FRAME THEORY AND COMPUTATIONS

Viacheslav E. WOLFENGAGEN

Moscow Engineering and Physics Institute,
Department of Cybernetics, Kashirsky Road
115409 Moscow, U.S.S.R.

Abstract. This paper deals with the integration of the "computational" knowledge representation model and "intensional" problem domain model. The first model is based on the calculi of \( \lambda \)-conversions, the theoretical foundations of LISP-like programming languages. The second model is grounded on the frame theory for the suitable problem domain "situational model". These formal and non-formal (formalized) primitives are built-in tools within the scenario-frame environment for computational purposes of the given problem domain. The introduced frames are in a considerable degree simple objects — they may be constructed by the application and \( \lambda \)-abstraction operators exclusively. The inter-convertibility of the knowledge representation "aspects", their interconnections by means of the convertibility relation and frame expressibility in LISP-like languages, are stressed in the paper.

1. INTRODUCTION

In the present paper some preliminary steps toward the theory of frames are proposed to be taken. In this theory the assumed initial knowledge would be simpler than in commonly used frame based languages.

Frame theory can be defined as an application of formal methods to the knowledge
representation domain that is considered as the analysis and criticism of thought. The essential purpose of the frame theory by this point of view is the construction of a considerably formalized theory such that the proper interpretation of its principal notions correlates with the valid analysis by which thinking goes on. Notion analysis naturally means that a complicated piece of knowledge decomposes into simpler initial pieces (up to the very beginning). The degree of simplicity exhibits how careful and explicit the analysis is.

The main feature of a really abstract theory (of frames) is the following: all the "meaning" has been abstracted. It is supposed, of course, that every object presented to the "mind" has a meaning, and an even partially "meaningless" theory is rejected. In application to the appointed consideration of the "concepts", two kinds of meanings are distinguished: the natural and conventional one. The conventional meaning is based on the relations of a given frame theory, and the natural one is available "within" our previous knowledge.

To be more precise we suppose in accordance with λ-calculus theories [1, 5, 6] that the frame theory begins with a set of primitive notions as follows:

1. Non-formal primitive ideas — entities, modes of combinations, assertions;
2. Formal primitive ideas — applicative programming language notions;
3. Postulates — purely conventional;
4. Rules — involving intuitive ideas that form the methods of "object" transitions (an addition of new ideas by definitions).

The frame theory itself may now be defined as a kind of integration of the "computational" knowledge representation model and "intensional" problem domain model. The first model is based on the calculi of λ-conversions — the theoretical foundation of LISP-like programming languages. The second model is grounded on the frame theory for the suitable problem domain "situational model". These formal and non-formal (formalized) primitives are built-in tools within the scenario-frame environment for computational purposes of the given problem domain. The introduced frames are in a considerable degree simple objects — they may be constructed by the application and λ-abstraction operators exclusively. The interconvertibility of the knowledge representation "aspects", their interconnections by means of the convertibility relation and frame expressibility in LISP-like languages are stressed in the paper.

Let us now turn to the traditional means of the artificial intelligence sphere. Consider, from the λ-calculus point of view, the following topics.

2. SEMANTIC NETWORKS

To model the semantics of data base, the information structure referred to as semantic network [9, 11, 3, 10] has been used extensively in various applications. The shortest definition of semantic network (SN) is the following: semantic networks are basically directed labelled graphs. Their most important feature is the organization they impose upon factual knowledge which is considerably more natural and structured than the other knowledge representations. According to the psycholinguistic theory, this kind of structuring models the associative linkages that humans use in their factual derivations. It also would be more difficult to possess the derivation of the far located facts from the particular fact in
a fixed sphere of knowledge, than other whose "distance" is smaller. The proposed
knowledge organization, although being originated by a hypothesis to be verified or
diversified, still provides the valuable and effective computational means.

Another attractive and important semantic networks feature is their clearness, sensibility
and aesthetic appeal. However, the theoretical study states that semantic networks make no
contributions and usually the graph representation has to be transformed to more suitable
and conventional data structures: strings, lists, matrices, etc. The practical consideration
exposes the very knowledge representation action by constructing a semantic network. It
contributes the detecting inconsistencies, missing data and structures, etc.

Upon the promising results the semantic networks researchers have faced some discom-
fort. Some researchers [19, 7, 2, 12] have criticized the semantic network constructs basically
for their interpretation ways. The main aspect under criticism is closely connected with
a "formal correctness" of semantic networks because there were no rules, neither for their
construction nor for evaluation. Usually these rules are intrinsic interpreter ability that is
implemented heuristically and poorly supplied by any kind of comments.

This paper deals with the original SN-definition expansion to obtain the representational
facilities of large "knowledge units". The "knowledge units" are invoked by "frame" [8] or
"script/plan" [12] notions. The expansion method guarantees the SN-application not only in
particular implementations but also for modelling various problem domains. The discussed
SN-model has been chosen to support the "case-grammar" predicate representation. We
distinguish the atomic SN-components: concepts and simple frames.

3. SITUATION MODEL

The semantic network is a graph containing the labelled nodes and edges that represent all
objects and correspondencies of a particular domain under consideration. Each element of
a graph is a (possible empty) set of nodes, edges, and graphs. The particular edge can
connect two separate nodes or two separate graphs. Any kind of "semantics" is indicated by
a label and an edge direction. Let us study two primitives: concepts and simple frames.

Now we consider and distinguish three kinds of concepts. Each of the concepts is assigned
to a different kind of SN-nodes.
1. Generic concepts (types) indicate the generic facts of physical or abstract objects
discovered and identified in a problem domain by a human. e.g. "person", "supplier",
"bureau", etc. The intensional objects are indicated by generic concepts and are supported
and interpreted as data sets.
2. Typed variables represent the problem domain sets. Every variable has a type assignment,
 i.e. it is linked to an SN-generic concept by a "t"-labelled edge leading from the variable to
the generic concept (Fig. 1a).
3. Typed constants represent the instantiations of evaluated generic concepts, e.g. "Smith"
and "bureau-1" indicate the supplier value or bureau value respectively. Every constant has
a type assignment, i.e. it is linked to an SN-generic concept by an "t"-labelled edge
(Fig. 1b). The direction of an edge is from the constant to the generic concept.
Intuitive reason of a simple frame (event-frame) is trivial. There is an underlined object, e.g. object $a$. It contains a "predicate" symbol gathering its arguments. Upon evaluation the frame represents the truth-valued proposition. The evaluation function makes it possible to assign the problem domain constants to the predicate arguments resulting in a proposition that has the truth value. The substitution which results in a true proposition is mentioned as the "instantiation" of frame. The set of all such substitutions is referred to as the "extension" of frame. One of the most prominent approaches to obtaining the frame theory is connected with the calculi of $\lambda$-conversion [5, 6, 4, 13, 14—18]. Fig. 1c and Fig. 1d show simple frames. Their arguments (substitutional variables) are bounded by an abstraction operator $\lambda$.

"Cases" or "roles" are referred to indicate the relationships between the predicate and its arguments. The event-frame network representation consists of a special node that indicates the event predicate occurrence. The "case"-labelled edges point to the nodes corresponding to the arguments of the predicate from the event-frame node. These arguments are either typed variables or typed constants of generic concepts. This frame is like a model or "description" of a situation because no assertion has been done about the event that is encoded by the predicate symbol (the variables are bounded by $\lambda$-operator). However, upon assigning constants of corresponding types from a real problem domain to all the frame variables, the resulting situation is a truth valued "proposition". The event arguments are
indicated by case-labeled edges. These edges specify the occurrence of every argument
within the event framework. This is the main observation that sets up distinguishing between
"roles" and "propositional" arguments of ordinary predicates used in logic. The roles
considered in the following are:

source of action \((a)\) — the object location before the action;
object \((o)\) — the thing that is under action;
destination \((l)\) — the object location upon the action;
intermediator \((p)\) — the position or the state which is crossed by the object during
the "navigation" from source to destination.

It may be shown that the given role set has a sufficient generality. However,
a simplification of the situational analysis in the specified problem domains makes it possible
to use some other roles (it leads to taking into account characteristic frames, functional
frames, etc. [10]). But in this paper we would not like to follow the traditional manner [11,
10, 7, 8] which leads to a simple typed theory (STT) and to studying its interpretations —
this way has been sufficiently investigated. Instead of the well known traditional
consideration we pay attention to studying the frame "interconversions" and to the
problems of "computational environment" arising within this framework. This con-
ideration, as will be shown, leads to the new notion of the semantic network and to the new
notion of the frame which allows to represent the knowledge associated with the (user)
language. Suppose that the language is the tool of a problem domain description but not
exclusively the "conceptual" modelling one.

4. CONCEPTUAL KNOWLEDGE REPRESENTATION FRAMES

Consider the following kinds of frames.

4.1. Simple event frames. Let \(\mathcal{X}\) be a nonempty set of variables types and every \(\tau \in \tau\) have
a countable set of variables, \(\forall \tau : x_1, x_2, \ldots\); let \(\mathcal{C}\) be a set of \(\tau\)-typed constants for all \(\tau \in \mathcal{X}\). \(\mathcal{B}\)
be a set of the predicate symbols \(P_{\alpha, \tau}\), where \(P_{\alpha, \tau}\) has a type \((\tau_1, \ldots, \tau_n), \ i = 1, \ldots, m, \ldots\).

Suppose that if \((\tau_1, \ldots, \tau_n)\) are the types, then \((\tau_1, \ldots, \tau_n)\) is a type.
Define a collection of all the terms having the type \(\tau\) (denoting it by \(\mathcal{T}_m\)) as follows:
(a) if \(c' \in \mathcal{C}\), then \(c'\) is a term of type \(\tau\);
(b) if \(x' \in \forall \tau\), then \(x'\) is a term of type \(\tau\);
(c) if \(s \in \mathcal{T}_{m_{(\alpha, \tau)}}\) and \(t \in \mathcal{T}_{m_{\tau}}\), then \((st) \in \mathcal{T}_m\);
(d) if \(y \in \forall \tau\) and \(s \in \mathcal{T}_{m_{\tau}}\), then \((\lambda y \cdot s) \in \mathcal{T}_{m_{(\alpha, \tau)}}\), and \(y\) becomes a bounded variable.

The collection of all terms is denoted by \(\mathcal{T}_m\). A simple event frame is defined as follows.
Let \(\alpha \in \mathcal{B}\) or \(\alpha \in \forall \tau_{(s_1, \ldots, s_m)}\) (\(\alpha\) is a free variable) and \(t_i \in \mathcal{T}_{m_{\tau}}, i = 1, \ldots, n\). Then \(\alpha t_1, \ldots, t_n\)
is a simple event frame (sentence, \(\text{Sent}\), atomic formula, etc.). A more traditional notation is
established by \(\alpha (t_i, \ldots, t_n) \equiv \alpha t_1, \ldots, t_n\) (the "\(\equiv\)" symbol is read as "is equal by the
definition"). The problem of the frame graphical representation has already been discussed.
4.2. Characteristic frames model properties or situations. They are similar to the event frames except that they use the edges labelled by "characteristic of" (abbr. "ch") and "value of" (abbr. "v"). The "ch"-labelled edges lead to the argument that is being characterized, but the "v"-labelled edge leads to the argument that specifies the value of characteristic.

4.3. "Logical predicates" or functional frames simulate the problem domain predicates and functions. They have the same structure as event frames. The only exception is the labelling of edges by the "argument 1" (abbr. "arg 1"), ..., "argument b" (abbr. "arg n"), and "result" (abbr. "r") cases.

4.4. Introduction of quantification. Many of the known semantic network notions do not provide quantification. The complicated forms of quantification are supposed to be handled by the SN-interpreter. However, the simpliest query: "to give all the transport bureaux that ship all the auto parts to at least 4 cities of the RSFSR" involves simultaneously three quantifiers, one of which indicates cardinality.

The difficulties of quantification concerning semantic networks are well known. The task is much more complicated because of quantified sentence ambiguity in the case of those systems that pretend language "understanding" (the so called "linguistic knowledge representing systems"). The completely non-trivial mechanism is always needed to omit such representation ambiguity. Introducing the quantification, we present the main premises as follows:

(i) Denoting the scopes of quantifiers and of the intrinsic argument dependencies.
(ii) The quantifier representation independency of any kind of evaluation that may be assigned later.
(iii) Facilitating the quantification semantics of the "improvement" ability.

According to WOODS [19] there are three ways of introducing the quantifiers in semantic networks:

(a) the quantifier nodes technique;
(b) Skolem functions technique;
(c) the \( \lambda \)-calculus technique.

In using the first and second ways of introducing the quantification WOODS has discovered hard problems originating in particular from the choice of edge labels. He believes that the third technique is a variant of the first one. In contrast with WOODS, we shall follow the third way, setting up the "computational frame" notion for the "linguistic knowledge" representation, and describe a simple technique of the "quantifiers" representation. It is based upon the following consideration. The quantifiers are not the original objects within the \( \Xi \)-system of conversions, but such objects may be introduced in it that simulate the "quantifiers" in the sense of their introduction rules [5, 6]. In that case we pin our hopes on constructing, within a \( \Xi \)-system, the objects that behave themselves like the natural language quantification forms. In reality, as it will be shwon in this paper, even without using a \( \Xi \)-operator, it is possible, within the \( \lambda \)-conversion system framework, to represent the frames with rather interesting and promising quantification forms.
5. LAMBDA-CONVERSION VS. SEMANTIC NETWORK STRUCTURE

Now we restrict our consideration to the standard introduction of quantifiers using, for example, the following definition.

Let $A$ be a simple frame, $a'$ a free variable, $x'$ a variable having the same type as $a'$ but not occurring in $A$, and let $A'$ be obtained from $A$ by replacing all the occurrences of $a'$ in $A$ with $x'$. Then $\forall x' A'$ and $\exists x' A'$ are the frames. (N.B. The notation $\forall x' \cdot A'$ and $\exists x' \cdot A'$ is used within the $\lambda$-conversion system. To distinguish both notations is beyond the scope of this paper.)

Let us call the collection

$$PS \rightleftharpoons \langle \{ T^o \}, \{ A_m \} \rangle$$

$SN$-prestructure (Semantic Network prestructure) if $T^o$ is a nonempty set for all $\sigma \in \mathcal{X}$ (i.e. $T^o \rightleftharpoons \text{Dom}(\sigma)$) and

$$A_m : T^{o, \tau} \times T^o \rightarrow T^o$$

for each type symbol $\sigma, \tau \in \mathcal{X}$. We require the following extensionality condition: if $x, y \in T^{o, \tau}$ and for each $z \in T^o$

$$A_m(x, z) = A_m(y, z),$$

then $x = y$.

Let us extend the frame language under consideration by adding to every $\tau \in \mathcal{X}$ all the domain elements $T^\tau$ to be the new $\tau$-typed constants. An assignment (Kripke-like “possible world”) in a system $PS$ is a function $h$ whose domain is the set of all variables and such that $h(x^z) \in T^o$. The set of all assignments is denoted by $\mathcal{ASg}$ (the collection of “possible worlds”).

An $SN$-structure is a system

$$S \rightleftharpoons \langle \{ T^o \}, \{ A_m \}, \mathcal{Val} \rangle,$$

where $\langle \{ T^o \}, \{ A_m \} \rangle$ is a prestructure and the mapping (evaluation) $\mathcal{Val}$ is defined as follows:

$$\mathcal{Val} : \mathcal{Fm} \times \mathcal{ASg} \rightarrow \bigcup_{\sigma \in \mathcal{X}} T^\sigma,$$

such that the following clauses hold:

1) $\mathcal{Val}(x^o, h) = h(x^o), \ x^o \in \mathcal{Vr}_o, \ h \in \mathcal{ASg}$;
2) $\mathcal{Val}(\langle s, t \rangle, h) = A_m(\mathcal{Val}(s, h), \mathcal{Val}(t, h))$, where $s \in \mathcal{Fm}_{m-1}, \ t \in \mathcal{Fm}_m$;
3) for all $a \in T^o$

$$A_m(\mathcal{Val}(\langle \lambda x \cdot s \rangle, h), a) = \mathcal{Val}(s, h^*_x),$$

where $s \in \mathcal{Fm}_m, \ x \in \mathcal{Vr}_o$ and $h^*_x$ is given by
\[ h^*_y := \begin{cases} h(y) & \text{for } y \neq x; \\ a & \text{for } y = x. \end{cases} \]

Suppose the frame truth valued set is a Boolean algebra \( \mathcal{B} = (B, \leq) \). Then each \( n \)-ary predicate symbol \( P \in \mathcal{B} \) is associated with the \( n \)-ary function

\[
\hat{P} : T^n \times \ldots \times T^n \rightarrow B, \\
Val(P, h) := \hat{P}.
\]

We now define a mapping that associates the value with the closed terms. Suppose \( \Val(a, h) := a \) if \( a \in T^* \).

1) \( \Val(P(t_1, \ldots, t_n), h) = \hat{P}(\Val(t_1, h), \ldots, \Val(t_n, h)) \);
2) \( \Val(\forall x.A, h) = \wedge \{ \Val([q/x]A, h) | q \in T^*, x \in \forall \} \);
3) \( \Val(\exists x.A, h) = \vee \{ \Val([q/x]A, h) | q \in T^*, x \in \exists \} \);
4) \( \Val(\lambda x_1, \ldots, x_n : A, h) \) is a function \( \Phi \)

\[
\Phi : T^n \times \ldots \times T^n \rightarrow B, \\
x_i \in \forall \text{, such that for all } q_i \in T^a \\
\Phi(q_1, \ldots, q_n) := \Val([q_1, \ldots, q_n/x_1, \ldots, x_n]A, h).
\]

The value of \( \Val(\lambda x_1, \ldots, x_n : A, h) \) is defined if all the values \( \Val([q_1, \ldots, q_n/x_1, \ldots, x_n]A, h) \) are defined and in addition \( \Phi \in T^* \), where \( \tau := (\tau_1, \ldots, \tau_n) \). Let us not mention the introduction of connectives "AND", "OR", "NOT" into the language and their interpretation on the S-structure. Instead of this subject we consider the semantic network theory in terms of the pure typed \( \lambda \)-conversion. The following chapters deal with the construction of computational model that uses the \( \lambda \)-conversion theory. This computational model captures the problem of the user language reality knowledge representation when data base management system (DBMS) is used in the interactive mode.

5.1. **Alphabet:** \( \odot (.) \lambda \Box \mid \text{conv} \)

5.2. **Type symbols:** If \( \sigma, \tau \) are nonempty words in the alphabet \( \odot \), then \( (\sigma, \tau) \) is a type symbol. The symbol \( \odot \) is by definition the type symbol itself. The type symbols are denoted as follows: \( \text{Sent} \) (sentence), \( \text{Tm} \) (term), \( \text{rel} (\alpha, \beta) \) (relator), possibly with the indices.

5.3. **Variables:** If \( \alpha \) is a nonempty word in the alphabet \( \Box \), then \( [\alpha] \) is a variable; variables are denoted by the letters \( x, y, z, R, T, P, r, s, t \), possibly with indices. Every type symbol is assigned to the (countable) set of variables. Let us write "type" instead of "type symbol". Then the variable with the assigned type is written as \( x; \sigma \). In those cases when the type assignment is available from the context without any difficulties, the explicit type assignment will be omitted.

5.4. **Objects:** The variables are objects. Suppose \( a \) and \( b \) are the objects. Then \( (ab) \) and \( (\lambda x a) \) are the objects, too. The first eight Roman alphabet letters are used for denoting the objects (the use of indices is allowed).
Variables in the objects are bounded only by the $\lambda$-abstraction operator. The fact that the variable and object set $x_1, ..., x_n, a, ..., a_m$ contains the object $a$, and the variable $x_i$ that has free occurrences in $a$ (i.e. $x_i$ is not bounded in $a$ by $\lambda$-operator) will be indicated by $x_i \in a$.

The expression $x_i \in a$ means that $x_i$ has no free occurrences in $a$. The notation generalization is $x_1, ..., x_n \in a, ..., a$, $(x_1, ..., x_n \in a, ..., a$, respectively). It will be assumed that "$ab(cde)$" abbreviates ""$((ab)(cd)e)$"" and that omitted parentheses are to be inserted as far to the left as possible. It will also be assumed that "$\Rightarrow$" abbreviates "is equal by the definition", "$\equiv$" abbreviates "is equal graphically".

$$\lambda x \cdot a \equiv \lambda xa; \quad \lambda(x_1, ..., x_n) \cdot a \equiv \lambda x_i(\lambda(x_1, ..., x_n) \cdot a), \quad n > 1.$$ The result of substituting the object $b$ in the object $a$ in place of the variable $x$ is denoted by $[b/x]a$.

5.5. Combinators:

- $I \equiv \lambda x \cdot x,$
- $K \equiv \lambda xy \cdot x,$
- $B \equiv \lambda xyz \cdot x(yz),$ $C \equiv \lambda xyz \cdot xzy,$
- $B_1 \equiv \lambda xzw \cdot x zw,$ $C_1 \equiv \lambda xzw \cdot xzwy,$
- $B_{11} \equiv \lambda xzw \cdot xzw,$ $C_{11} \equiv \lambda xzw \cdot xzy,$
- $\mathcal{E}.$

5.6. Terms of the typed $\lambda$-calculus: The terms $d$, their types, their sets of free variables $\mathcal{F}_v(d)$, and their sets of bound variables $\mathcal{B}_v(d)$ are given by

1) $x_n : \delta$ is a term of type $\delta$,

$$\mathcal{F}_v(x_n) = \{x_n\}, \quad \mathcal{B}_v(x_n) = \emptyset;$$

2) if $d$ is a term of type $(\delta, \Delta)$, $e$ a term of type $\delta$,

then $(de)$ is a term of type $\Delta$,

$$\mathcal{F}_v((de)) = \mathcal{F}_v(d) \cup \mathcal{F}_v(e), \quad \mathcal{B}_v((de)) = \mathcal{B}_v(d) \cup \mathcal{B}_v(e);$$

3) if $d$ is a term of type $\Delta$, $y$ a variable of type $\delta$,

then $\lambda y d$ is a term of type $(\alpha, \Delta)$,

$$\mathcal{F}_v((\lambda y d)) = \mathcal{F}_v(d) - \{y\}, \quad \mathcal{B}_v((\lambda y d)) = \mathcal{B}_v(d) \cup \{y\}.$$  

5.7. Postulates of $\lambda$-conversion:

$$\begin{align*}
\text{(a):} & \quad \lambda x \cdot a \text{ conv } y/x/a; \quad y \in \mathcal{F}_v(a) \cup \mathcal{B}_v(a); \\
\text{(b):} & \quad (\lambda x \cdot a)b \text{ conv } [b/x]a; \quad \mathcal{B}_v(a) \cap \mathcal{F}_v(b) = \emptyset; \\
\text{(m):} & \quad \frac{a \text{ conv } b}{ca \text{ conv } cb}; \quad \frac{a \text{ conv } b}{ac \text{ conv } bc}; \\
\text{(e):} & \quad \frac{a \text{ conv } b}{\lambda x \cdot a \text{ conv } \lambda x \cdot b}; \quad \text{(q):} a \text{ conv } a;
\end{align*}$$
(σ): \[ a \text{ conv } b \quad ; \quad b \text{ conv } a \]
(τ): \[ a \text{ conv } b \quad ; \quad b \text{ conv } c \quad ; \quad a \text{ conv } c \]
(η): \[ \lambda x \cdot bx \text{ conv } b \quad ; \quad x \in \mathcal{F}_v(b) \]

Note that the postulates of the reflexivity (σ), the symmetry (σ), and the transitivity (τ) of
the convertibility relation allow to manipulate it as the equivalence relation (Fig. 2a—d).

![Diagram of convertibility relations](image)

Fig. 2. Graphical representation of the convertibility.

a) The α postulate (full notation).
b) The α postulate (abbreviated notation).
c) The β postulate (full notation).
d) The β postulate (abbreviated notation).

As both postulates (σ), (τ), (η) and (ξ) are available, the arising equivalence will be
called ηξ-equivalence. It is denoted by "=" , i.e. symbols "=" and "conv" can be used at
a time as the synonyms.

The aim of the following chapter is to fill the gap between the LISP formalization and the
models of situations in order to introduce the notion of multilevel finite sequences within the
λ-calculus.

6. FINITE SEQUENCES

Given an arbitrary sequence of objects, to each object we assign the variable (of the
appropriate type). The problem is the following: to build the object \( A \) which is assigned to
the sequence \( x_1, x_2, \ldots, x_n \) for the fixed natural number \( n \geq 0 \), i.e. \( A \equiv \langle x_1, x_2, \ldots, x_n \rangle \).
Continuing the reasoning, we introduce the object \( LENGTH \) which, upon applying it to \( A \),
results in the length of the sequence, i.e. \( LENGTH \ A = n \). The sequence whose length is
zero, is denoted by \( \langle \rangle \) or NIL. Let us consider the choice of a certain set of operators to
make the construction and the manipulation of finite sequences possible.

1. In order to construct from an atom a sequence of length one, the object (operator) \( UNIT \)
must be represented:

\[
UNIT \ x = \langle x \rangle .
\]
2. An associative operator \textit{APPEND} has to concatenate two sequences. Its infixed denotation is \texttt{\textunderscore\textunderscore\textunderscore}.

3. The operator \textit{CAR} is needed to extract the first component from a sequence:
\[ \text{CAR} \langle x_1, \ldots, x_n \rangle = x_1, \quad n > 0. \]

4. The operator \textit{CDR} is needed for erasing the first component from a sequence:
\[ \text{CDR} \langle x_1, \ldots, x_n \rangle = \langle x_2, \ldots, x_n \rangle, \quad n > 0. \]

5. The operator \textit{NULL} is introduced in order to discriminate the empty sequence:
\[ \text{NULL} \langle \rangle = 1; \quad \text{NULL} \langle x_1, \ldots, x_n \rangle = 0, \quad n > 0. \]

It has been assumed that \( A, B \) and \( C \) are sequences,
\[ A \equiv \langle a_1, \ldots, a_n \rangle, \quad B \equiv \langle b_1, \ldots, b_m \rangle, \quad C \equiv \langle c_1, \ldots, c_p \rangle, \]
where \( n, m, p \geq 0 \) and \( A \land B \equiv \langle a_1, \ldots, a_n, b_1, \ldots, b_m \rangle \), \( 0 \) and \( 1 \) are the numerals, \( 0 \equiv \mathcal{K} \mathcal{K} \), \( 1 \equiv \mathcal{S} \mathcal{B} (\mathcal{K} \mathcal{S}) \), i.e. \( 0 \equiv \lambda x y \cdot y, \quad 1 \equiv \lambda x y \cdot x y \).

7. \textbf{COMPUTATIONAL MODEL — THE MAIN PROPERTIES}

Summing up, we consider the operator set that was mentioned above:
\[ \{ \text{APPEND}, \text{NIL}, \text{UNIT}, \text{CAR}, \text{CDR}, \text{NULL} \} \]

and characterize it by the set of axiom-schemes as follows:
1. \textit{APPEND} \( a \) (\textit{APPEND} \( b \ c \)) = \textit{APPEND}(\textit{APPEND} \( a \ b \)) \( c \);
2. \textit{APPEND} \( \text{NIL} \ a \) = \textit{APPEND} \( a \ \text{NIL} \);
3. \textit{CAR}(\textit{APPEND}(\textit{UNIT} \ a) \ b) = a;
4. \textit{CDR}(\textit{APPEND}(\textit{UNIT} \ a) \ b) = b;
5. \textit{NULL} \( \text{NIL} \) = 1;
6. \textit{NULL}(\textit{APPEND}(\textit{UNIT} \ a) \ b) = 0,

where \( a, b, c \) are arbitrary objects (in particular, variables).

Let us prove that the axiom-schemes 1—6 are admissible in the \( \lambda \)-calculus.

\textbf{Proposition.} (Embedding the computational model into the \( \eta_5 \)-calculus of \( \lambda \)-conversions). The axiom-schemes 1—6 are derivable in the \( \lambda \)-calculus with the postulates (\( \alpha \)), (\( \beta \)), (\( \mu \)), (\( \iota \)), (\( \xi \)), (\( \eta \)), (\( \kappa \)).

\textbf{Proof.} Let us present the correspondence between the computational model (\textit{CM}) operators and the \( \lambda \)-calculus objects (Table 1). Now prove by axiom analysis that this correspondence is valid.
1. \( \mathcal{A}(\mathcal{B}bc)d = a(\mathcal{B}bcd) = a(b(cd)) = \mathcal{B}ab(cd) = \mathcal{B}(\mathcal{R}ab)cd. \) Let \( d \) be a variable having no occurrence in \( a, b, c \). Then \( \lambda d \cdot \mathcal{A}(\mathcal{B}bc)d = \lambda d \cdot \mathcal{B}(\mathcal{R}ab)cd \) (by (\( \xi \))) and \( \mathcal{A}(\mathcal{B}bc) = \mathcal{B}(\mathcal{R}ab)c \) (by (\( \eta \)), (\( \iota \))).
Table 1.
Correspondence between the CM-operators and the λ-calculus objects.

<table>
<thead>
<tr>
<th>CM-operator</th>
<th>APPEND</th>
<th>NIL</th>
<th>UNIT</th>
<th>CAR</th>
<th>CDR</th>
<th>NULL</th>
</tr>
</thead>
<tbody>
<tr>
<td>λ-cal. object</td>
<td>(\mathcal{B})</td>
<td>(\mathcal{I})</td>
<td>(\mathcal{E}_{127}\mathcal{I} \Rightarrow \mathcal{B})</td>
<td>(\mathcal{E}_{127}\mathcal{I}a\mathcal{K})</td>
<td>(\lambda xy \cdot xy0)</td>
<td>(\mathcal{E}_{127}\mathcal{I}0(\mathcal{K}(\mathcal{I}0)))</td>
</tr>
</tbody>
</table>

2. For an arbitrary variable \(\text{zēa}\)

\[\mathcal{B}a\mathcal{I}z = a(\mathcal{I}z) = az = \mathcal{I}(az) = \mathcal{B}a\mathcal{I}za\]

and hence \(\mathcal{B}a\mathcal{I} = \mathcal{B}a\mathcal{I}a\).

3. \(\mathcal{B}a\mathcal{K}(\mathcal{B}(\mathcal{D}b)c) = \mathcal{B}(\mathcal{D}b)ca\mathcal{K} = \mathcal{D}b(ca)\mathcal{K} = \mathcal{K}b(ca) = b\).

4. \((\lambda xy \cdot xy(\mathcal{K}I))(\mathcal{B}(\mathcal{D}a)b)z = \mathcal{B}(\mathcal{D}a)bz(\mathcal{K}I) = \mathcal{D}a(bz)\mathcal{K}I = \mathcal{K}Ia(bz) = \mathcal{I}bz = bz\),

where \(z\) is a variable, \(\text{zēa}\), \(b\).

5. \(\mathcal{D}(\mathcal{K}I)(\mathcal{K}0(\mathcal{K}I)) = \mathcal{I}(\mathcal{K}I)(\mathcal{K}0(\mathcal{K}I)) = \mathcal{K}I(\mathcal{K}0(\mathcal{K}I)) = \mathcal{I} = \lambda xy \cdot xy = 1\), where \(\mathcal{K}^{+1}\)

\(a = \mathcal{K}(\mathcal{K}a)\), \(i > 0\); \(\mathcal{K}^i \neq \mathcal{K}\), \(\mathcal{K}^0 a = a\).

The last two conversions are justified as follows: \(x = \lambda y \cdot xy\) (by (η)) i.e. \(i = 1\). Note that this proof was originated by postulates (ξ), (η) of \(\lambda\)-conversion.

6. \(\mathcal{D}0(\mathcal{K}0)(\mathcal{B}(\mathcal{D}a)b) = \mathcal{B}(\mathcal{D}a)b0(\mathcal{K}0) = \mathcal{D}a(b0)(\mathcal{K}0) = \mathcal{K}0a(b0) = \mathcal{B}a\mathcal{K}0a(b0) = \mathcal{K}0a(b0) = \mathcal{K}0(b0) = 0\).

The proof of the proposition (embedding the CM) is completed by the six conversions (ηι- conversions) having been established.

Let \(\ast\) be an infixed version of the composition operator:

\[a^{\ast\ast}b \Rightarrow a(\ast b),\]

where \(i \geq 0\), \(a^{\ast}b \Rightarrow b\). It is easy to conclude that the applicative version of the composition operator is \(\mathcal{B}\). In addition to Table 1 we assume the following.

1. Numerals: \(0 \Rightarrow \mathcal{K}\mathcal{I}\); \(1 \Rightarrow (\mathcal{I}\mathcal{B})\mathcal{K}\mathcal{I}\); \(n \Rightarrow (\mathcal{I}\mathcal{B})^n(\mathcal{K}\mathcal{I})\),

i.e. \(n \Rightarrow \lambda xy \cdot x^n y\).

2. Successor function: \(\sigma \Rightarrow \lambda xyz \cdot xy(zy)\).

3. Length function:

\[\text{LENGTH} \Rightarrow \lambda xy \cdot \text{NULL} x 0 (\sigma(\text{LENGTH}(\text{CDR} x)) y)\].

Let \(\mathcal{R}\) be the finite sequences set taking into account the following objects:

4. Abstract pair function:

\[\mathcal{D} \Rightarrow \lambda yz \cdot zxy \Rightarrow \lambda y \cdot [x, y]\].

5. Sequence:

\[A \Rightarrow \langle a_1, \ldots, a_n \rangle = \mathcal{D}a_1 \ast \cdots \ast \mathcal{D}a_n = \lambda z \cdot [a_1, \ldots, [a_n, z] \ldots] \]

\[\text{if } n > 0; \quad A \Rightarrow \mathcal{I} \text{ if } n = 0\].

If \(Ob\) is the \(\lambda\)-calculus set of objects, then \(\mathcal{R} \subseteq Ob\) is defined as follows.
**Definition.** The finite sequences set $\mathcal{R}$ is the smallest (minimal) object set $\mathcal{O}b_{\mathcal{R}} \subseteq \mathcal{O}b$ that satisfies the following three restrictions:

1) $\text{NIL} \in \mathcal{R}$;
2) if $a \in \mathcal{O}b$, then $(\text{UNIT } a) \in \mathcal{R}$;
3) if $a \in \mathcal{O}b$ and $A \in \mathcal{R}$, then 
   $$(\text{APPEND}(\text{UNIT } a) \ A) \in \mathcal{R}.$$ 

To verify the correctness of the given definition we wish to show that sequences are introduced very "naturally" in respect to concatenation, i.e.

$$\langle a_1, \ldots, a_n \rangle \cdot \langle a_{n+1} \rangle = \langle a_1, \ldots, a_n, a_{n+1} \rangle.$$ 

The proof of this fact is clear enough and may be obtained by induction:

$$\langle a_1, \ldots, a_n \rangle$$
$$\equiv (\text{DA}_{a_1} \ldots \cdot \text{DA}_{a_n} \cdot \text{DA}_{a_{n+1}})$$
$$= \mathcal{R}(\lambda t'. [a_1, \ldots [a_n, t']] \ldots)],(\text{DA}_{a_{n+1}})$$
$$= \lambda t' \cdot ([\lambda t'. [a_1, \ldots [a_n, t']] \ldots]) [a_{n+1}, t]$$
$$= \lambda t' \cdot [a_1, \ldots [a_n, [a_{n+1}, t]] \ldots]].$$

**Proposition.** The finite sequences set $\mathcal{R}$ is given by the conditions listed below:

1) if $A \in \mathcal{R}$, then there exists a natural number $n \geq 0$ such that $\text{CDR}^* \ A = \text{NIL}$;
2) if $A \in \mathcal{R}$, then the object (sequence) $A$ is represented by $A = \text{NULL } A$ $(\mathcal{X} \text{ NIL})$

$$(\text{APPEND}(\text{UNIT}(\text{CAR } A))(\text{CDR } A)).$$

The first condition is called the finiteness and the second one is called the **fixed point equation**.

**Proof.** 1) By induction on the value of $n$.

1.1) $\text{CDR}^* \text{NIL} = \text{NIL}$.

1.2) $\text{CDR}(\text{UNIT } a) = (\lambda xy : xy0)(\text{DA}_{a}) = \lambda y : y \neq \emptyset \neq \text{NIL}$.

1.3) Let $\exists n \geq 0$ such that $\text{CDR}^* \ A = \text{NIL}$.

The sequence of the length $n + 1$ is derivable from $A$ by $\text{APPEND}(\text{UNIT } a) \ A = \langle a, A \rangle$. Furthermore, $\text{CDR}^{*+} \langle a, A \rangle \equiv \text{CDR}^* (\text{CDR} \langle a, A \rangle) = \text{CDR}^* \ A = \text{NIL}$.

2) Analyzing repeatedly the three cases according to the construction rules of the set $\mathcal{R}$, we obtain the following.

2.1) $A \Rightarrow \text{NIL} \neq \langle \rangle$.

$$\text{NULL} \langle \rangle (\mathcal{X} \text{ NIL}) (\text{APPEND}(\text{UNIT}(\text{CAR } \langle \rangle))(\text{CDR } \langle \rangle))$$
$$= 1 (\mathcal{X} \text{ NIL})(\ldots) = \text{NIL}.$$

2.2) $A \Rightarrow \text{UNIT } a$.

a) $\text{CDR}(\text{UNIT } a) = \langle \rangle$.

b) $\text{CAR}(\text{UNIT } a) = a$.

c) $\text{APPEND}(\text{UNIT } a) \langle \rangle = \text{UNIT } a$.

Then $\text{NULL}(\text{UNIT } a)(\mathcal{X} \langle \rangle)(\text{UNIT } a) = \mathcal{E}_{\text{nil}} \emptyset (\mathcal{X} \emptyset) \langle a \rangle (\mathcal{X} \langle \rangle)(\langle a \rangle)$
2.3) $A \iff \text{APPEND}(\text{UNIT } a) B$.
   a) $\text{CDR } A = B$.
   b) $\text{CAR } A = a$.

Then $\text{NULL } A (\mathcal{H}(\ ))(\text{APPEND}(a) B) = 0(\mathcal{H}(\ )) A = A$.

The case analysis completes the proof of this proposition. Note that the following sequence length definition can be introduced (for the sequence $A \in \mathfrak{S}$): it is the minimal number $m$ such that $\text{CDR}^m A = \text{NIL}$. Now the application of the finite sequences theory (computational model, CM) will be shown for situational analysis.

8. CM-APPLICATION FOR SITUATIONAL ANALYSIS

Let the situation model (Fig. 3) be a space containing the following components:
1) one moving point (objective, $o$);
2) the initial point of the movement (ablative, $a$);
3) the terminal point of the movement (allative, $l$);
4) the intermediate point in the movement trace (prolative, $p$).

![Fig. 3. Basic model of the convertible situation.](image)

The points of the space are assumed to be the cases of situation participants. The components of this space, i.e. relations linking the situation cases, are called the relators and denoted by $R$, $r$ with possible indices. The intensional theory has thus arisen involving variables for the sets and classes and the following set theoretical axioms: equality ((($X \subseteq Y$) & ($Y \subseteq X$)) = ($X = Y$)), where $X, Y$ are variables for the classes; the axioms of pair, empty set, existence of the classes, union, the power set, separation, replacement and infinity; we shall call it $\mathfrak{R}^*$-theory. The $\mathfrak{R}^*$-theory constructs, needed for our purposes, will be labelled by the asterisk and will be assigned to some $\lambda$-calculus constructions (Table 2).

The $\mathfrak{R}^*$-theory objects are given by $|t|^{\ast}, t \ast [e, R_{ab} \ast]$, where $\alpha, \beta \in \{o, a, l, p\}$ $\alpha \neq \beta$ ("\neq" is used for "graphically discriminates");
Table 2.
Assignment of the $\lambda$-calculus objects to $\mathcal{H}^*$-theory constructs.

<table>
<thead>
<tr>
<th>$\mathcal{H}^*$-theory</th>
<th>Ordered n-tuple</th>
<th>$[a_1^<em>, ..., a_n^</em>]$</th>
<th>$[a_1^<em>, ..., a_n^</em>[\varepsilon b]^*]$</th>
<th>$a^* \varepsilon b^*$</th>
<th>$a^* \subseteq b^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$-calculus</td>
<td>$[a_1, ..., a_n]$</td>
<td>$[a_1, ..., a_n[\varepsilon b]$</td>
<td>$[a, b[$ or $(\lambda r \cdot ra)b]$</td>
<td>$\Xi_{\omega}$ (\Xi-system construction cf. [9])</td>
<td></td>
</tr>
</tbody>
</table>

$[t_1^*, t_2^*, t_3^*[\varepsilon R_{\omega}\theta^*, \text{ where } \alpha, \beta, \gamma \in \{o, a, i, p\}, \alpha \neq \beta, \alpha \neq \gamma, \beta \neq \gamma; \]

$[t_1^*, t_2^*, t_3^*[\varepsilon R_{\omega}\theta^*, \text{ where } \alpha, \beta, \gamma, \delta \text{ are pairwise distinct indices.} \]

These objects within the $\eta\Xi$-system of $\lambda$-conversion can be represented by

$[t_1, t_2] R_{\omega} = R_{\omega} t_1 t_2 \text{ etc.}$

We consider the objects like $[[T, T_o] R_{\omega}, T_o] R_{\omega}$. Note that they can be represented by the $\mathcal{H}$-theory objects (the $\mathcal{H}^*$-theory embedded in $\lambda$-calculus will be called $\mathcal{H}$-theory) and introduced by the axiom scheme

$$[[T, T_o] R_{\omega}, T_o] R_{\omega} \]

$$ \equiv [T, T_o, T_o](\mathbb{R}^2 R_{\omega} R_{\omega}) \]

$$ \equiv [T, T_o, T_o] R_{\omega} = [T, T_o, T_o](\mathbb{C}, R_{\omega}). \]

(1)

To comment the notions from Table 2 and below in more detail, it should be recalled that $[a_1, ..., a_n] \equiv \lambda r \cdot ra_1 ... a_n, \quad n \geq 0$. Then taking into account the case $n = 2$, we conclude that $[a_1, a_2] \equiv \lambda r \cdot ra_1 a_2 = \mathcal{D}a_1 a_2$ in accordance with the previous remarks and constructions. Consider furthermore the family

$\mathcal{C}_{[1]} \equiv \mathbb{C}; \]

$\mathcal{C}_{[n+1]} \equiv (\mathbb{C} \times \mathbb{C}) \times \mathbb{C}, \quad n \geq 0.$

It is obvious that $\mathcal{C}_{[1]} \not\subset \mathcal{D}$ gives the definition of the pair operator. On the other hand, the family of ordered $n$-tuples is defined by $\mathcal{C}_{[n]} \not\subset (\mathbb{R}^2 \mathbb{C}) \times \mathbb{C}, \quad n \geq 1$ (recall that an ordered $n$-tuple is denoted by $[a_1, ..., a_n] \equiv \lambda r \cdot ra_1 ... a_n$. Remark that the ordered $n$-tuple $[a_1, ..., a_n]$ is not the same as the $n$-sequence $(a_1, ..., a_n)$.

We now turn to axiom (1) analysis. Let $R_{\omega} \equiv to \ cause_{\omega}, \ R_{\omega} \equiv to \ be_{\omega}, \ R_{\omega} \equiv to \ have_{\omega}, \ $ where “to cause” is an abbreviation for “to be a motivation of...” (in a common sense). Then:

if $T_o \equiv bureau_{o}, \ T_o \equiv cargo_{o}, \ T_i \equiv city_{i}, \ $ then $[city_{i}, cargo_{i}] to be_{\omega} \equiv T_o; \]

$[T_o, bureau_{o}] to cause_{\omega}; \]

or $[(city_{i}, cargo_{i}) to be_{\omega}, bureau_{o}] to cause_{\omega}.$
This version of the axiom (1) is to be interpreted by the hypothetical sentence: "The cargo BUREAU causes the CARGO to be located in the CITY" (Fig. 4a,b; 5a,b).

Fig. 4. The frame examples.
   a) The object frame.
   b) The axiom example.

Fig. 5. Graphical representation of the semantic field.
   a) The sample frame and the example frame.
   b) The example of scenario frame construction.
Certainly, the stepwise reasoning leads to the assumption

\[ (\mathcal{B}^2 \text{ to cause}_{\text{on}} \text{ to be}_{\text{on}}) \models \text{ to give}_{\text{on}}, \]

i.e. the domain \( \text{Dom} \) of the variable \( R_{\text{on}} \) contains "give"-like predicates:

\[ \text{Dom}(R_{\text{on}}) \models \{ \text{to assign, to hand, to give, to present, to acquire, to get, to lend, to pay, to spare, ...} \}. \]

This reasoning also leads to the concept of the semantic field (SF). (N.B. Several definitions of the semantic field can be summed up in this paper as follows: the set of derivations from the axiom \( \mathcal{A} \) is called the semantic field (type). The SF-notation is introduced by \( \text{TYPE}_{\mathcal{A}}(x_1, \ldots, x_n) \), where \( x_1, \ldots, x_n \) are the distinct variables, \( n \geq 0 \).

The consideration of the axiom (1) gives the following example of the semantic field (SF1):

(SF1) (a1) The bureau carries the cargo to the city;
(b1) The cargo is being carried by the bureau to the city;
(c1) The bureau supplies the city with the cargo;
(d1) The city is being supplied by the bureau with the cargo;
(e1) The city receives the cargo by means of the bureau;
(f1) The cargo is being received by the city by means of the bureau.

Let us demonstrate the derivation of SF1 from the axiom (1).  

**Proposition.** In general, the semantic field SF1 has the type in accordance with Fig. 6, where \( \mathcal{B}^2 R_{\text{on}} = R_{\text{on}} \), and \( R_{\text{on}}, R_{\text{on}}, T_{1}, T_{0}, T_{a} \) are variables.

---

**Fig. 6.** The semantic field derivation from the axiom

\[ [[T_1, T_2]R_{\text{on}}, T_a]R_{\text{on}}. \]
Proof. It is sufficient to check up the following conversions:

\[ [T_1, T_2, T_3] R_{sw} = [T_1, T_2, T_3] (\mathcal{E}_1 R_{sw}) = [T_1, T_2, T_3] (\mathcal{E}_2 R_{sw}) = [T_1, T_2, T_3] (\mathcal{E}_1 R_{sw}) = [T_1, T_2, T_3] (\mathcal{E}_2 R_{sw}) \]

Now it is easy to conclude that according to Fig. 6

\[ TYPE_{[T_1, T_2]} (R_{sw}, R_{sw}, T_1, T_2, T_3) \equiv \{(a), (b), (c), (d), (e), (f)\} \]

Fig. 7. The semantic field for the example frames.
Corollary. $\mathcal{B}^2 R_{\text{os}} R_{\text{o}} \equiv \text{"to cause to have";}$
$R_{\text{os}} \equiv \text{"to give";}$  $C_i R_{\text{os}} \equiv \text{"to be given";}$
$C_i R_{\text{os}} \equiv \text{"to supply";}$
$C_{i_2} R_{\text{os}} \equiv \text{"to be supplied";}$
$C_{i_2}(C_{i_2} R_{\text{os}}) \equiv \text{"to receive";}$
$C_{i_2} R_{\text{os}} \equiv \text{"to be received".}$

N.B. In establishing the correspondence between the relator $R_{\text{os}}$ and the frames we say that
the relator $R_{\text{os}}$ generates six scenario (script) frames, i.e.

$SCENARIO_{R_{\text{os}}} \equiv \{R_{\text{os}}, C_2 R_{\text{os}}, C_i R_{\text{os}}, C_{i_2} R_{\text{os}}, C_{i_2}(C_{i_2} R_{\text{os}}), C_{i_2} R_{\text{os}}\}.$

The illustration is given in Fig. 7. Then the abstracting of the variables in the elements
distinguishes between the set $SCENARIO_{R_{\text{os}}}$ and the set $TYPE_{[[T_o, T_o]R_{\text{os}}, T_o]}$ (Fig. 8; compare with Fig. 6).

Consider the following axiom scheme:

$$[[[T_o, T_o]R_{\text{os}}, T_o]R_{\text{ol}}, T_o]R_{\text{os}} = [T_o, T_t, T_o](\mathcal{B}^2(\mathcal{B}^2 R_{\text{os}} R_{\text{ol}}) R_{\text{ol}}, T_o) = [T_o, T_t, T_o]P_{\text{old}}, \tag{2}$$

where $R_{\text{os}} \equiv \text{to cause}_{\text{os}};$  $R_{\text{ol}} \equiv \text{to have}_{\text{ol}};$  $R_{\text{os}} \equiv \text{to be}_{\text{os}};$  $T_t \equiv \text{knowledge}_t;$  $T_o \equiv \text{facts}_o;$
$T_t \equiv \text{inspector}_t;$  $T_o \equiv \text{eyewitness}_o.$

Example 1. The axiom scheme (2) under the appointed substitution allows the following
interpretation (the 1st line in Table 3):

![Diagram](image-url)

Fig. 8. The derivation of the axiom $[[[T_o, T_o]R_{\text{os}}, T_o]R_{\text{os}}$ scenarios for $R_{\text{os}}.$
[knowledge, facts] to be\text{\_n} \Leftrightarrow x; \\
[x, 1, \text{inspector},] to have\text{\_n} \Leftrightarrow y1; \\
[y, 1, \text{eyewitness},] to cause\text{\_n}.

It can be established that defining

\[ P_{\text{ola}} \mathcal{B}((\mathcal{B}^2 \text{ to cause}) \text{ to have}) \text{ to be, knowledge,} \]

implies that the "to warn"-like predicates are assigned to \( P_{\text{ola}} \):

\[ \text{Dom}(P_{\text{ola}}) \Leftrightarrow \{ \text{report, to depose, to cry, to declare, to write, to tell, to apprise, to warn, } \ldots \} \]

i.e. \( P_{\text{ola}} \Leftrightarrow "\text{to warn"}. \) The semantic field of \( P_{\text{ola}} \) is the following:

(a2) The eyewitness is apprising the inspector of the facts.

(b2) The facts are being apprised by the eyewitness to the inspector;

(c2) The eyewitness tells the inspector of the facts;

(d2) The inspector is being warned by the eyewitness of the facts;

(e2) The inspector is clearing up the facts from the eyewitness;

(f2) The facts are being cleared up by the inspector from the eyewitness.

**Proposition.** The semantic field (SF2) is, in general, the same as shown in Fig. 9.

![Fig. 9. The derivation of the axiom \([[T', T_n,]R_{\text{ola}}, T], R_{\text{ola}}, T], R_{\text{ola}} \) semantic field.](image)

**Proof.**

(i) Prove \((2) \rightarrow (a2)\). Indeed,

\[(2) \Leftrightarrow [[T', T_n,]R_{\text{ola}}, T], R_{\text{ola}}, T], R_{\text{ola}} \]

\[\text{conv} [T', T_n, T], P_{\text{ola}} \text{ conv} [T, T_n, T], (\forall, P_{\text{ola}}) \Leftrightarrow (a2).\]

where \( P_{\text{ola}} \Leftrightarrow \mathcal{B}((\mathcal{B}^2 R_{\text{ola}}) R_{\text{ola}} T). \)
(ii) Prove \((2) \rightarrow (b2)\). Indeed,
\[
\begin{align*}
(2) \quad & \text{conv} (a2) \text{ conv } [T, t, T'] P_{ole} \\
& \text{conv } [T, t, T'] (\langle \varepsilon_{[2]} P_{ole} \rangle) \equiv (b2).
\end{align*}
\]

(iii) Prove \((2) \rightarrow (c2)\).
\[
(2) \quad \text{conv} (a2) \text{ conv } [T, t, T'] P_{ole} \equiv (c2).
\]

(iv) Prove \((2) \rightarrow (d2)\).
\[
(2) \quad \text{conv} (c2) \text{ conv } [T, t, T'] (\langle \varepsilon_{[2]} P_{ole} \rangle) \equiv (d2).
\]

(v) Prove \((2) \rightarrow (e2)\).
\[
(2) \quad \text{conv} (d2) \text{ conv } [T, t, T'] (\langle \varepsilon_{[2]} P_{ole} \rangle) \equiv (e2).
\]

(vi) Prove \((2) \rightarrow (f2)\).
\[
(2) \quad \text{conv} (b2) \text{ conv } [T, t, T'] (\langle \varepsilon_{[2]} P_{ole} \rangle) \equiv (f2).
\]

Now \[\text{TYPE}_{[T, t, T']} \text{Pole}(R_{oa}, R_{ot}, R_{ot}, T, t, T, T, T) \Rightarrow ((a2), (b2), (c2), (d2), (e2), (f2))\],
because all \([T, t, T']\) transpositions have been produced.

**Corollary.**
\[
\begin{align*}
\varepsilon_{[2]} P_{ole} & \Leftarrow \text{to apprise}; \langle \varepsilon_{[2]} P_{ole} \rangle \Leftarrow \text{to be apprised}; \\
P_{ole} & \Leftarrow \text{to tell}; \langle P_{ole} \rangle \Leftarrow \text{to be warned}; \\
\varepsilon_{[1]} (\langle \varepsilon_{[2]} P_{ole} \rangle) & \Leftarrow \text{to clear up}; \\
\varepsilon_{[1]} (\langle \varepsilon_{[2]} P_{ole} \rangle) & \Leftarrow \text{to be cleared up}.
\end{align*}
\]

**N.B.** The six scenario frames are given by the predicate \(P_{ole}\):
\[\text{SCENARIO}_{Pole} \Leftarrow \{ \varepsilon_{[1]} P_{ole}, \varepsilon_{[2]} P_{ole}, P_{ole}, \varepsilon_{[1]} (\langle \varepsilon_{[2]} P_{ole} \rangle), \varepsilon_{[1]} (\langle \varepsilon_{[2]} P_{ole} \rangle) \}.\]

9. STUDY OF THE CONVERSION OF SITUATIONS

Let \(R_{oa}\) and \(s\) be
\[R_{oa}: (\text{Sent}_r, (\text{Tm}, \text{Sent})), s: \text{Sent},\]  
(3)
i.e. the first argument is the embedded sentence (here: \(s \Leftarrow R_{oa}, T, t\), \(\text{Sent}_r \Leftarrow (\text{Sent}, \text{Tm})\)).

The \(R_{oa}\) can be treated as the relation between the source of the situation and the very situation.

**Interpretation.** We assume that
\[\text{Dom}(R_{oa}) \Leftarrow \{ \text{to cause, to force} \}.\]

**Example 2.** Consider the situation: "I force the car to be stopped". (Я заставляю, чтобы автомобиль остановился.) (See line 2 in Table 3): [\(s, t\) \(R_{oa}\)].

Let \(R_{oa}: (\text{Tm}, (\text{Tm}, \text{Sent})), \)
\[\text{Dom}(R_{oa}) \Leftarrow \{ \text{to be, to be located} \}.
\]

Then
\[\text{[s, t]R_{oa} \Leftarrow [(T, t)R_{oa}, T, t]R_{oa} = \text{I force the car to be stopped}.}\]
Generalization. The sentence "I force the car to be stopped" can be considered more generally:

\[[t_1, t_2]R_{\omega}, [t_1, t_2]R_{\omega}]R_{\omega},

i.e. we can assume that \( t \equiv s_r \).

**Example 3.** Consider the situation: "I emit the actions that cause the car to be stopped". (От меня исходят действия, вследствие чего автомобиль находится в (состоянии) останове (a)) (1.3 in Table 3).

It is clear now that a more general situation is considered (cf. 1.2 in Table 3).

Let

\[ R_{\omega}: (Tm, (Sent_r, Sent)) \quad \text{and} \quad s \equiv [t_1, s_r]R_{\omega}, \tag{4} \]

i.e. the subject (topic) in (3) converted into object in (4). \( R_{\omega} \) can be assumed as follows: the situation that has arisen is the topic of the sentence.

**Interpretation.** We assume that

\[ \text{Dom}(R_{\omega}) \equiv \{ \{ \text{the causal connectives} \} \}, \]

\[ \text{Dom}(t) \equiv \{ \{ \text{the animated nouns} \} \}. \]

**Example 4.** Consider the situation: "I cause that the car is in movement". ((To, что) автомобиль в движении, каузируется мною) (1.4 in Table 3).

Let

\[ R_{po}: (Tm, (Tm, Sent)), \quad R_{op}: (Tm, (Tm, Sent)), \]

\[ s_1 \equiv R_{po}t_1, t_2, \quad s_2 \equiv R_{op}t_1, t_2. \]

The \( R_{po} \) is supposed as follows: \( t_1 \) denotes the instrument mediating the action or the point being interposed by the movement; \( t_2 \) denotes the object mediating the instrument \( p \) or the moving object that interposes the point \( p \).

**Example 5.** Consider the situation: "That the human uses the car causes the speed to be bounded". ((To, что) человек пользуется автомобилем каузирует ограничение над скоростью) (1.5 in Table 3).

Let

\[ R_{po}: (Sent_r, (Sent_r, Sent)), \]

\[ s \equiv R_{po}s_1s_1s_2 \equiv [s_1, s_1, s_2]R_{po}, \]

i.e. both arguments are the embedded sentences.

In the situation \( R_{po} \) the \( s_r \) is the means of obtaining the situation \( s_r \), i.e. \( s_r \) and \( s_r \) are connected by the circumstantial-instrumental relations.

**Interpretation.** The predicate \( R_{po} \) allows the following consideration:

\[ \text{Dom}(R_{po}) \equiv \{ \text{by means of, because of} \}. \]

**Example 6.** The situation: "The car was accurate by means of repair". (Автомобиль был исправлен посредством ремонта (того, что проведен ремонт)) is assigned to the scheme in the line 6 (Table 3).
The reversed scheme

\[ R_{op} : (\text{Sent}_r, (\text{Sent}_r, \text{Sent})) \]

differs from the previous one because of the reversed consideration of the topic and the object.

**Example 7.** "It is by means of repair that the car is accurate". (То, что автомобиль был в ремонте послужило средством достижения того, что автомобиль исправен.)

The schemes

\[ R_{pr} : (\text{Tm}, (\text{Sent}, \text{Sent})) \]
\[ R_{rp} : (\text{Sent}, (\text{Tm}, \text{Sent})) \]

can be assumed as the particular cases of the previous ones. Now let

\[ R_{rs} : (\text{Tm}, (\text{Tm}, \text{Sent})), \quad s \models R_{rs} t_1 t_2, \]

i.e. we consider the sentence where the arguments are related by the objective-locative relations (the argument \( t_2 \) "is located into" the argument \( t_1 \)).

**Interpretation.**

\[ \text{Dom}(R_{rs}) \models \{ \text{to be, to be located} \}, \]
\[ \text{Dom}(t_1) \models \{ \langle \text{the nouns (concrete and abstract, animated and inanimated)} \rangle \}, \]
\[ \text{Dom}(t_2) \models \{ \langle \text{the nouns (concrete, animated and inanimated)} \rangle \}. \]

**Example 8.** Consider the situations:

"The car is located in the crossroad" and "John belongs to the class of drivers" (i.e. "John is a driver"). (Автомобиль на перекрестке. Иван – в классе шоферов (Иван-шофер)).

These sentences are assigned to the schemes in 1.8a,b; Table 3.

The reversed scheme

\[ R_{sr} : (\text{Tm}, (\text{Tm}, \text{Sent})) \]

is interpreted by

\[ \text{Dom}(R_{sr}) \models \{ \text{to have, to contain} \}. \]

**Example 9.** "The car contains the passengers". (Автомобиль содержит пассажиров.)

Consider the scheme

\[ R_{os} : (\text{Sent}_r, (\text{Sent}_r, \text{Sent})) \]

and its reversion

\[ R_{rs} : (\text{Sent}_r, (\text{Sent}_r, \text{Sent})). \]

In particular the following schemes can be considered:

\[ s_1 \vdash [(t_1, t_2) R_{os}, [t_1, t_2] R_{os}] R_{os}, \]
\[ s_2 \vdash [(t_1, t_2) R_{os}, [s_1, t_2] R_{os}] R_{os}, \]
\[ s_3 \vdash [(t_1, t_2) R_{os}, [t_1, t_2] R_{os}] R_{os} \text{ etc.} \]

**Interpretation.**

\[ \text{Dom}(R_{os}) \models \{ \text{where, when, if} \}. \]
Example 10. Consider the situation:

"If we have the combustibles, then we will drive (localize in driving) the car". (Если у нас будет горючее, мы сможем управлять автомобилем) (1.10; Table 3).

Let us consider the particular case of the previous schemes in more detail:

$$R_w: (Tm, (Sent, Sent)).$$

$$s \models [t_1, s_r, 1]R_w,$$

where

$$Dom(t_1) := \{\text{possibly, necessarily, probably, ...}\},$$

i.e. we assume that $s_r, 1$ is localized in some modality.

Example 11. “Perhaps, the car is off”, “It is localized in me that the car emits the noise”. (Возможно, автомобиль уехал. То, что шум идет от автомобиля, воспринимается мною) (1.11a,b; Table 3).

The consideration of the reversed schemes is clear.

10. VOICES

Now we consider the constructions of the active and passive voices. They are assigned to

$$R_{am}: (Sent_r, (Tm, Sent))$$ (the active voice)

$$R_{om}: (Tm, (Sent_r, Sent))$$ (the passive voice),

where

$$(t_3 : Sent_r) \in \{[t_1, t_2]R_{om}, [t_1, t_2]R_w\}.$$

Examples 12—13. “I cause that the car is captured by the movement”. (Я вызываю то, что движение распространяется на автомобиль):

$$(\lambda (r_2 : \text{rel}(o, a)), s_r, 1, t_1) \cdot r_2, s_r, 1, t_1,$$

$$r_2((\lambda (r_1 : \text{rel}(o, l)), t_1, t_2) \cdot r_1, t_1, t_2, r_1, t_2) t_3,$$

where

$$(I : Tm) \models t_3,$$

(to cause: $\text{rel}(o, a)) \models r_2,$$

(movement: $Tm) \models t_1,$

(to be captured: $\text{rel}(o, l)) \models r_1,$

(car: $Tm) \models t_2.$

In this context

$$s_r \models [t_1, t_2]R_{am}, t_3]R_{om},$$

that is why the more precise treatment is the notation

$$[(\lambda s : \text{that s})([t_1, t_2]R_{om}), t_3]R_{om},$$

instead of

$$[[t_1, t_2]R_{om}, t_3]R_{om}.$$
Frame theory and computations

действие распространяется на автомобиль, исходит от меня (каузируется мною)):

\[ [t, (\lambda s \cdot \text{that } s)([t, t_s]R_{sw})]R_{sw}. \]

**Example 15.** “It is caused by me that the car captures the movement”. (То, что автомобиль в движении, кazuируется мною (исходит от меня)):

\[ [t, (\lambda s \cdot \text{that } s)([t, t_s]R_{sw})]R_{sw}. \]

11. RECEPTION AND SENSATION (ABERRATION)

Consider the four axiom schemes as follows:

\[ R_{sw}: (Sent_{tr}, (Tm, Sent)) \] (active voice),
\[ R_{so}: (Tm, (Sent_{tr}, Sent)) \] (passive voice),

where

\[ (t_s: Sent_{tr}) \in \{[t, t_s]R_{sw}, [t, t_s]R_{sw}\}. \]

**Example 16.** “I localize that the car emits the noise” (cf.: “I listen to the noise from the car”). (Я локализую в себе то, что автомобиль издает шум. (Я слышу шум, исходящий от автомобиля)):

\[ \lambda (s_r, t) \cdot ([s_r, t](\text{localize})) \]
\[ ((\lambda s \cdot \text{that } s) \]
\[ ((\lambda (t, t_s) \cdot ([t, t_s](\text{emits} \ the \ noise \ car))) \] I.

**Example 17.** “I localize that the noise is emitted by the car”. (Я локализую в себе то, что шум исходит от автомобиля):

\[ \lambda (s_r, t) \cdot ([s_r, t] \text{localize}) \]
\[ ((\lambda s \cdot \text{that } s) \]
\[ ((\lambda (t, t_s) \cdot ([t, t_s](\text{is} \ emitted))) \]
\[ (by \ the \ car \ noise)) \] I.

**Example 18.** “It is localized in me that the car causes the noise”. (То, что автомобиль каузирует шум, локализуется во мне):

\[ ((\lambda (t, s_r) \cdot ([t, s_r](\text{is} \ localized)) \ (\text{in} \ me) \]
\[ ((\lambda s \cdot \text{that } s) \]
\[ ((\lambda (t, t_s) \cdot ([t, t_s](\text{causes})) \]
\[ (the \ noise) \ (the \ car))). \]

**Example 19.** “It is localized in me that the noise is emitted by the car”. (То, что шум исходит от автомобиля, локализую Я (в себе) (локализуется во мне)):

\[ \lambda (t, s_r) \cdot ([t, s_r](\text{is} \ localized)) \ (\text{in} \ me) \]
\[ ((\lambda s \cdot \text{that } s) \]
\[ ((\lambda (t, t_s) \cdot ([t, t_s](\text{is} \ emitted)) \]
\[ (by \ the \ car) \ (the \ noise))). \]
Table 3.
The situations of \textit{N}-theory ("Knowledge base").

<table>
<thead>
<tr>
<th>Example number</th>
<th>Situation description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[y_{1_1}, \text{eyewitness}, \text{(to cause)}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>2</td>
<td>[x_{2_1}, \text{force}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>3</td>
<td>[x_{2_1}, z_{4_1}, \text{(to cause)}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>4</td>
<td>[x_{3_1}, \text{force}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>5</td>
<td>[x_{4_1}, v_{1_1}, \text{(to cause)}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>6</td>
<td>[x_{6_1}, x_{5_1}, \text{(by means of)}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>8a</td>
<td>[\text{crossroad, car}<em>{., \text{(to be located)}</em>{o_{e}}}]</td>
</tr>
<tr>
<td>8b</td>
<td>[\text{class of the drivers, John}<em>{., \text{(to belong)}</em>{o_{e}}}]</td>
</tr>
<tr>
<td>10</td>
<td>[y_{2_1}, x_{10_1}, \text{if}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>11a</td>
<td>[\text{perhaps, x}<em>{13_1}, \text{(to be localized)}</em>{o_{e}}]</td>
</tr>
<tr>
<td>11b</td>
<td>[(\text{in me}), z_{9_1}, \text{(to be localized)}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>12</td>
<td>[y_{3_1}, \text{force}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>13</td>
<td>[x_{3_1}, \text{force}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>14</td>
<td>[I_{e}, y_{3_1}, \text{(to be caused)}<em>{o</em>{e}}]</td>
</tr>
<tr>
<td>15</td>
<td>[I_{e}, x_{3_1}, \text{(to be caused)}<em>{o</em>{e}}]</td>
</tr>
</tbody>
</table>
We conclude the paper by summarizing the situation in Table 3 which integrates the considered examples. Their generalization is given in Table 4 that establishes the complexity degree of the ℜ-theory objects assigned to the involved situations. Finally, Fig. 10 represents the ℜ*-theory relations assigned to the ℜ-theory situations.

12. CONCLUSIONS

The main goal of this paper has been to develop a computational framework capable of dealing with the knowledge representation problem. An attempt has been done to formalize the frames. This frame theory offers some advantages: a small set of primitive notions (application, abstraction, etc.), the purely applicative model of situations, the searching algorithm along the relator hierarchy, a small set of generic roles, an explicit correspondence of the scripts with an initial axiom, etc.

One of the principal topics for future research is the development of mathematical tools for solving the problems in the field of data base management and especially in the field of knowledge representation.

1. The priority topic is the incorporation of every user’s knowledge in the frame theory. Some preliminary results have been obtained in intensional logic.

2. Another topic is the development of computational environment for the purposes of artificial intelligence.

Some lambda-calculus results are promising. But the incorporation of every user’s knowledge within the unified framework leads to inconsistencies. On the other hand, the research in the field of lambda-calculus and pure combinatory logic dates back to
Table 4.
Objects of the N\textsuperscript{1}-theory.

<table>
<thead>
<tr>
<th>Example number</th>
<th>Object description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>$[[T, T_R^u, T, T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>2.</td>
<td>$[[T, T_R^u, T, T_R^u]_{\omega}$</td>
</tr>
<tr>
<td>3.</td>
<td>$[[T, T_R^u, [T', T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>4.</td>
<td>$[[T, T_R^u, T, T_R^u]_{\omega}$$</td>
</tr>
<tr>
<td>5.</td>
<td>$[[T', T, [T, T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>6.</td>
<td>$[[T, T_R^u, [T', T, T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>8a, 8b</td>
<td>$[T, T, R_{10}]_{\omega}$</td>
</tr>
<tr>
<td>10.</td>
<td>$[[T, T, T_{10}, [T', T, T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>11a.</td>
<td>$[T, [T, T', T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>11b.</td>
<td>$[T, [T, T', T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>12.</td>
<td>$[[T, T_R^u, T, T_R^u]_{\omega}$</td>
</tr>
<tr>
<td>13.</td>
<td>$[[T, T_R^u, T, T_R^u]_{\omega}$</td>
</tr>
<tr>
<td>14.</td>
<td>$[T, [T, T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>15.</td>
<td>$[T, [T, T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>16.</td>
<td>$[[T, T_R^u, T, T_R^u]_{\omega}$</td>
</tr>
<tr>
<td>17.</td>
<td>$[[T, T_R^u, T, T_R^u]_{\omega}$</td>
</tr>
<tr>
<td>18.</td>
<td>$[T, [T, T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
<tr>
<td>19.</td>
<td>$[T, [T, T_R^u]<em>{\omega}]</em>{\omega}$</td>
</tr>
</tbody>
</table>

"Hilbert—Curry program" and leads to paradoxes. In the 1960th and early 1970th this was the main difficulty in computer applications of these mathematical tools. It may be accepted that some fruitful attempts devoted to that end are those of A. S. KUZICHEV on the inconsistency omission method. This method follows Gentzen’s, Lorentzen’s and A. A. Markov’s constructions.

To date three types of inconsistency have been distinguished: absolute, internal and arithmetical ones. Perhaps, the integration of different "views" of the problem domain (and data base) is related to the analysis of the kinds of inconsistency.

Acknowledgement. I wish to express my appreciation of the attention that G. S. POSPELOV, L. T. KUZIN, A. S. KUZICHEV and D. A. POSPELOV have payed to this research.
<table>
<thead>
<tr>
<th>$R_{lw}$</th>
<th>$l$</th>
<th>$o$</th>
</tr>
</thead>
<tbody>
<tr>
<td>to be</td>
<td>knowledge</td>
<td>facts</td>
</tr>
<tr>
<td>to be</td>
<td>stopped</td>
<td>car</td>
</tr>
<tr>
<td>to be</td>
<td>movement</td>
<td>car</td>
</tr>
<tr>
<td>to be</td>
<td>bounded</td>
<td>speed</td>
</tr>
<tr>
<td>to be</td>
<td>accurate</td>
<td>car</td>
</tr>
<tr>
<td>to be</td>
<td>remount</td>
<td>car</td>
</tr>
<tr>
<td>to be located</td>
<td>crossroad</td>
<td>car</td>
</tr>
<tr>
<td>to belong</td>
<td>class of drivers</td>
<td>John</td>
</tr>
<tr>
<td>if</td>
<td>$y2$</td>
<td>$x10$</td>
</tr>
<tr>
<td>will drive</td>
<td>car</td>
<td>we</td>
</tr>
<tr>
<td>to be localized</td>
<td>in me</td>
<td>$z9$</td>
</tr>
<tr>
<td>to be localized</td>
<td>perhaps</td>
<td>$x13$</td>
</tr>
<tr>
<td>to be</td>
<td>set off</td>
<td>car</td>
</tr>
<tr>
<td>to be localized</td>
<td>in me</td>
<td>$z10$</td>
</tr>
<tr>
<td>to be localized</td>
<td>in me</td>
<td>$w4$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$R_{ow}$</th>
<th>$o$</th>
<th>$a$</th>
</tr>
</thead>
<tbody>
<tr>
<td>to cause</td>
<td>$y1$</td>
<td>eyewitness</td>
</tr>
<tr>
<td>to force</td>
<td>$x2$</td>
<td>1</td>
</tr>
<tr>
<td>to cause</td>
<td>$x2$</td>
<td>$z4$</td>
</tr>
<tr>
<td>to emit</td>
<td>actions</td>
<td>1</td>
</tr>
<tr>
<td>to cause</td>
<td>$x3$</td>
<td>1</td>
</tr>
<tr>
<td>to cause</td>
<td>$x4$</td>
<td>$v1$</td>
</tr>
<tr>
<td>to cause</td>
<td>$y3$</td>
<td>1</td>
</tr>
<tr>
<td>to emit</td>
<td>noise</td>
<td>car</td>
</tr>
<tr>
<td>to cause</td>
<td>noise</td>
<td>car</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$R_{ol}$</th>
<th>$o$</th>
<th>$l$</th>
</tr>
</thead>
<tbody>
<tr>
<td>to have</td>
<td>$x1$</td>
<td>inspector</td>
</tr>
<tr>
<td>to have</td>
<td>combustibles</td>
<td>we</td>
</tr>
<tr>
<td>to be captured</td>
<td>movement</td>
<td>car</td>
</tr>
<tr>
<td>to localize</td>
<td>$z9$</td>
<td>1</td>
</tr>
<tr>
<td>to localize</td>
<td>$w4$</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$R_{op}$</th>
<th>$p$</th>
<th>$o$</th>
</tr>
</thead>
<tbody>
<tr>
<td>to use</td>
<td>car</td>
<td>human</td>
</tr>
<tr>
<td>by means of</td>
<td>$x6$</td>
<td>$x5$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$R_{oa}$</th>
<th>$a$</th>
<th>$o$</th>
</tr>
</thead>
<tbody>
<tr>
<td>to be caused</td>
<td>1</td>
<td>$y3$</td>
</tr>
<tr>
<td>to be caused</td>
<td>1</td>
<td>$x3$</td>
</tr>
<tr>
<td>to be emitted</td>
<td>car</td>
<td>noise</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P_{oa}$</th>
<th>$o$</th>
<th>$l$</th>
<th>$a$</th>
</tr>
</thead>
<tbody>
<tr>
<td>to apprise</td>
<td>facts</td>
<td>inspector</td>
<td>eyewitness</td>
</tr>
</tbody>
</table>

Fig. 10. The situational relations of the $\mathcal{A}$-theory ("Data base").
REFERENCES


Received June 30, 1982