic for its own activity* (Lorents).

One representative of intelligence systems is a human. No doubt people obtain logic in common way (as to say outside – inside): by learning and tracing the usage of constructions. But the another possibility also exists: people create logics during growth and progress in accordance with necessity. For example multivalent logic for quaint mechanics [see for ex. Birkhoff, Neumann 1936], dynamic logics for researching of computer software engineering [see for ex. Hoare 1969, Harel 1978].

One way for extracting human (as intelligence systems) logical mechanisms is researching and analysing natural language texts. In detail: (1) to transform the propositions and reasons from the text to logical formulas, (2) to analyze the order and frequency of the appearing constructions in texts and propositions.

This paper consists of the results of a short report, which the author received from the text of children between the ages of 4-6, by speech recording and from typed essays of an older group of children (aged 10-11). The essays have been kindly given by K. Pata (collected during her pedagogical research). But the main part of the paper describes the algorithm of extraction of logical construction from natural language texts. To explain the basics of algorithm P. Lorents' method for system mining has been used.

2. Lorents' method in the case of mining of inference rules

The main steps of this method in the mentioned case are as follows:

- To fix the system of research (in this case the text or speech of people).
- To fix the age of a person who delivers the texts or speech
- To extract logical constructions from the texts: (1) constructions which are presented by propositional formulas extracted by the text transformation procedure [see Lorents 2000, Matsak 2005, Matsak 2006] and (2) derivation steps, which give expression for the reasons, extracted by procedures related to the *matapredicate of Relatedness* and *matapredicate of Similarity*.
- To create a 3-dimensional array, where one parameter is time (age of a person), the second consists of all logical constructions at the moment in time and the third is the frequency of construction appearance.

If a larger amount of data is collected, it will give an understanding how people use logical construction and show us the growth of construction usage by age. In another words, show us the dynamic picture of logical construction "toolbox" development and usage.

Such an overview further provides the base for the research, whose aim is to understand how intelligent and self developing systems obtain and build up the logical construction "toolbox" (including different logics).

3. Short overview of logical constructions, extruded from the children recorded speeches in age 4-6.

The author has used the dialog system DST [Matsak 2005, 2006] in her own researches. By this dialog system the logical construction of children aged 4-6 was extracted.

As a result it occurred that children use all components of first-level predicate calculus, including:

- sorts of individuals
- time moment and time related symbols
- unary and n-ary predicates
- all logical operators ($\neg, \&, \vee, \supset$)
- quantifiers (\forall, \exists)
- modalities (\diamond, \square)

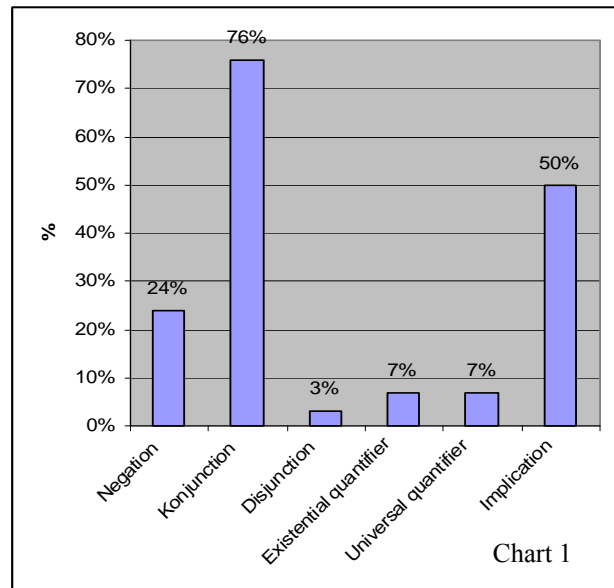
It has been demonstrated that children use for proposition creation the quite complicated logical instruments, including multisort object symbols, modalities, higher-order predicates and functionals. Also the application of many different gradations of assessment (as partially-ordered structures) has been demonstrated.

4. Examples of logical constructions, extruding from children texts aged 10-11

This year researches have been based on children's essays with answers to the questions and their reasons. The amount of essays was 176. Data was collected for K. Pata pedagogical research as a part of the study on the development of primary students' cohesiveness and consistency of conceptual frameworks about seasonal changes. This was investigated as students were studying in a computer-supported inquiry environment called "Young Scientist." [Pata, submitted]. The given texts have been kindly given by her for logic researches. These texts have enabled us to not only consider (or examine) logic constructions, which are necessary for proposition creation, but also for reason conception.

Children aged 10 – 11 years also use the same set of logic constructions, as those at the earlier ages of 4-6. The use of sets and operations above them are added to this. Also we can find the use of terms and various ways of comparing predicates beginning with "more-less" and finishing with functional dependences.

Further is given diagram (chart 1), which shows the results by quantity of usage the standard logic operators and quantifiers.



In the given texts children between the ages of 10-11 answered a question "Why does the winter (summer) come?"

5. The prototype of dialog system for extraction of derivation steps

The author of the paper has been discovering the possibilities for human logic extraction from natural language text for 3 years. At first software was created, which gave the logical formulas of separate sentences [Matsak 2005, 2006]. During this year the prototype of a dialog system was created, which will be extracting the derivation steps. Realized algorithm can be described mathematically. One example of such an explanation has been given by P. Lorents [P. Lorents 1993, 2002]. Shortly we can say that we use two kinds of Metapredicates: Metapredicate of Similarity and Metapredicte of Relatedness.

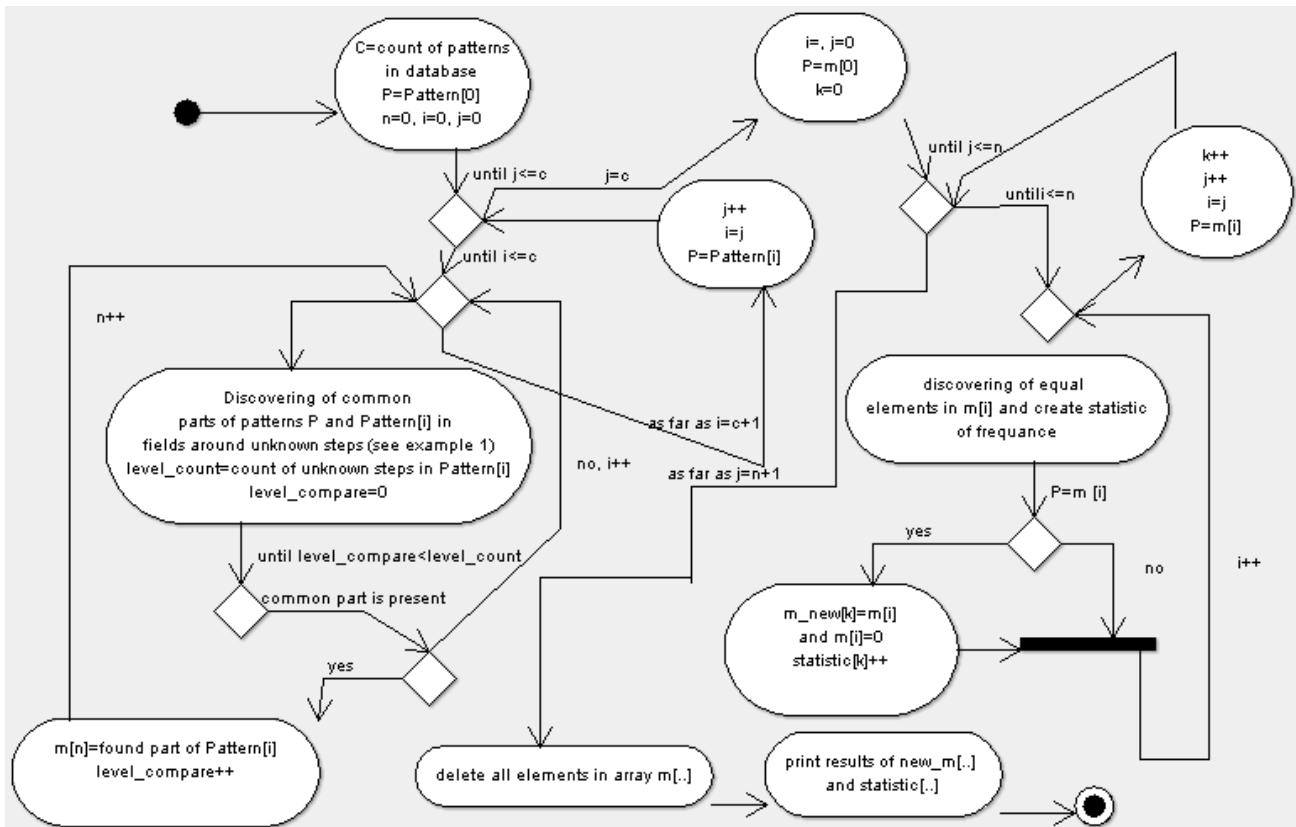
The denotation of Metapredicate of Relatedness is, in some meaning, our intuition about what kind of elements may have the important properties for us and between which elements some important property may be present. However, we don't need to know, what kind of properties or relation are searched. For at first, we just accept, that something exist and this is the explanation of Metapredicate of Relatedness.

The Metapredicate of Similarity is, in some meaning, our intuition about some similarity existence between two elements or sets of elements.

In common words, this procedure works as follows:

- at first we need to create the Cartesius degrees of set $H : H^{(n)}$, where $n=1,2,3, \dots$

- next, we compare parts (kits) of set $H^{(n)}$ (pairs, ordered ternaries, etc) and decide (A) are the elements from the



Picture 1. Algorithm for extraction the derivation steps from the natural language texts.

kit related between each other or not, (B) is this kit similar to some other kit from $H^{(n)}$.

- if we collect the similar kits in separate classes, then we finally get all the relations in a set H (unary, binary, ternary etc. relations).

If set H consists of propositions, then using the before mentioned procedure gives us the derivation steps for reason explanation [Lorents 1993, 2002].

6. Conclusions

For each person there exists a knowledge base. In reasoning each person uses this knowledge, leaning before as axioms. Thus it is possible to tell, that for everyone there is the world W with a set of axioms and rules. Wrong answers as a rule take place because of wrong initial knowledge (axioms). As far as there are different types of thinking, so-called mathematical, abstract and etc., it is quite probable that different groups of people use different steps to formulate conclusions. Unfortunately at a given stage of research we cannot tell which existing set of derivation steps and for what group of people they are most usable.

From the available texts some repeating steps of conclusions have been allocated. The most popular (~85 %) was still mentioned by Aristotle Modus Ponens, which at Gentzen [Gentzen, 1936] has the name *cutting*.

$$\frac{\rightarrow A \quad A \rightarrow B}{\rightarrow B}$$

However, the interesting variations of *modus ponens* appeared.

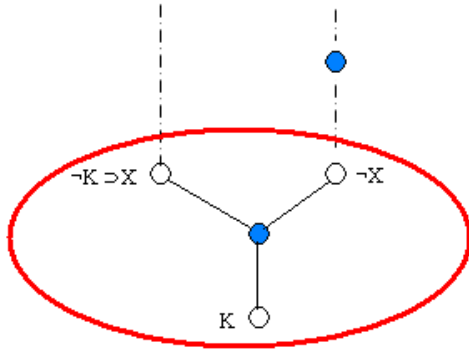
Example (see also the picture 1)

Children have used the next derivation step:

$$\frac{\neg K \rightarrow X \quad \rightarrow \neg X}{\rightarrow K}$$

Where K denotes the preposion: „Spring comes“, and X: „Summer comes after winter“

This construction can be visualized by a graph. For derivation step extraction we need to discover the field around the “?” (Unknown) step.



Picture 2. Unknown derivation step

If we find in the next text the same part of graph (K and X may be present by other letters, but logical operator and quantifiers must be the same) we can call them as “Similar” derivation steps.

The problem lies in the hidden conclusions or axioms. The people who collect the answers must pay special attention to the very detailed explanation. In this example the sentence was used:

“If spring does not come, then summer comes after winter.” And the child knows (but this is hidden), “that summer comes after winter“ is absurd.

The next frequently used derivation step a well know *kontopository rule*

$$\frac{\neg K \rightarrow \neg S}{S \rightarrow K}$$

For example: “If spring does not come, the summer will not come too. And as far as summer does come, then spring will come too.”

Acknowledgement: The author of the given work expresses profound gratitude to the professor Peeter Lorents for assistance in a writing of given clause.

7. References

- [1] Birkhoff G., von Neumann J. 1936. *The logic of quantum mechanics*. Ann. Math. 37, 823–842.
- [2] Curry H. B. 1963. *Foundation of Mathematical Logic*. McGraw-Hill Book Company. New-York, San Francisco, Toronto, London.
- [3] Feys R. 1965. *Modal Logics*. Gauthier-Villars. Paris.
- [4] Frege G. 1892. *Über Sinn und Bedeutung*. Zeitschrift für Philosophie und philosophische Kritik. Nr 100. p. 25 -50.

[5] Gentzen G. 1936 *Die Widerspruchsfreiheit der reinen Zahlentheorie*, Mathematische Annalen, 112: 493-565.(1936)

[6] Harel D. 1978. *Logic of programs; Axiomatic and descriptive power*. Technical report. Massachusetts Institute of Technology. May, 1978.

[7] Hoare C. A. R. 1969. *An axiomatic basis for Computer programming*. CACM 1969, 13, p.576 – 580.

[8] Lorents P. 2000. *Language and logic*. EBS Print. Tallinn

[9] Lorents P. 2001. *Formalization of data and knowledge based on the fundamental notation-denotation relation*. Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2001. Volume III. p. 1297 – 1301.

[10] P. Lorents. 2002. *A System Mining Framework*. The 6th World Multiconference on Systemics, Cybernetics and Informatics. Proceedings. Vol. 1. 195 – 200.

[11] Matsak E. 2005. *Dialogue system for extracting Logic constructions in natural language texts*. Proceedings of the International Conference on Artificial Intelligence. IC – AI’2005. Volume II p. 791 – 797.

[12] Matsak E. 2006a. *Using Natural Language Dialog System DST for Discovery of Logical Constructions of Children's Speech*. The 2006 International Conference on Artificial Intelligence IC-AI 2006. Las Vegas, Nevada, USA (June 26-29, 2006)

[13] Matsak E. 2006b *System DST for Transforming Natural Language Texts, Representing Estimates and Higher Order Predicates and Functionals*. The 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2006. July 20-23, 2006 . Orlando, Florida, USA.

[14] Pata K. (submitted) „Students’ Conceptual Development Related to Seasonal Changes by Virtual Inquiry Using “Young Scientist” as a Learning Environment” Submitted to the “Journal of Learning Science”

English version is edited by Leon M. Miller

**IMPROVED VERSION OF THE NATURAL LANGUAGE
DIALOG SYSTEM DST AND ITS APPLICATION FOR
DISCOVERY OF LOGICAL CONSTRUCTIONS IN
CHILDREN'S SPEECH**

Erika Matsak

Reprinted with permission, from Proceedings of the International Conference on Artificial Intelligence. IC-AI 2008. Volume II p. 332 – 338. Las Vegas, Nevada, USA

IMPROVED VERSION OF THE NATURAL LANGUAGE DIALOG SYSTEM DST AND ITS APPLICATION FOR DISCOVERY OF LOGICAL CONSTRUCTIONS IN CHILDREN'S SPEECH

Erika Matsak

**Faculty of Information Technology
Tallinn University of Technology**
15 Raja, 12617 Tallinn, Estonia
E-mail: erika.matsak@tlu.ee

Abstract

The improved version of the dialogue system DST for extracting logical constructions from natural language texts is considered. Results received by means of DST and by the analysis of children's texts are described. Many interesting logical constructions, including some steps of a conclusion are revealed.

Introduction

The improved version of the DST dialogue system for extracting logical constructions from natural language texts [Matsak 2005, 2006, 2007] is considered in the present work. The purpose of the creation and usage of the DST system is the study of mechanisms of formation of logical constructions in self-developmental intelligent systems. Children are examples of such systems. Logical constructions (including formation of formulas and a deduction of formulas) are in turn connected with mechanisms of extraction and use of knowledge [Jackson 1985, Padhy 2006]. It is natural to believe that development of the above-stated mechanisms is reflected in texts, which children create and apply. It brings the necessity of transforming text from a natural language to a logical language (for example the language of calculation of predicates). Next step is to reveal the use of logic steps and rules of a conclusion (as sets of similar steps [Lorents 2002]). For this purpose it is necessary to modify and improve the DST system with additional modules.

By means of the advanced system it was possible to show that even very small children (for example 3 years old) can use an unexpectedly rich set of logic means.

Procedures for extraction of logic steps of a conclusion

Here we bring only the basic procedures that are necessary for extracting logic steps of a conclusion [Matsak 2007].

1. Find formulas in which repetition of individuals and predicates exist (at a designation symbols x or A)
2. If formulas satisfy this requirement, then print them out with transformed sentences
3. According to the amount of the chosen formulas the operation cycle is started
 - 3.1. Split each formula to segments, new variables store separate designations of individuals and predicates
 - 3.2. For definition of pairs with repeating individuals and predicates the new operation cycle is started
 - 3.2.1. Search pairs of formulas which contain at least one same individual and also together with individual the same predicate. *The found pairs formulas can be logic steps.*

Logical constructions and their increase at age of 3-5 years

During last year 216 different sentences used by children between the ages of 2,7 and 3,1 have been discovered. A surprising revelation was that children at this age use a very rich set of logic constructions, including an extensive alphabet of

logic calculations. A young child starts using speech by using separate names of individual objects to create ordered pairs: (the name, a real subject), which represents knowledge [Lorents, 2001]. However, children use separate predicates too, which speaks about formation of simple subsets of objects in a child's brain.

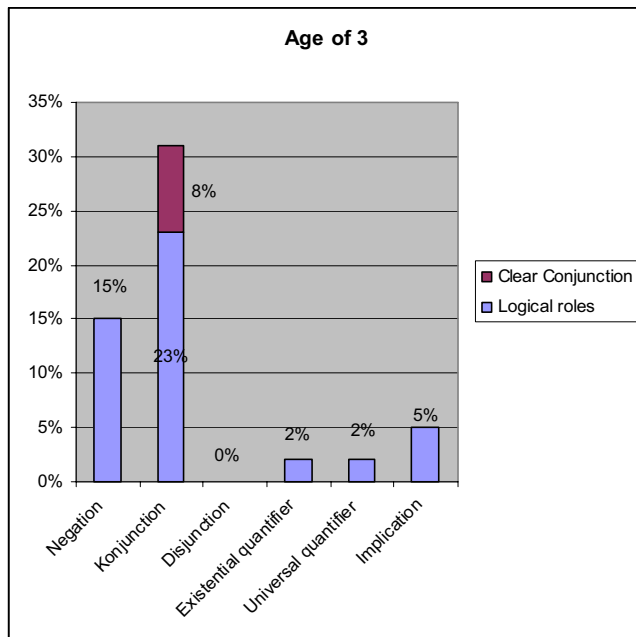


Figure 1

Remarks: A child may not initially represent understanding in a full expression like "Yellow ducklings". For example, the child shows the ducklings on the picture and says: "ducklings". When somebody asks about the color of the ducklings the child answers: "Yellow".

Texts of three year old children will be studied in the near future in more detail in order to reveal the order of occurrence of logic operators and quantifiers. But it is already possible to see that the first logic operator that appears at the child text is negation. After a very small time interval (2-3 months) the child already uses almost all logic operators in the obvious or implicit form. The diagram of figure 1 shows the usage of classical operators and quantifiers. However, it is necessary to note a situation with conjunction. Only 8 % of the conjunctions were used in a natural way by the children. Another part is the

hidden conjunction as a result of text transformation.

Procedure of transformation has been described for the first time by Lorents in 2000 [Lorents, 2000] and used for development of dialogue system DST [Matsak 2005, 2006, 2007]. Two years ago elements of assessments of predicates, individuals and time and different modalities were found in children's speech.

Examples:

– In the sentence "It is too big birdie" we have to deal with assessment „too“ for the predicate „big“.
 $P1(q1 q2) \& [Val(P1) = \epsilon 1]$

– In the sentence „A lot of children are here“ the amount of people is assessment.
 $P1(q1 q2) \& [Val(q2) = \gamma 1]$

– In the sentence „You will be soon healthy“ the word „soon“ is assessment, which declare the time moment.
 $P1(q1 t0) \& [Val(t0) = \tau 1]$

The statement that somebody may do something means that maybe is the possibility to do something and it is real: $\models [\Diamond P(x1, x2)]$

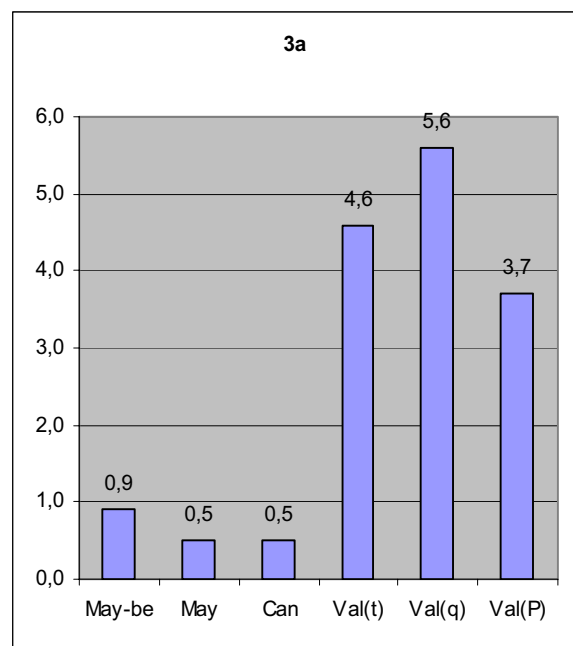


Figure 2

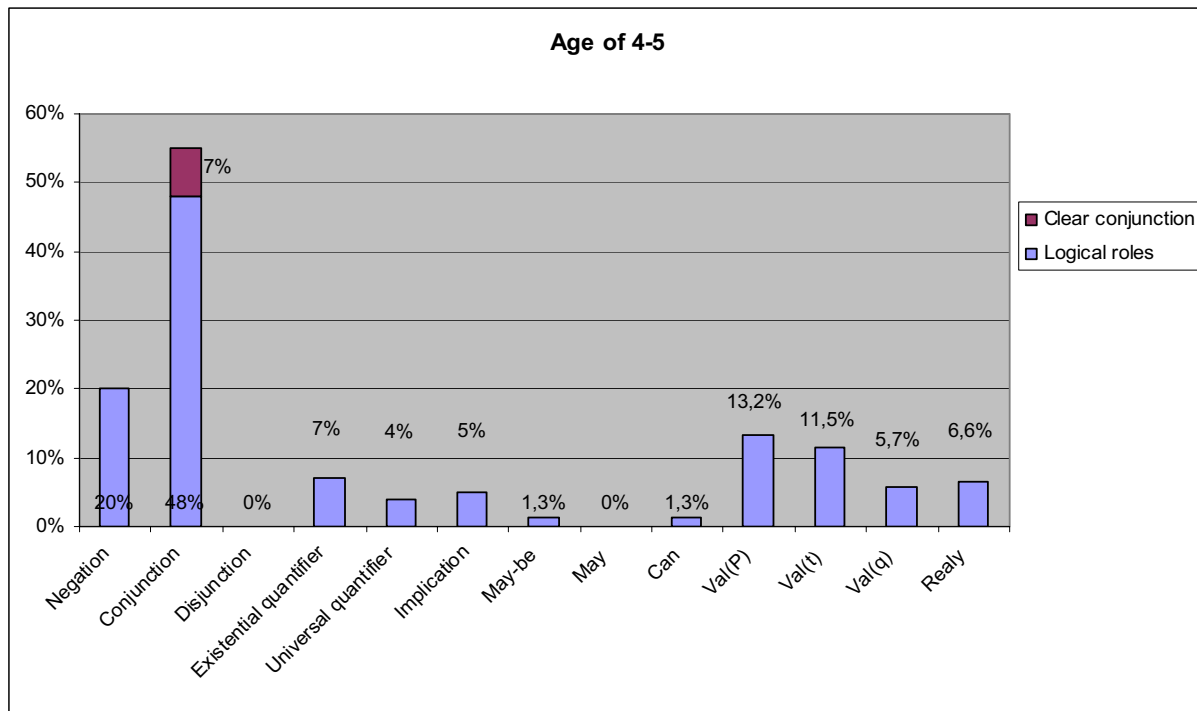


Figure 3

The statement that somebody can do something means that if somebody asks to do something then this really will happen, if somebody really performs this. $\models [\Box \Diamond P(x1, x2, t)]$

The sentences, which include such parts as “really do, really have etc”, have special semantics [Matsak 2006]. It means that some argument is controlled and the result is “true”. A second-level predicate may be used with sign \models , which means that statement interpretation is true. $\models (P(x)) \int \varphi(P(x))=1$, where the φ is the formula interpretation.

The following diagram (figure 2) shows the use of the logic elements described above. On the next diagram (figure 3) the same logic constructions are present, but already for age of 4-5 years (amount of sentences is 227). It is interesting to notice, that despite of the increased conjunction, frequency of use of conjunction *in a clear way* was the same, also the quantity of implication, use of modalities and assessment of individuals essentially has not changed.

However, the amount of assessments of predicates and time is rising, and also the new logic construction has appeared at this age.

Hidden use of equivalence

During the transformation of the text the sentences with the word "otherwise" have drawn some attention. We shall show, that sentences like "it is forbidden to come here, otherwise the uncle will be displeased" include hidden equivalence.

Let's transform the text for extraction of hidden equivalence:

If you come here, then uncle will be displeased and if you don't come here, then event X will take a place (Most likely in many cases under X understand a phrase "uncle will not be displeased"). In other words we can write down after text transformation the sentence in a following way: $(A \supset B) \& (\neg A \supset \neg B)$. It is easy to find, that the formula represents equivalence.

Extracted inference steps

In the report [Matsak, 2007] inference steps were discussed that had been extracted from the texts of older children (10-11 y.o).

- Aristotle *Modus Ponens* (which is variations of Gentzen's [Gentzen, 1936] *Cutting*).

$$\frac{\rightarrow A \quad A \rightarrow B}{\rightarrow B}$$

- some variation of *Modus Ponens*, as

$$\frac{\rightarrow \neg A \quad \neg B \rightarrow A}{\rightarrow B}$$

- *Contrapository rule*

$$\frac{\neg B \rightarrow \neg A}{A \rightarrow B}$$

There has been no research about the logic steps that children of the age 3-6 use. However, available texts allow to us to see without special difficulties a lot of transitions, which are correct steps of a logic conclusion, or are transitions similar to them.

The simplest step which has been used for confirming the ideas was a trivial tautology:

$$\frac{\rightarrow A}{\rightarrow A}$$

Children of young age (for example 2 years of 11 months) willingly used it for confirmation of the correctness.

Example:

1. I like this book. Why? Because I like it.
2. I eat. Why? Because I eat.

In the three year olds some incorrect, but nevertheless interesting steps were noticed:

$$\frac{(\exists t)A(x, t)}{A(x, t)}$$

Example:

The alarm clock rings. Why? Because alarm clock rings sometimes.

Also:

$$\frac{(\exists t) A(x_1, t) \ \& \ (\exists t) P(x_2, t)}{A(x_1, t)}$$

And the correct logical transition with conjunction removing (\rightarrow & \neg)

$$\frac{A \ \& \ B}{A}$$

Remark:

The text examples for the last two transitions and the explanation in detail how the DST dialogue system extracts the steps will be described in the next chapter.

Children of the age of 5-6 years have been noted to use steps of conclusion similar to syllogisms, in both incorrect and correct ways.

Example 1 (incorrect step):

All wild animals hibernate in the winter. Frog is not a wild animal. Frog doesn't hibernate in the winter.

$$\frac{S(x) \supset (\forall x)A(x)}{\neg S(q)} \\ \neg A(q)$$

Example 2 (correct step):

Student: If you could be an animal, who you would be?

Child: Bird. Student: Is the bird an animal?

Child: No. Because a bird can fly, but animals can't fly.

$$\frac{A \rightarrow \neg B}{B \rightarrow \neg A}$$

Extraction of inference steps using the DST dialogue system

Let's examine the next part of text:

Ducks swim here. But ducks can fly too. Bad dog wants to eat duck. But he is bad. Why? Because he is bad sometimes and he is good sometimes.



Figure 4

With input of text it is necessary to turn on an option, which gives the possibility to transform the text into a form where repeating individuals and predicates are defined. Next the DST rejects the sentences which have no true-values. In this case a sentence without a true-value is the question "Why".

Further it is shown that the program has found repeating individuals "ducks" and "he" and has designated them accordingly x1 x2 (figure 4). It is also detected that the repeating predicate A1 "is bad".¹

The second sentence "But ducks can fly too" has a transformation for extracting the hidden conjunction. As a result we get: "Somebody can fly and ducks can fly too" and the formula $(\exists q2)A2(q2) \& A2(x1)$.

The third sentence is transformed into: "Dog is bad and dog wants to eat duck" and the new repeating individual - x3 "dog" is found.

Further it is necessary to replace the pronoun with concrete name and as a result our sentence will be "But dog is bad" $A1(x3 t0)$. And at last, the sentence "Because dog is bad sometimes and dog is good sometimes" will give the formula: $(\exists t0)A1(x3 t0) \& (\exists t1)P3(x3 t1)$.

In process of search of steps of the conclusion DST has the following operations. First of all a dialogue system separates out all sentences where individuals designated by "x" or predicates designated by "A" are used. Further, the system chooses from a set of sentences, extracted before, the pairs that simultaneously use both a repeating predicate and a repeating individual (figure 5).

In our case there are the following pairs:

Dog is bad and dog wants to eat duck

But dog is bad

$A1(x3 t0) \& P1(x3 x1)$

--

$A1(x3 t0)$

Dog is bad and dog wants to eat duck

Because he is bad sometimes and he is good sometimes

$A1(x3 t0) \& P1(x3 x1)$

--

$(\exists t0) A1(x3 t0) \& (\exists t1) P2(x3 t1)$

¹ Designations "x" and "A" are used for sign of repeating individuals and predicates

Eestikeelsetes tekstides sisalduvate loogiliste konstruktsioonide väljaeraldamise süsteem *Valemite tase.*

Tagasi Välju programmist Tuletussammud Annuleeri tekst-> Esilehele Kustuta andmebaas

Andmebaas: sammud7

Lapse s.p.	Uuringute k.p.	Esialgne lause	Transformeeritud lause	Valem
?..?	?..?	Pardid ujuvad siin.	Pardid ujuvad siin	$P1(x1 q1)$
?..?	?..?	Aga pardid oskavad lennata ka.	leidub keegi kes oskab lennata ja pardid oskavad lennata	$(\exists q2) A2(q2) \& A2(x1)$
?..?	?..?	Paha kutsu tahab pardi ära süüa.	kutsu on paha ja kutsu tahab pardi süüa	$A1(x3) \& P2(x3 x1)$
?..?	?..?	Aga tema on paha.	Aga tema on paha	$A1(x3)$
?..?	?..?	Sellepärast, et vahepeal on ta paha ja vahepeal on ta hea.	Sellepärast et vahepeal on ta paha ja vahepeal on ta hea	$(\exists t0) A1(x3 t0) \& (\exists t1) P3(x3 t1)$

Valitud laused:

Lisa tuletussammud andmebaasi -->

kutsu on paha ja kutsu tahab pardi süüa
 Aga tema on paha
 $A1(x3) \& P2(x3 x1)$
 $A1(x3)$
 Opti meeritud kujul:
 $A1(x3) \& P1(x3 x1)$
 --
 $A1(x3)$

kutsu on paha ja kutsu tahab pardi süüa
 Sellepärast et vahepeal on ta paha ja vahepeal on ta hea
 $A1(x3) \& P2(x3 x1)$
 $(\exists t0) A1(x3 t0) \& (\exists t1) P3(x3 t1)$
 Opti meeritud kujul:
 $A1(x3) \& P1(x3 x1)$
 --
 $(\exists t0) A1(x3 t0) \& (\exists t1) P2(x3 t1)$

Aga tema on paha
 Sellepärast et vahepeal on ta paha ja vahepeal on ta hea
 $A1(x3)$
 $(\exists t0) A1(x3 t0) \& (\exists t1) P3(x3 t1)$
 Opti meeritud kujul:
 $A1(x3)$
 --
 $(\exists t0) A1(x3 t0) \& (\exists t1) P1(x3 t1)$

Figure 5

Remark. In this and also other cases the premise and conclusion in natural language text are put contrariwise, in comparing with predicate calculation schema. Such rearrangement of premise and conclusion is normal and it removes the wish to affect the conclusion.

But dog is bad

Because he is bad sometimes and he is good sometimes

$A1(x3 t0)$

--

$(\exists t0) A1(x3 t0) \& (\exists t1) P1(x3 t1)$

Next operation is renumbering individuals and predicates. In each pair the individual or predicate with least index should be numbered by index 1 and the others in the increasing order.

Conclusions

The usage of the DST dialogue system has brought out the fact that in one kind of intellect system, like a child, the appearance and usage of logical constructions may have jumps: during a rather short time interval children learn to use negation, and then practically all other operators, modalities and quantifiers. In case of inference

steps the picture is not yet clear and additional research is needed.

At the same time by means of the developed dialogue system it was possible to show the following:

1. Ability to prove the statement (in a context of logic - a deduce of formulas) can develop in early years of a person's development
2. Ability to prove the statement can be developed almost in parallel with development of ability to set up the statements (in a context of logic - formulas). The following is still not clear :
 - a. how the ability to apply correct steps of a conclusion (in understanding of logic) is developed
 - b. how the rules of a conclusion are formed from similar steps of a conclusion

For the decision of these problems the further development of the DST system is required.

Acknowledgements: The author of the given work expresses profound gratitude to professor Peeter Lorents for assistance in a writing of given clause, to Kaire Kollom and students of Tallinn University for recording of children speeches and to Rain Ottis for English version edition.

Matsak E. 2007. The prototype of system for discovering of inference rules. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2007. Volume II p. 489 – 492.

References

Jackson Ph. C. 1985. Introduction to artificial intelligence. Second, enlarged edition. Dover Publications, INC., New York.

Padhi N. P. 2006. Artificial Intelligence and Intelligent Systems. Oxford University Press.

Gentzen G. 1936 Die Widerspruchsfreiheit der reinen Zahlentheorie, *Mathematische Annalen*, 112: 493–565.(1936)

Lorents P. 2000. Language and logic. EBS Print. Tallinn

Lorents P. 2001. Formalization of data and knowledge based on the fundamental notationdenotation relation. Proceedings of the International Conference on Artificial Intelligence. IC – AI' 2001. Volume III. p. 1297 – 1301.

P. Lorents. 2002. A System Mining Framework. The 6th World Multiconference on Systemics, Cybernetics and Informatics. Proceedings. Vol. 1. 195 – 200.

Matsak E. 2005. Dialogue system for extracting Logic constructions in natural language texts. Proceedings of the International Conference on Artificial Intelligence. IC – AI'2005. Volume II p. 791 – 797.

Matsak E. 2006a. Using Natural Language Dialog System DST for Discovery of Logical Constructions of Children's Speech. The 2006 International Conference on Artificial Intelligence IC-AI 2006. Las Vegas, Nevada, USA (June 26-29, 2006)

Matsak E. 2006b System DST for Transforming Natural Language Texts, Representing Estimates and Higher Order Predicates and Functionals. The 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2006. July 20-23, 2006 . Orlando, Florida, USA.

ON THE LOGIC MODULE IN INTELLIGENT SYSTEMS.

Erika Matsak

Reprinted with permission, from Proceedings of the International Conference on Artificial Intelligence. IC-AI 2009 Las Vegas, Nevada, USA

On the logic module in intelligent systems

Erika Matsak

Faculty of Information Technology, Tallinn University of Technology, Tallinn, Estonia

Abstract - While developing and implementing the natural language text transformation dialogue system prototype DST, an unexpectedly diverse and large logical arsenal has been found in children's texts. This is true for the levels of logical operators and quantifiers as well as for the applications of inference rules. The identified logical tools and the analysis of their implementation points to one way to construct the logic module in intelligent systems. We have investigated technical, mathematical and in some sense even philosophical aspects of development of the logical module.

Keywords: DST dialogue system; logic operations, quantifiers, modalities and inference steps in children's texts; implementing logic formulas and inference steps with digital circuits; logic module.

1 Introduction

The ability to generate and implement correct decisions in intelligent systems (IS) becomes increasingly important with every passing day. One formal representation of making correct decisions is the use of the correct formulas of mathematical logic. As a rule, there are two sources for these formulas:

- a priori correct formulas (axioms, basic postulates etc.);
- results of implementing correct inference steps (where inferring from correct formulas results in new correct formulas).

Depending on options and choices, there are other approaches for getting correct decisions (Tamisier 2008.). For example, various graph based methods (see Bo Haoy, Tomohiro Yoshikaway, Takeshi Furuhashiyz, Shin-ichi Sugiuraz. 2008.). In several studies on deductive synthesis of programs (especially when this area of research was still taking shape), two types of graphs were used to represent steps that engineers use to solve problems (see Tyugu 1988, 2007). An interesting example is the use of graphs to model the thought process of the famous military commander and politician Otto von Bismarck (see Lukov, Sergejev 1983).

Regardless of the theoretical representation and choice of implementation tools of correct decisions and their generation, two important questions must be dealt with:

- how are correct decisions and the tools to make them implemented in natural intelligent systems (NIS), including

humans;

- how are the formalization and the tools to generate correct decisions acquired or developed in intelligent systems.

Assuming that the mechanisms in natural intelligent systems are good enough to guarantee the subsistence of the organisms (including humans) that use them in various environments, we could take these mechanisms as examples in our search for sufficiently robust and effective solutions for artificial intelligent systems (AIS). Based on this premise the author has spent several years studying the logic constructs in children's (mainly between the ages of two and five years) texts, in order to explain their existence and the development of their use (especially so-called first sightings) compared to the children's development in time. These studies (see Matsak 2005 – 2008) have identified surprising results in logic operations, quantifiers, modalities, as well as very „early” sightings of inference steps. This has raised questions about which of these logic constructs are so-called software based, which are hardware based and how are they all implemented (see Matsak 2009). In the following chapters we discuss aspects related to these problems

2 Logic operations, quantifiers and modalities in children's texts

One of the most foolproof ways to identify the presence of some sort of logic constructs is to look at their „traces” in the texts of a natural language. The Lorents' text transformation procedure can be used to step-by-step transform the text segments that represent a logic construct into logic formulas (see Lorents 2000). The DST dialogue system prototype, which was developed to implement this transformation procedure, has enabled the author to study recordings of children of various age groups, collected over a period of more than five years, as well as the texts in the Childe database (see Child Language Data Exchange System). The results of this study are somewhat surprising in terms of the presence of various logical operators (including negation, conjunction, disjunction, implication and equivalence), quantifiers and modalities (see Matsak 2005, 2006, 2007). It turned out (see Figure 1) that the first operation detected in children is the negation (including in a – NB! 1.2-year-old child). Two-year-olds demonstrate most of the „common” logic operations, including quantifiers and various modalities (see Figures 1 and 2). This abrupt emergence of many simultaneous logic operations points to a

possibility that certain permanent configurations are formed in the „elemental basis” of the human brain during this time

(see Matsak 2009) that then become usable for describing and analyzing situations.

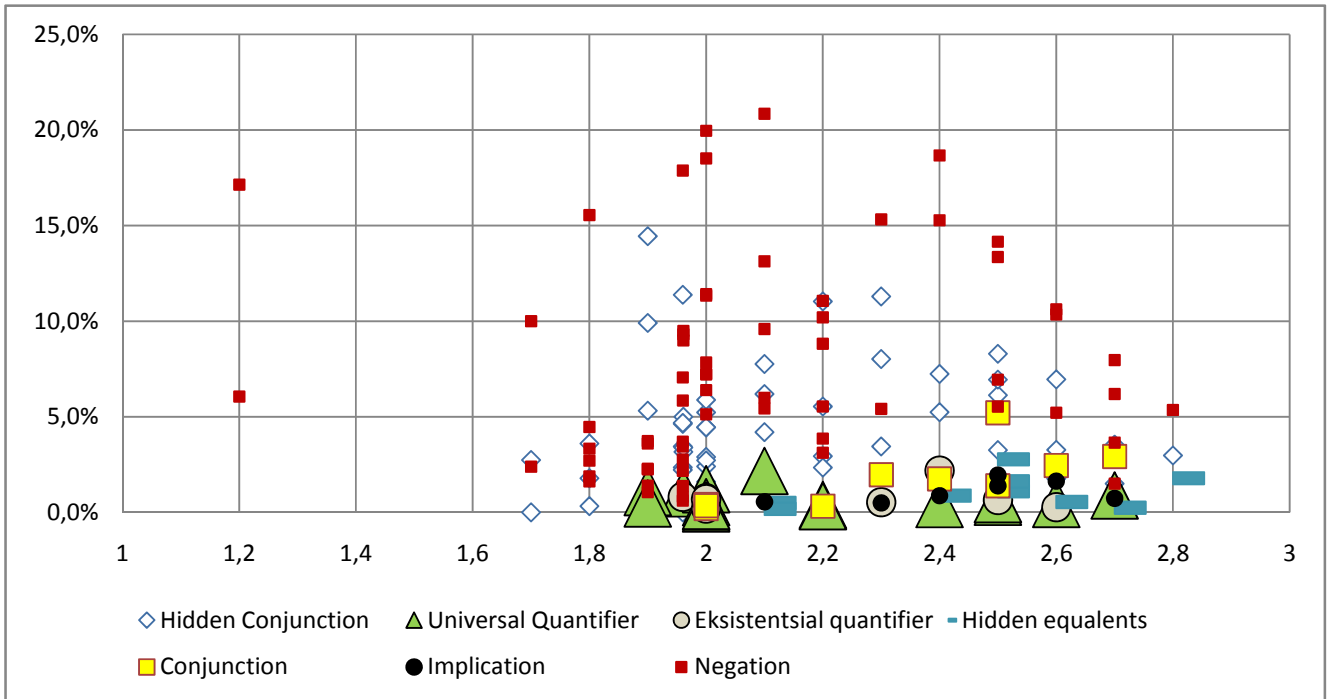


Figure 1 Operations and quantifiers identified in children's texts

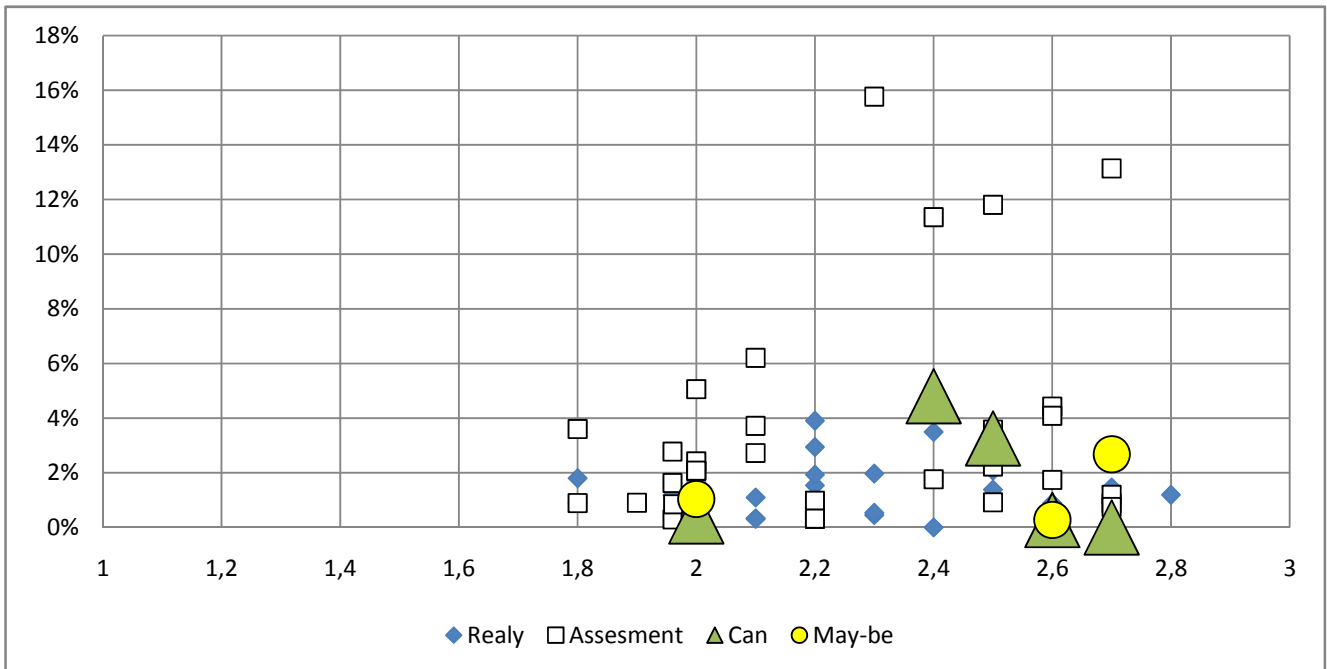


Figure 2 Modalities identified in children's texts

3 Logical inference steps in children's texts

The development and implementation of the DST dialogue system prototype identified a diverse arsenal of inference tools that are used even during very young age to back up one's arguments. Just like many adults, children try to reason by using both logically correct, as well as seemingly correct (but in reality incorrect) inference steps. Table 1 shows the inference steps identified in the studied texts.

Table 1 Inference steps identified in children's texts

$\frac{A}{A}$	(2.9)	$S(x) \supset (\forall x)A(x)$	(5-6)
		$\neg S(q)$	
		$\neg A(q)$	
$\frac{A \supset \neg B}{\neg A \supset B}$	(2.6)	$\frac{\neg(\exists t A(t,x))}{(\forall t)\neg A(x)}$	(5.9)
$\frac{A \supset B}{\neg A \supset \neg B}$	(2.6)	$\frac{A \supset \neg B}{B \supset \neg A}$	(5.11)
$\frac{(\exists \alpha A(\alpha) \& \exists \beta A(\beta))}{\forall x A(x)}$	(2.6)	$\frac{A \quad A \supset B}{B}$	(10-11)
$\frac{A \& B}{A}$	(3.1)	$\frac{\neg K \supset X \quad \neg X}{K}$	(10-11)
$\frac{(\exists t)A(x, t)}{A(x, t)}$	(3.1)	$\frac{\neg K \supset \neg S}{S \supset K}$	(10-11)

Note. As with logic operations, logic inference steps may raise questions about their implementation with hardware. One option is to use digital circuits using logic gates (see Matsak 2009).

4 On the logic module of intelligent systems

While studying the use of logic tools by children as (self)evolving natural intelligent systems, the author has proposed a hypothesis about what the corresponding logic module could be. The author does not claim that this type of thing has never been created or researched before. The question in focus is fairly simple: has nature, which can be used as an inspiration for technological development, „designed“ humans with the necessary logic construct processing „machinery“, as described below?

Let us start with Lorents' definition of an intelligent system: we call a system intelligent, if it is capable of operating with knowledge (see Lorents 2002, 2003, 2008). In here, knowledge is defined as an ordered pair $\langle A, B \rangle$, where A and B are sets, where A is the notation (symbol, sign) for B while B is the denotation (meaning) of A (see Lorents 2001, 2004, 2008).

From the aspect of subsistence of intelligent systems it is important to know, which things are the meaning of specific signs or notations and whether it is correct or incorrect that, for example, things with notations x, y, ..., z are related to each other with a relation that has the notation R. In other words, does the intelligent system under observation know, what is the truth value of the atomic formula $R(x, y, \dots, z)$ if the meanings of R, x, y, ..., z are known.

One way to determine the truth value of the formula $R(x, y, \dots, z)$ is the so-called direct check: whether the objects with notations x, y, ..., z in reality have or have not a relation with the notation R. Another way is to deduce the formula from other correct formulas by using correct inference steps (the correctness of the inference steps guarantees the correctness of the resulting formula). It is interesting to note that natural intelligent systems (as observed from the study of children's texts) often rely on so-called secondary methods to check the correctness of formulas that are more complex than atomic:

- by using previously known correct formulas (from a reputable source, checked previously by the actor itself, etc.);
- by using proofs (which guarantee correctness of the result if the premises and inference steps are correct).

The author's studies indicate that when implementing quantifiers or corresponding quantification rules, in reality the finite sets of meanings for the corresponding variables are used. This changes quantifiers into longer or shorter (atomic formula) conjunctions or disjunctions, which, in turn, enables the use of relatively simple digital circuits when operating with these quantifiers (see Figures 3 and 4 below).

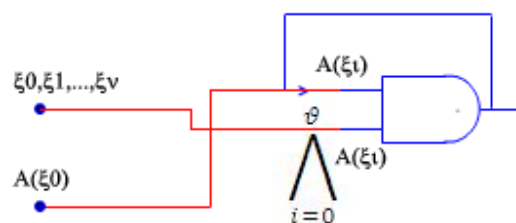


Figure 3 A circuit diagram for the universal quantifier

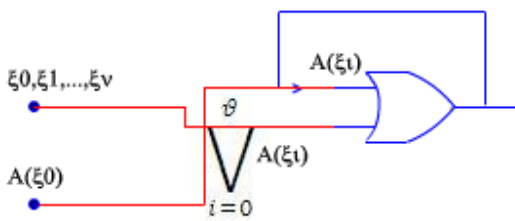


Figure 4 A circuit diagram for the existential quantifier

The logic module of an intelligent system could consist of the following components:

- a binary matrix of notation-denotation (symbol-meaning)

relations, where rows represent notation (symbols, signs) and columns represent denotation (meaning). The intersections contain markers (for example 1 or 0) that indicate that the corresponding notation applies to the denotation (or not). It is important to remember, that according to Lorents (2001, 2004, 2008) some notations may have multiple denotations, and some denotations may have multiple notations;

- the set of correct formulas;
- the set of inference rules.

Note. It may become necessary to modify one or more of the three components (the matrix, formula set or inference rule set). The reason could be changes in the environment, which

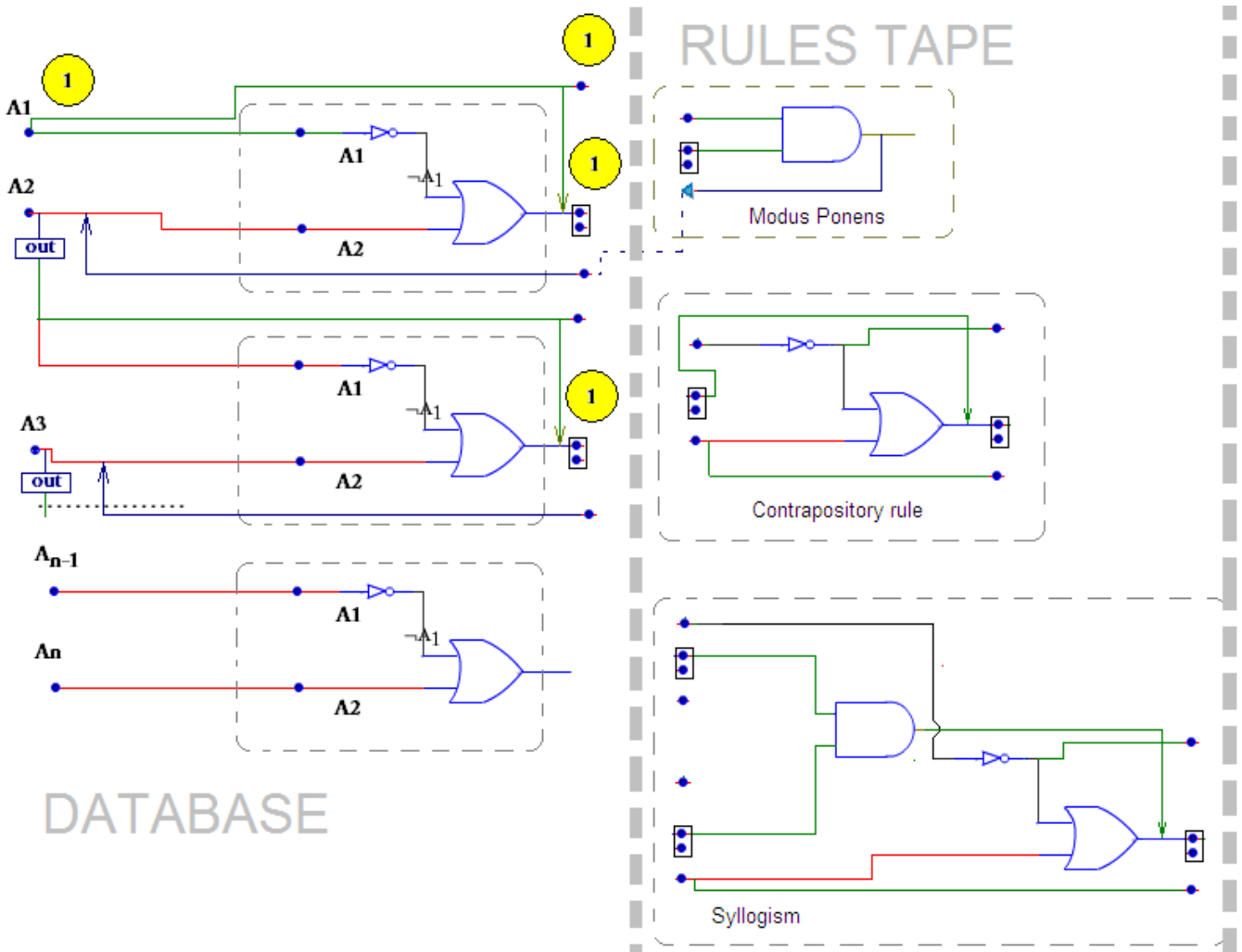


Figure 5 Fitting formulas and inference rules

could result in:

- some notations no longer having the previously recorded denotations. For example, after ten years of marriage the word „mom” may no longer be the primary notation for the man’s mother, but instead of his wife, who is the mother of his children.

- some previously correct formulas no longer being correct. For example, after a few years a boy’s older sister may no longer be taller and stronger.

- new inference steps that can be added to the set. For example, induction as a reasoning technique can be learned in high school.

Implementation of the logic module can be achieved by using relatively well known digital circuits for describing notation-denotation relations and formulas, as well as the inference set circuits developed by the author (see Matsak San Diego). The construction of such a system may take the simple form of „fitting” puzzle +pieces, where the „suitable” premise set of an inference rule allows the rule to be matched to a combination of existing formulas that normally do not involve more than a few formulas (see Figure 5).

5 Conclusion

An essential aspect of the intelligent systems is the ability to apply logical instruments in description of situations and in making correct decisions on this basis. A considerable part of the logical instruments that have been investigated and have been applied in the intelligent systems have been in some way extracted from the texts produced by humans.

However, it is not clear from where the humans get these instruments. It would be interesting to know how much of this capability is given on the so called base software level (at birth); what and when is added later during the development of a person. One has to have answers to these questions in order to be able to develop new intelligent systems with a capability of autonomously adjusting/developing a logic for the needs of a particular domain. We have used the dialogue system DST in finding answers to these questions. We have discovered jumps of the ability to use logic in a child’s development.

Acknowledgements:

The author of the given work expresses profound gratitude to professor Peeter Lorents, professor Enn Tyugu and Rain Ottis from the Cooperative Cyber Defense Centre of Excellence for assistance in writing this paper.

6 References

[1] Bo Haoy, Tomohiro Yoshikaway, Takeshi Furuhashiyz, Shin-ichi Sugiuraz. 2008. "A Development of Early Diagnosis and Hospital Search Support System for Integrate

Medical Support System". International Multi-Conference on Engineering and Technological Innovation, 2008, Orlando, Florida, USA, June 29-July 2.

[2] Child Language Data Exchange System <http://childes.psy.cmu.edu/>

[3] Lorents P, “Keel ja loogika (Language and Logic).” EBS Print. Tallinn, 2000.

[4] Lorents P. “Formalization of data and knowledge based on the fundamental notation-denotation relation”. Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2001. Volume III. p. 1297 – 1301, 2001.

[5] P. Lorents. “A System Mining Framework.” The 6th World Multiconference on Systemics, Cybernetics and Informatics. Proceedings. Vol. 1. 195 – 200, 2002.

[6] Lorents P., Lorents D. “Intelligence and the notation-denotation relation.” Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2003. Vol. II. p. 703 – 707, 2003

[7] Lorents P. “Knowledge and understanding.” Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2004. Volume I p. 333 – 337, 2004.

[8] Lorents P. “The role of equality in knowledge acquisition.” Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2005. Volume II p. 555 – 561. 2005

[9] Lorents P. “Associated knowledge.” Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2006. Volume II p. 537 – 544, 2006.

[10] Lorents P. “Taxonomy of intellect.” Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2008. Volume II p. 537 – 544. 2008.

[11] Lukov V. B., Sergejev V. M. Луков В. Б., Сергеев В. М., „Опыт моделирования мышления исторических деятелей: Отто фон Бисмарк, 1866 – 1876 гг.“ В сборнике Вопросы кибернетики. Логика рассуждений и ее моделирование. АН СССР. Научный Соет по комплексной проблеме “Кибернетика”. Москва. p. 148 – 161, 1983.

[12] Matsak E. “Dialogue system for extracting Logic constructions in natural language texts”. Proceedings of the International Conference on Artificial Intelligence. IC – AI’ 2005. Volume II p. 791 – 797, 2005.

[13] Matsak E. “Using Natural Language Dialog System DST for Discovery of Logical Constructions of Children’s Speech.” The 2006 International Conference on Artificial

Intelligence IC-AI 2006. Las Vegas, Nevada, USA (June 26-29, 2006)

[14] Matsak E. "System DST for Transforming Natural Language Texts, Representing Estimates and Higher Order Predicates and Functionals." The 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2006. . Orlando, Florida, USA. (July 20-23, 2006)

[15] Matsak E. "The prototype of system for discovering of inference rules." Proceedings of the International Conference on Artificial Intelligence. IC – AI'2007. Volume II p. 489 – 492, 2007.

[16] Matsak E. "Improved version of the natural language dialog system DST and its application for discovery of logical constructions in children's speech." Proceedings of the International Conference on Artificial Intelligence. IC – AI'2008. Volume II p. 332 – 338, 2008.

[17] Matsak E. 2009. "Representing logical inference steps with digital circuits." Lecture Notes in Artificial Intelligence: HCI International 2009. Springer, 2009.

[18] Tamisier Thomas. "Cadral: an automated decision-making framework for business collaboration." International Multi-Conference on Engineering and Technological Innovation, Orlando, Florida, USA, (June 29-July 2, 2008)

[19] Tyugu E. "Knowledge - Based Programming." Addison-Wesley. 1988.

[20] Tyugu E. "Algorithms and Architectures of Artificial Intelligence." IOS Press. Amsterdam. Berlin. Oxford. Tokyo. Washington, DC. 2007.

REPRESENTING LOGICAL INFERENCE STEPS WITH DIGITAL CIRCUITS

Erika Matsak

Reprinted with permission, from Lecture Notes in Artificial Intelligence: HCI
International 2009. Springer

Representing logical inference steps with digital circuits

MSc Erika Matsak

Tallinn University, Department of Computer Science, 25 Narva Road, 10120 Tallinn, Estonia

Tallinn University of Technology, Department of Computer Engineering, Raja 15, 12618
Tallinn, Estonia

e-mail: erika.matsak@tlu.ee

Abstract. The use of inference steps in natural language reasoning is observed. An algorithm is presented for representing logically correct inference steps with digital circuits. New foundations for creating decision making systems are studied.

Keywords. Logical inference steps, logic gates, digital circuits representing logical inference steps

1 Introduction

Many decisions that are required for efficient results with modern systems need to be made without human intervention. For example, driving a Mars rover remotely from Earth is not practical because the sensor information from Mars takes tens of minutes to reach Earth and it takes equally long for the steering commands to reach the rover. Another example would be taking defensive action in case of cyber attacks: a human will not be able to understand the situation and make a (informed) decision in a fraction of a second. Therefore, computers must make these necessary decisions. At the same time we want that the computer-made decisions would be at least as reliable as the one that an intelligent person would make (if he/she would be able to do that).

This brings us to the point that the decision making computer must possess logical instruments: logic formulas (for formulating propositions) and logic inference rules (for constructing an argument). A problem in this case is that a number of different logic systems are in use. For example, classical logic is suitable for operating with legal arguments, while intuitionistic logic (see E. Tyugu, G. Mints 1982, 1987) can be used for structural program synthesis. One of the ways to distinguish the various logic systems is to use inference rules and the corresponding inference steps. In information

technology the inference rules (using inference steps to move between formulae) are implemented in software (see C. Chang, R. Lee, 1973, M. Fitting. 1996). However, this does not have to be the only viable option. It is not excluded, in principle, that some of the inference steps could be more efficiently implemented at the hardware level. For this we would first need to develop instruments that allow the separation of logical constructs, such as formulas and inference steps, from natural language (see P. Lorents 2000, E. Matsak 2005, 2006, 2007, 2008). We could then proceed to implement these constructs with digital circuits using logic gates (see W. Kunz, D. Stoffel, 1997).

2 Inference steps and implications in logic gate circuits

Let us agree that within this paper we rely on the classic bivalent logic and that we will stay in the confines of first-order predicate calculation. In this case we can use the fact that each correct inference step corresponds to a correct implication, which has the conjunction of the premises of the inference step as an antecedent and the formula of the conclusion of the inference step as a result (see Lorents 2000):

$$\begin{array}{l} \underline{M \ P \ \dots \ Q} \quad - \text{inference step, (1)} \\ R \\ (M \& P \& \dots \& Q) \supset R \quad - \text{corresponding implication.} \end{array}$$

From this point on the implication representing an inference step will be matched with a two-part digital circuit, where the first part (above the dotted line on drawings) represents the conjunction of the premises of the inference step $M \& P \& \dots \& Q$, and the second part represents the formula R .

Note. In bivalent logic the implication can be replaced by the disjunction of the negation of the antecedent and the result (for example, the implication $X \supset Y$ can be replaced with the disjunction $\neg X \vee Y$). We **did not use** such replacements above! Therefore, for example, the *Modus Ponens* inference step is not represented with the formula $\neg(A \& (\neg A \vee B)) \vee B$, but with a two part circuit, where the first part represents the formula $A \& (\neg A \vee B)$ and the second part represents the formula B .

The solution described above allows for representing inference steps with traditional digital circuits composed of three types of logic gate elements: negation, conjunction and disjunction. As explained previously, each circuit is divided into two parts, where the first part represents the list of premises and the second part represents the conclusion formula. The *use of the inference step* therefore corresponds to moving from the first part of the circuit to the second part. A separate problem in here is creating such circuits as well as suitable visualization software, which was not as simple as it first appeared.

3 An algorithm for using logic gates to design a digital circuit that represents inference steps

While designing a digital circuit we assume that as we move from left to right in the formula all signals must have reached the corresponding gates. In order to guarantee this property, we will change the formula (and sub-formulas) as necessary:

- If the formula contains a conjunction that *is not in parentheses* and immediately before or after it are other operations then the conjunction must be surrounded by parentheses.
- For example we replace the formula $A \vee B \& C \supset D$ with the formula $A \vee (B \& C) \supset D$
- If the formula contains sub-formulas or their negations then we nest the components from left to right in successive parentheses.
- For example we replace the formula $\neg A \& B \& \neg C \& D$ with the formula $((\neg A \& B) \& \neg C) \& D$. We use an analogous process in a formula consisting of only disjunctions.
- If conjunctions (disjunctions) contain sub-formulas of various lengths (including negations or „quantifications” of formulas) then we arrange them from left to right by order of decreasing length (number of symbols).
- For example we replace the formula $(\Delta \vee \Gamma) \& (\neg A \& \Gamma) \vee ((A \& B) \& X)$ with the formula $((A \& B) \& X) \vee (\neg A \& \Gamma) \& (\Delta \vee \Gamma)$.
- We replace implications with applicable formulas consisting of negations, conjunctions and disjunctions. For example we replace the formula $X \supset Y$ with the formula $\neg X \vee Y$.
- If following the rearrangements there is a negation at the right end of the formula then we surround it with parentheses. For example we replace $\Delta \& \neg B$ with $\Delta \& (\neg B)$. If there are two conjunctions or two disjunctions without parentheses at the right end of the formula, then we surround them with parentheses. For example we replace $\Delta \& A \& B$ with $\Delta \& (A \& B)$. Similarly, we replace $\Delta \vee A \vee B$ with $\Delta \vee (A \vee B)$.
- The final change is perhaps the most unusual. We write the negation symbol after the formula in question, not before. For example, we replace $\neg C$ with $C \neg$.

The described changes enable the use of the algorithm in Figure 1.

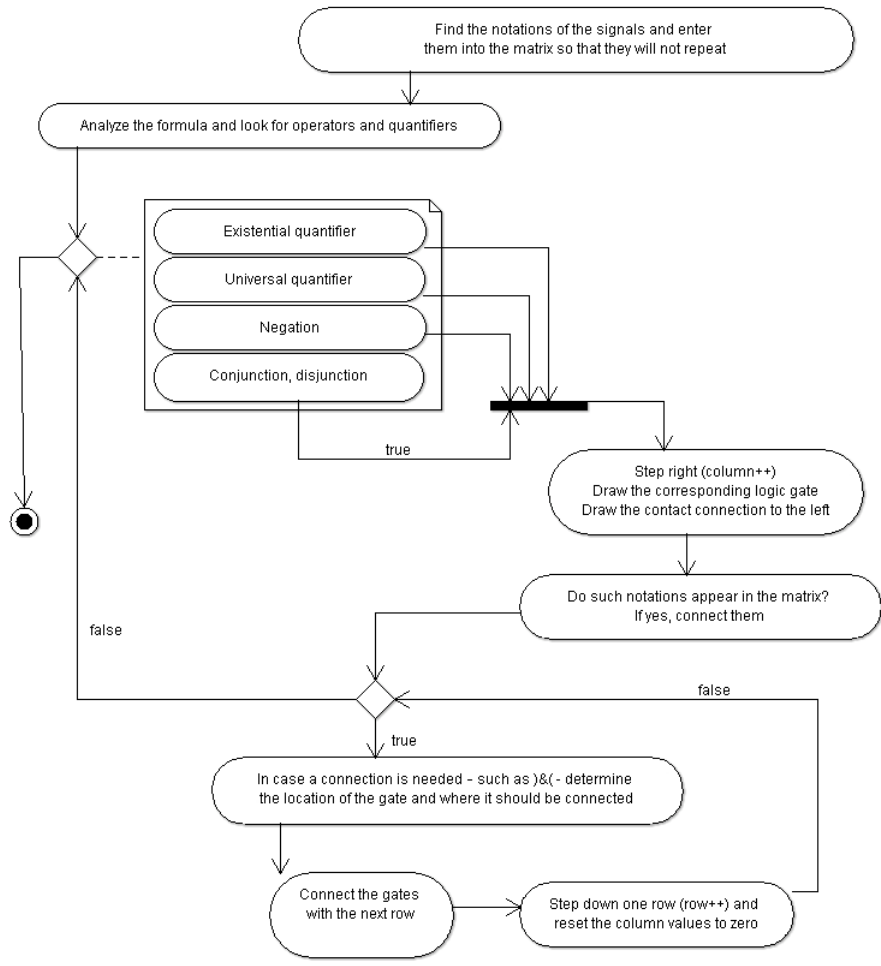


Fig. 1. The algorithm for designing an inference step.

Using the algorithm in figure 1 we get the following circuit for the *Modus Ponens* inference step:

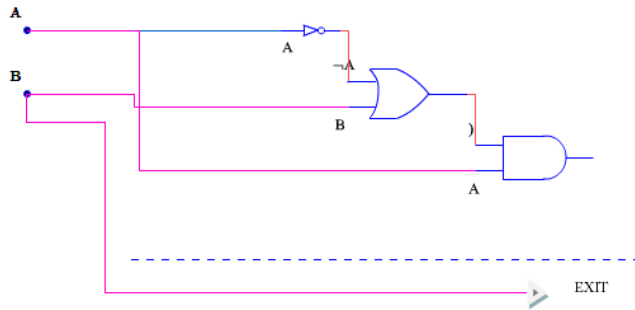


Fig. 2. „Digital“ Modus Ponens.

By introducing the universal quantifier to the rule (see Gentzen 1936)

$$\frac{(A(\beta) \& \Gamma) \supset \Delta}{(\forall \xi A(\xi) \& \Gamma) \supset \Delta} \quad (2)$$

we get the following digital circuit:

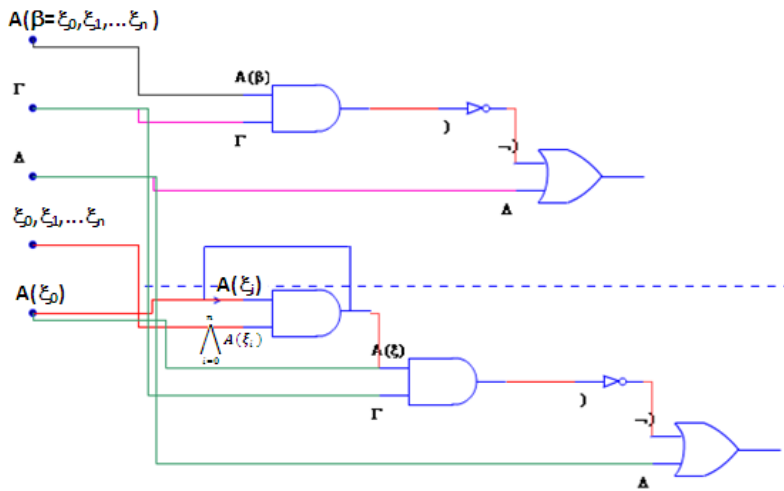


Fig. 3. Digital circuit of the univernal quantifier rule ($\forall^+ \rightarrow$)

4 Circuits in practice

The logic module of decision system could consist of the following components:

- a binary matrix of notation-denotation (symbol-meaning) relations, where rows represent notation (symbols, signs) and columns represent denotation (meaning). The intersections contain markers (for example 1 or 0) that indicate that the corresponding notation applies to the denotation (or not). It is important to remember, that according to Lorents (2001, 2004, 2008) some notations may have multiple denotations, and some denotations may have multiple notations;
- the set of correct formulas;
- the set of inference rules.

Before implementing the inference steps by using digital circuits, the data in the role of predicates must be inserted. In order to achieve this, the relation between formulas and digital logic gates must be established. Since classically there are two possible truth values 1 and 0 (or true and false) and each logic gate also has two values 1 and 0 (or High Voltage (+5V) and Low Voltage (0V)), then it is natural to connect the gates in a way that correct atomic formulas are represented by the signal „1”. Non-atomic formulas should be treated in the following way:

- Identify the part of the circuit that corresponds to the non-atomic formula in question;
- Identify the input points (corresponding to the atomic formulas) for that specific circuit part;
- Identify an input signal combination for the above input gates that produces „1” as an output for that circuit part.

This way we can provide the necessary input signals to the (upper) part of digital circuits, which corresponds to the predicates of the inference step.

The construction of decision may take the simple form of „fitting” puzzle pieces, where the „suitable” premise set of an inference rule allows the rule to be matched to a combination of existing formulas that normally do not involve more than a few formulas.

5 Advantages of the proposed circuits

While creating decision making systems (that are based on, for example, binary decision diagrams (BDD), negation normal form (NNF), propositional directed acyclic graph (PDAG), etc.) data structures related to Boolean functions are often used. The logical operations used to form decisions are simple: AND, OR, NOT. In recent years, several problems have surfaced in solutions relying on neural networks or graphs. This does not mean that these methods should be cast aside (for example, neural networks have advantages in modeling non-linear characteristics of sample data – see Kim D., Lee J. 2001). However, systems based on implementing inference steps with digital circuits also have advantages. One source of these advantages is the ability to include „regular” operations (AND, OR, NOT), as well as other operations (implication, etc.) and quantifiers. Second and more important advantage is the possibility to notably „shorten” the decision making process (it is well known from

logic studies that manipulating with the rules may sometimes allow an exponential (!) decrease in the number of inference steps).

6 Conclusion

The described digital circuits consisting of logic gates are not the only way to represent inference steps. In principle, using special transformations one could implement them in neural networks (see Minsky 1967) or other circuits. The important part here is how to implement logical inference steps in hardware based on the logical constructs extracted from natural language. It is possible that a similar implementation is present in the human brain, which allows us to use logical constructs, including the ability to formulate propositions and to come up with the correct conclusion.

Acknowledgements: The author of the given work expresses profound gratitude to professor Peeter Lorents for assistance in a writing of given clause and to Rain Ottis for English version edition.

7 References

1. W. Kunz, D. Stoffel Reasoning in Boolean Networks: Logic Synthesis and Verification Using Testin Techniques. Kluwer Academic Publishers.(1997)
2. C.Chang, R. Lee.: Symbolic logic and mechanical theorem proving. Academic Press, New York San Francisco London (1973)
3. M. Fitting.: First-Order Logic and Automated Theorem Proving (2nd edition ed.). Springer (1996).
4. Mints G., Tyugu E.: Justification of the structural synthesis of programs. Science of computer programming 2(3), 215-240 (1982).
5. Mints G., Tyugu E.: The programming system PRIZ. Journal of Symbolic Computation, (4). (1987).
6. Lorents P.: Language and logic. EBS Print. Tallinn (2000)
7. Matsak E.: Dialogue system for extracting Logic constructions in natural language texts. Proceedings of the International Conference on Artificial Intelligence. ICAI'2005. Volume II p. 791 – 797. (2005)
8. Matsak E.: Using Natural Language Dialog System DST for Discovery of Logical Constructions of Children's Speech. The 2006 International Conference on Artificial Intelligence ICAI 2006. Las Vegas, Nevada, USA (2006)
9. Matsak E.: System DST for Transforming Natural Language Texts, Representing Estimates and Higher Order Predicates and Functionals. The 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2006. Orlando, Florida, USA. (2006)

10. Matsak E.: The prototype of system for discovering of inference rules. Proceedings of the International Conference on Artificial Intelligence. International Conference on Artificial Intelligence ICAI'2007. Volume II p. 489 – 492 (2007)
11. Matsak E.: Improved version of the Natural Language Dialog System DST and its application for discovery of logical constructions in children's speech. International Conference on Artificial Intelligence ICAI'2008. Volume I p. 332 – 338. (2008)
12. M. Minsky: Finite and Infinite machines. Prentice-Hall, Inc. Englewood Cliffs, N.J. (1967)
13. G. Gentzen: Die Widerspruchsfreiheit der reinen Zahlentheorie. Mathematische Annalen, 112: 493-565. (1936)
14. Lorents P.: Formalization of data and knowledge based on the fundamental notation-denotation relation. Proceedings of the International Conference on Artificial Intelligence. IC – AI' 2001. Volume III. p. 1297 – 1301. (2001)
15. Lorents P.: Knowledge and understanding. Proceedings of the International Conference on Artificial Intelligence. IC – AI' 2004. Volume I p. 333 – 337. (2004)
16. Lorents P.: Taxonomy of intellect. Proceedings of the International Conference on Artificial Intelligence. IC – AI' 2008. Volume II p. 537 – 544. (2008)
17. Kim D., Lee J.: Rule Reduction over Numerical Attributes in Decision Trees Using Multilayer Perceptron. Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2001)

**DISSERTATIONS DEFENDED AT
TALLINN UNIVERSITY OF TECHNOLOGY ON
INFORMATICS AND SYSTEM ENGINEERING**

1. **Lea Elmik**. Informational modelling of a communication office. 1992.
2. **Kalle Tammemäe**. Control intensive digital system synthesis. 1997.
3. **Eerik Lossmann**. Complex signal classification algorithms, based on the third-order statistical models. 1999.
4. **Kaido Kikkas**. Using the Internet in rehabilitation of people with mobility impairments – case studies and views from Estonia. 1999.
5. **Nazmun Nahar**. Global electronic commerce process: business-to-business. 1999.
6. **Jevgeni Riipulk**. Microwave radiometry for medical applications. 2000.
7. **Alar Kuusik**. Compact smart home systems: design and verification of cost effective hardware solutions. 2001.
8. **Jaan Raik**. Hierarchical test generation for digital circuits represented by decision diagrams. 2001.
9. **Andri Riid**. Transparent fuzzy systems: model and control. 2002.
10. **Marina Brik**. Investigation and development of test generation methods for control part of digital systems. 2002.
11. **Raul Land**. Synchronous approximation and processing of sampled data signals. 2002.
12. **Ants Ronk**. An extended block-adaptive Fourier analyser for analysis and reproduction of periodic components of band-limited discrete-time signals. 2002.
13. **Toivo Paavle**. System level modeling of the phase locked loops: behavioral analysis and parameterization. 2003.
14. **Irina Astrova**. On integration of object-oriented applications with relational databases. 2003.
15. **Kuldar Taveter**. A multi-perspective methodology for agent-oriented business modelling and simulation. 2004.
16. **Taivo Kangilaski**. Eesti Energia käiduhaldussüsteem. 2004.
17. **Artur Jutman**. Selected issues of modeling, verification and testing of digital systems. 2004.

18. **Ander Tenno**. Simulation and estimation of electro-chemical processes in maintenance-free batteries with fixed electrolyte. 2004.
19. **Oleg Korolkov**. Formation of diffusion welded Al contacts to semiconductor silicon. 2004.
20. **Risto Vaarandi**. Tools and techniques for event log analysis. 2005.
21. **Marko Koort**. Transmitter power control in wireless communication systems. 2005.
22. **Raul Savimaa**. Modelling emergent behaviour of organizations. Time-aware, UML and agent based approach. 2005.
23. **Raido Kurel**. Investigation of electrical characteristics of SiC based complementary JBS structures. 2005.
24. **Rainer Taniloo**. Ökonoomsete negatiivse diferentsiaaltakistusega astmete ja elementide disainimine ja optimeerimine. 2005.
25. **Pauli Lallo**. Adaptive secure data transmission method for OSI level I. 2005.
26. **Deniss Kumlander**. Some practical algorithms to solve the maximum clique problem. 2005.
27. **Tarmo Veskiõja**. Stable marriage problem and college admission. 2005.
28. **Elena Fomina**. Low power finite state machine synthesis. 2005.
29. **Eero Ivask**. Digital test in WEB-based environment 2006.
30. **Виктор Войтович**. Разработка технологий выращивания из жидкой фазы эпитаксиальных структур арсенида галлия с высоковольтным р-п переходом и изготовления диодов на их основе. 2006.
31. **Tanel Alumäe**. Methods for Estonian large vocabulary speech recognition. 2006.
32. **Erki Eessaar**. Relational and object-relational database management systems as platforms for managing softwareengineering artefacts. 2006.
33. **Rauno Gordon**. Modelling of cardiac dynamics and intracardiac bio-impedance. 2007.
34. **Madis Listak**. A task-oriented design of a biologically inspired underwater robot. 2007.
35. **Elmet Orasson**. Hybrid built-in self-test. Methods and tools for analysis and optimization of BIST. 2007.
36. **Eduard Petlenkov**. Neural networks based identification and control of nonlinear systems: ANARX model based approach. 2007.

37. **Toomas Kirt**. Concept formation in exploratory data analysis: case studies of linguistic and banking data. 2007.
38. **Juhan-Peep Ernits**. Two state space reduction techniques for explicit state model checking. 2007.
39. **Innar Liiv**. Pattern discovery using seriation and matrix reordering: A unified view, extensions and an application to inventory management. 2008.
40. **Andrei Pokatilov**. Development of national standard for voltage unit based on solid-state references. 2008.
41. **Karin Lindroos**. Mapping social structures by formal non-linear information processing methods: case studies of Estonian islands environments. 2008.
42. **Maksim Jenihhin**. Simulation-based hardware verification with high-level decision diagrams. 2008.
43. **Ando Saabas**. Logics for low-level code and proof-preserving program transformations. 2008.
44. **Ilja Tšahhirov**. Security protocols analysis in the computational model – dependency flow graphs-based approach. 2008.
45. **Toomas Ruuben**. Wideband digital beamforming in sonar systems. 2009.
46. **Sergei Devadze**. Fault Simulation of Digital Systems. 2009.
47. **Andrei Krivošei**. Model based method for adaptive decomposition of the thoracic bio-impedance variations into cardiac and respiratory components.
48. **Vineeth Govind**. DfT-based external test and diagnosis of mesh-like networks on chips. 2009.
49. **Andres Kull**. Model-based testing of reactive systems. 2009.
50. **Ants Torim**. Formal concepts in the theory of monotone systems. 2009.