Computer Science Journal of Moldova, vol.**24**, no.1(70), 2016

Universics: a Theory of Universes of Discourse for Metamathematics and Foundations

Ioachim Drugus

Abstract

A new type of structures called "universes" is introduced to subsume the "von Neumann universe", "Grothendieck universes" and "universes of discourse" of various theories. Theories are also treated as universes, "universes of ideas", where "idea" is a common term for assertions and terms. A dualism between induction and deduction and their treatment on a common basis is provided. The described approach referenced as "universics" is expected to be useful for metamathematical analysis and to serve as a foundation for mathematics. As a motivation for this research served the Harvey Friedman's desideratum to develop a foundational theory based on "induction construction", possibly comprising set theory. This desideratum emerged due to "foundational incompleteness" of set theory. The main results of this paper are an explication of the notion "foundational completeness", and a generalization of well-founded-ness.

Keywords: induction, deduction, reduction, universe

1 Introduction

"Universics" is the term used by philosophers for the approach, which presupposes the treatment of any object from the perspective of the whole Universe, as if "the object was made in the image of the Universe". This is a limit case of the "holistic approach", when the "whole" is chosen to be the "maximal whole", i.e. "the Universe". The term "universics" was coined in the form of plural of modifier "universic" (similar to "mathematics" which comes from modifier "mathematic",

"related to knowledge") – a term customarily used for a feature of an object, whereby the Universe "reflects" in the object.

The "reflection principle" in set theory states that in any universe of discourse of a full-fledged set theory there are sets which "reflect" the features of the universe; examples of sets which are in the image of the von Neumann universe are Grothendieck universes. Thus, universics is not new to set theory domain. The goal of this paper is to give to universics the features of a mathematical discipline with its specific methodology, in particular, manifesting as (a) treating both theories and their universes of discourse on equal footing, as "universes", and (b) treating induction and deduction as dual notions.

A term which sounds to be closely correlated with "universics" is "multiverse", used to refer to the multitude of set theories viewed from the perspective of their universes of discourse – a multitude, which emerges due to various methods of constructing universes, among which the best known is "forcing method" [1, 2]. The difference between two approaches is that universics is a theory of separate structures called "universes", whereas in the approach using the term "multiverse", all such structures are treated as parts of a "multiverse".

The Grothendieck universes were introduced as a foundation for category theory – a foundation needed due to the fact, that the main concepts of this theory cannot be expressed in terms of sets, as this is explained next. Really, the "category of categories" is one of the key concepts of category theory, since "functor", "adjointness", "natural equivalence" and other central notions, which determine the value of this theory, are defined proceeding from the supposition that "category of categories" exists. But the existence of such an object presupposes the existence of the "class of classes" – a concept which is "contradictory by definition", since no proper class can be member of another class. Moreover, category theory uses such notions as "functor of functors", which imply existence of "proper classes of proper classes" – objects inadmissible in any set theory or class theory. Thus, since there are mathematical concepts, which cannot be expressed or represented in ZF set theory or NBG (von Neumann-Bernays-Godel) class theory, these theories cannot be said to be "foundations for mathematics".

On the other hand, surprisingly or not, only one of the Grothendieck universes turns out to be sufficiently rich to serve as a "foundation for mathematics" – true, to the same extent as set theory is, that is, as an "incomplete foundation". Merely this fact calls to have clarified the notion "foundational completeness" used by Harvey Friedman in a message to the automated email list "Foundations of Mathematics" (https://cs.nyu.edu/pipermail/fom/1997-November/000143.html). As he wrote, set theory "does not come close to doing everything one might demand of a foundation for mathematics" and it cannot be said to be "foundationally complete" for mathematics. Despite that this is an intuitively clear notion, Friedman mentions that he does not know how to define it. In universics this term will obtain a natural explication and this is one of the main results of this paper.

In addition to the strong conceptual arguments mentioned above to support the statement that set theory is foundationally incomplete for mathematics, there are reasons supporting the thesis that set theory is also foundationally incomplete for informatics (here, the term "informatics" is used to reference mainly the "data structures" used in computer science). Namely, set theory is too poor for representing the data structures used in informatics. On the other hand, *no* arguments were found in favor of the thesis that category theory, founded on Grothendieck universes, or just these universes alone may be foundationally incomplete. Therefore, a theory of universes, among which are also the Grothendieck universes, can be expected to be a foundationally complete theory. Based on this thesis, the axiomatic system of universics introduced in this paper is expected to be of good service to both mathematicians and computer scientists.

As an informal theory, universics was developed in [3, 4], where "universes" were treated as the largest structures, similarly to how the proper classes called "universes" are treated in set theory. There are, though, two major differences between set theory and universics, namely: (a) set theory studies conceptions which are obtained by abstraction from any kind of order, but the "universes", about which universics discusses, are *structures*; universics is a "structuralist" theory, (b) set theory studies "small scale" universes (see section 2.), but

universics studies any structures, even though it could be said to be focused on "large scale" universes. The first attempt to present universics as a framework of axiomatic theories was undertaken in [5].

In papers [3] and [4], any structure was considered built by repetitive application of three operations, called "aggregation", "association", "atomification" – operations for building sets, ordered pairs and atoms, respectively. The reason for the choice of these notions as a starting point in building a foundational theory is the belief that the notions "set", "ordered pair" and "atom" are sufficient to serve as a "conceptual orthogonal basis" for a universe of concepts. This conceptuality is intended to describe the "fabric" of a universe, and could be called "small scale universics". The current paper presents a theory of structures called "universes", which can be called "large scale universics". Since the "fabric" of a universe is irrelevant here, any knowledge of [3, 4] is not required for understanding the current paper.

2 On the terminology and conceptuality used in metamathematics

The terms used in a meta-discourse necessarily contain an amount of ambiguity, but one of the goals of universics is to serve as a language of metamathematics. Therefore, this section is intended to sort out some of the terms used in metamathematics and contribute to their precise use. Several other terms used in metamathematics will get a precise treatment later, when these terms will be explicated. In order that a term needing clarification can be easily found while reading the main text, the terms explained in this section are italicized.

A *universe of discourse* is correlated with a theory, which is said to "discuss about" the entities populating this universe, but to "describe" the universe (as a whole). The distinction between to "discuss about" and to "describe", is that one *discusses* necessarily about objects *within* a "universe of discourse", but one can *describe* something, which is not in a "universe of discourse" – say, one can describe the (whole) "universe of discourse", informally or by (the axioms of) a theory.

There are clearly specified universes (like Grothendieck universes), for which no theory describing them have been presented; about these they discuss informally. Also, the von Neumann universe was invented as a view upon the totality of sets and only later it was found that ZF theory extended with terms for ranks of the sets can describe it. Finally, two theories may have the same universe of discourse. These arguments support the treatment of the notion "universe" as a notion on its own, prior for it to be correlated with a theory with this universe as its "universe of discourse". Thus, in this paper the term "universe" will be used without necessarily being followed by the phrase "of discourse".

The term "universe of discourse" of a theory was introduced by George Boole (the inventor of the "algebra of logic") and this term turned out to be very useful in the early days of metamathematics, but later, this term created difficulties, for example, when it was used for a theory treated (itself) as a universe. Currently, the term "domain of discourse" is preferred (say, a search in Wiki of the phrase "universe of discourse" will result in an article titled "Domain of discourse"), even though it did not replace the term "universe of discourse", which remains as its synonym.

In universics, the notion "universe" is treated on the same footing as the notion "theory", and the phrase "universe of discourse" may create even more difficulties in forming correct expressions. But, the term "domain of discourse" will not be used in this paper, and the term "universe of discourse" will *not* create problems, since it will be used only in proper contexts. Also, another phrase starting with particle "of" will follow the word "universe" according the "pattern" of forming the term *universe of sets* customarily used interchangeably with the expression "the universe of discourse of (a) set theory". Similarly, the terms *universe of ideas* (as a generalization of "theory"), *universe of objects* (for the universe described by a theory), *universe of structures* (for any of the previous two) will be here preferred to the longer expressions using the phrase "of discourse".

Fraenkel used the term "object" for the entities in a universe, and since anything populating a universe is called "object", the expression "universe of objects" does not sound to be always convenient. There

is, though, a special case, when this term is the most appropriate one, and only this case occurs in this paper – this is when one opposes a "universe of ideas" to a "universe of objects". Thus, the term *universe of objects* will be used here to express a meaning opposed to that of the term *universe of ideas*. The totality of relationships between elements of one of these universes with those of the other universe is called here *reflexion* – a generic term for both the "interpretation" of elements of the universe of ideas into the universe of objects, and for the inverse relations of "naming" and "describing" the latter through the former ones.

An example of a "universe of objects" is the totality of all the things called "sets", "classes", "classes-as-many" (a term introduced by Russel to refer to collections which reflect the plural of a noun; the regular classes were shown to incorrectly reflect it), "multi-sets" , or "aggregates" (a generic term introduced also by Russel seemingly for any kind of set-like objects). Here, the term "aggregation" instead of "aggregate" is preferred for any "set-likes" – both because this term is widely used in computer science, and in order to avoid confusions with many "theories of aggregates". Accordingly, the "universe of aggregations" refers to a universe containing any set-likes.

The term *collection* is commonly informally used for a notion generalizing the notions of set and of class. A collection is presupposed to have no repeating elements, i.e. to be really a set or a class, so that the "universe of collections" satisfies the extensionality axiom – an axiom, which occurs in all mainstream set theories. Mathematics does not seem to be same focused on multi-sets as it is on collections. Here, the "universe of collections" is considered as the most important universe for the metamathematics of the foundations of mathematics. On the other hand, the set-likes used in building data structures of informatics, obviously, do not satisfy the extensionality axiom. Therefore, here, the "universe of aggregations" is considered as the most important universe for the foundations of informatics.

The term "collection" is convenient in metamathematical analysis, especially, when the notion of "size" (see below) is irrelevant to the topic of discussion. In set theory, a set is treated as a class which is a

member of another class. There are also classes which are not members of other classes, and these are called "proper classes". Similar to category theory, in metamathematics, the modifiers "small" and "big" are appropriate to modify the noun "collection" in order to distinguish between sets and proper classes in this manner: "small collection" as synonym for "set", "big collection" as synonym for "proper class".

The modifiers "small" and "big" reference two values on a scale called "size" expressed in terms of membership relation "$\in$". Namely, a "big collection" is a maximal collection within the universe of collections governed by the membership relation, and a "small collection" is not maximal in this universe. This dimension can also be referenced as "height/depth" (depending on the perspective from which the membership relation is viewed). Accordingly, in category theory, a category with a big collection of objects and morphisms is said to be a "big category", and one with such a collection small, is called "small category".

The universe of discourse of a set theory, let this be ZF, is a proper class (i.e. a "big collection"), but there are also other proper classes, and a question arises: "what singularizes the universe of discourse of ZF among other proper classes making it a 'universe' ". This question cannot be answered in terms of "size", as this term is used today. To answer it, yet another dimension needs to be considered – one which is referenced here as "extension" in this manner: a collection C will be said to have a *smaller extension* than a collection D, if $C \subseteq D$. Notice, that in addition to being "big" in size, *the universe of discourse of ZF has the largest extension*, and this answers the question regarding what singularizes the universe of discourse among other classes. To account for both "size" and "extension" dimensions, in universics the terms *small scale* and *large scale* will be used.

Finally, notice that the word "idea" is treated here as a term. Notice, that logicians consider the things used in a theory to be of two kinds – "notions" and "assertions" and they use the generic term *idea* for them (here, the "notions" can be – "properties", "relations", "functions", "operations", etc.). A theory is also an "idea" which can be treated as an inhabitant of a *universe of ideas*, and if a universe of ideas is populated only by theories, the universe is called "universe of

theories". In universics, this is an important universe, without which such notions as "foundational completeness" cannot be explicated in precise terms.

Due to the special role of the universe of theories, certain terms like "foundation", "foundational", "fundamental" which will be explicated here, will be borrowed from it for describing (any) structure called "universe". In particular, the term "foundation of a theory" will be preferred to the term "basis of a theory". This borrowing of terms "from above", considering them as "more general", will provide for a uniform treatment of any universes and will help discover (maybe, totally unexpectedly) similarities between apparently distant notions.

3 What is a universe?

Since the notion "universe" originates in logic (if only set theory is considered as a chapter of logic), one can get a hint on what kind of mathematical structure might be a universe exactly from logic, and namely from the definition of the notion "axiomatic theory", or shorter, "theory". Logicians define a theory as having a "basis" consisting of ideas of two sorts – "basic notions" and "axioms" and consider the other ideas of the theory as obtained either by definition or deduction. A common term used here for basic notions and axioms is the term "axiom". Also, since the process of definition and the process of deduction are similar, the term *reduction* is customarily used for both these processes. According to definition of "theory", a theory uses two sorts of entities – notions and assertions. But the notions can be missing from a theory, and this can be a "theory of assertions". Similarly, by admitting that the assertions can also miss from the theory, one gets a "theory of notions". Customarily, the logicians do not use such "theories", but nothing in this definition prevents these from being "theories". Any of these structures is referenced as 'universe of ideas".

Proceeding from the definition of the notion "theory", and recalling that we prefer the term "foundation" to the term "basis", the notion "universe" is defined here as a triple (U, F, R), where U is a collection called *support*, F is a subcollection of U called *foundation*, and R a

binary relation on U called *reduction* relation. As per usual practice for other types of structures, a universe and its support will be denoted and referenced by the same name or notation.

The notion defined next is not primitive, i.e. it does not occur in definition of "universe", but it is the same important as the notion "foundation", since it can be treated as its dual and called "co-foundation". But this notion is intended to serve as an explication of a notion informallly treated in [7] and its name is taken from there. Namely, the collection $U\backslash F$ (complement of F in U) is called *super-structure* of the foundation F in universe U.

In this paper, the meaning of symbol turnstile "$\vdash$" used in sequent calculi is extended to denote the reduction relation correlating any ideas, assertions or notions. A "universe of ideas" is treated as a simple mathematical structure denoted as a triple $(U, R, \text{“}\vdash\text{”})$. Moreover, the use of turnstile symbol will be further extended to stand for arbitrary binary relations. This complies with its use in an increasingly "generic" manner in so called "sub-structural sequent calculi", where this relation is not even supposed to necessarily be reflexive and transitive.

Reduction is to be treated as a "generalization" of many relations between different types of objects. Here, "generalization" is used within quotation marks to emphasize that various other kinds of relations are not really "partial cases" of reduction – they are "reducible to reduction". This means that reduction is a fundamental binary relation and even "generalization", whatever this is, must be treated in terms of reduction.

The ideas about objects and the objects (themselves) abide in a relationship called here "reflection", and the supposition is made that this relation, similar to a "homomorphism", "projects" the same structure "universe" from ideas onto objects. Therefore, a "universe of objects" will be considered here to be the same type of structure as a "universe of ideas". In this manner we came to a definition of the notion "universe", applicable both to theories and to (their) universes of discourse.

One may wonder why the notion of theory is "replaced" with the notion "universe of ideas". Aside from getting to treat theories and

their universes of discourse on the same footing, there are also other benefits from this "replacement". To illustrate this, notice that alongside many alternative "set theories", there is also one discipline called "set theory" and this kind of "theory" defies any definitions given by logicians. The theories like "set theory" turn out to be "universes of ideas", with a foundation consisting of assertions generally accepted as being about the conception "set". Such theories are referenced here as "informal theories".

A remark is also in place regarding the treatment of reduction as a *binary* relation. Notice, that in case of deduction, in a correlation like "$x \vdash y$", x is a list of assertions and y is one assertion, things of different sorts. This forces considering both the lists of assertions and the assertions as things of one sort, "object". Similarly, a notion is generally reducible to a set of other notions and a "set of notions" is of a sort different from "one notion". This forces considering reduction of notions also as a relation between things of the same sort, "ideas". The reason why reduction is treated as a relation between two things, and not between one thing of a sort and many things of the same sort, is that the reduction relation *implicitly presupposes* the existence of a multitude of things to which one thing is reducible. It is exactly this implicit presupposition which made possible development of an alternative set theory to be presented in a future paper.

The symbol "$\dashv$" symmetric to "$\vdash$" will be also used. The formula "$x \dashv y$" is read "x is reducible to y" or "x is generated from/by y", and is called "direct presentation" of reduction ("direct", because the order of arguments in formula and in English expression coincides). This "presentation" is more convenient in many cases. The formula "$x \vdash y$" is read "x reduces to y" or "x generates y and is called "inverse presentation".

As a standard notation for the property "to be a member of foundation" can serve the symbol of reduction "$\vdash$" used as a *unary* predicate symbol as in expression "$\vdash x$" (which is regularly used in sequent calculi), and symmetrically, the symbol "$\dashv$" like this: "x $\dashv$". Notice, that the notations "$x \vdash$", "$\dashv$ x" have another meaning, and in order to avoid confusions, remember for this meaning to place the argument against

the "dash", and not the "bar", of the turnstile symbol. Such use of the symbol of reduction for a unary predicate symbol shows that the property "to be in foundation" can be intuitively treated as a "rudiment" of the relation "to be reducible to".

The reason why we focused in so much detail on the two presentations is due to our treatment of induction and deduction as two principles "dual" to one another, i.e. as *de facto* one principle governing the two *symmetrical* universes – the universe of ideas and the universe of objects. It sounds like a good practice to use the symbol "$\vdash$" for deduction and its generalizations, and the symbol "$\dashv$" for all other cases. Also, it is not easy to remember which presentation is "direct" and which is "inverse", but the terms "deductive presentation" and "inductive presentation" are more suggestive and will be preferred (it will become clear later, why "inductive"). This notation practice suggests that membership relation "$\in$" is treated as "dual" to deduction "in a certain sense", and this sense will become clear after precise terms are introduced in the next section. Meanwhile, here are two important classes of universes, dual to each other.

1. $(V, F,$ "$\in$"$)$, where V is the universe of discourse of a "collection theory" (a set theory or a class theory), and F is the set of its atoms. Call this kind of universes "collection universes".

2. $(U, F,$ "$\vdash$"$)$, where U is the set of formulas in a predicate language, F is a subset of closed formulas from U considered as axioms, and "$\vdash$" is the symbol of deduction. Call this kind of universes "deduction universes".

The intuitive meaning of the notion "foundation of universe" is "the totality of all objects from which all objects in the universe can be "built" or "deducted", where "to build" and "to deduct" are described as "dual" actions to one another. The set of atoms is the foundation of a collection universe, and "the set of 'possible axioms', i.e. of formulas from which the axioms for axiomatic theories can be selected" is the foundation of the deduction universe. From this treatement it follows that the notion of deduction universe can be treated as an explication of an "informal theory", since an informal theory is said to be "axiomatized" by an axiomatic theory, and such axiomatization is nothing else,

but selecting a set of axioms from among the assertions considered as "fundamental" for the conception of "set".

Notice, that the foundation of a collection universe without atoms is the empty set. Thus, the empty set the existence of which is required in any set theory is to be treated as a foundation, which is "mandatory". This shows that even though the "universe of discourse" of set theory is customarily said to be (just) a "class", it is actually treated by set theorists as a full-fledged "universe" as this notion is defined in this paper! Here is a statement of Friedman, which sounds relevant to this: "The viewpoint is that the empty set of set theory has a unique unequivocal meaning independently of context".

4 Basic notions related with universes

The first "basic notion" defined here is easier to be understood if the reduction is denoted in its "inductive presentation", i.e. as "$\dashv$".

Definition. A sub-collection X of a universe $(U, F,$ "$\dashv$"$)$ is called *transitive collection* in universe U, if the following condition, called *fundamentality principle*, is satisfied:

$$(\forall u, v \in U)((v \in X)\&(u \dashv v) \rightarrow u \in X). \tag{1}$$

The inductive presentation of reduction was preferred, because if reduction is membership, then what was defined above is exactly the well-known property for a sub-collection of a universe of sets to be called "transitive". In a dual presentation, this condition is a kind of "dual *modus ponens*" law for a sub-collection:

$$(\forall u, v \in U)((v \in X)\&(u \vdash v) \rightarrow u \in X). \tag{2}$$

A universe is called *fundamental*, if its foundation is transitive. The mainstream collection universes, where reduction is membership, are trivially fundamental by definition of atoms (see section 6). The non-mainstream collection universes, like the universe of discourse of Aczel's set theory with the "anti-foundation" axiom, are also fundamental. Basically, all theories of sets are fundamental, trivially or non-trivially.

The deduction universes are "fundamental" only if condition (2) is satisfied. Even though *modus ponens* is the main law in the axiomatic theories, the "dual *modus ponens*" law (2) might not hold for a deduction universe. At this point, it is appropriate to explain this term. Notice, that unlike the *modus ponens* law (also called "detachment law"), where the consequent is "detached", in (2) the antecedent is "detached", and this explains the use of the word "dual". For comparison, notice that if the subcollection X is a theory, then the (regular) modus ponens law, which holds for the theory, looks like this:

$$(\forall u, v \in U)((u \in X)\&(u \vdash v) \to v \in X). \tag{3}$$

The intuitive meaning of "fundamentality principle" is expressed by this reading: "if an idea is fundamental, then another idea to which it can be reduced is also fundamental". This principle makes little sense for axiomatic theories, where the axioms are already chosen as the "fundamental ideas", but it makes a lot of sense for informal theories. So, the intuitive set theory is an informal theory – it is a universe of statements among which some statements are considered as mandatory for describing the conception "set", i.e. are regarded as "fundamental". If a statement B is regarded as fundamental, and later another statement A was found, such that $A \vdash B$, then the fundamentality principle prescribes to consider "fundamental" also the statement A. Thus, fundamentality principle can be used in development of axiomatic theories, which "axiomatize" an informal theory.

The intersection of all transitive subcollections of U containing the collection X is called *transitive closure* of X in U and is denoted as "$[X]$". This is a simple notion for collection universes, but it permits to discuss also about deduction, and in a rather concise manner. So, if X is the axiom set of theory, then "$[X]$" is the set of its theorems.

The *co-universe* of a universe $(U, F, \text{“}\dashv\text{”})$ is defined as universe $(U^c,\ F^c,\ \text{“}\dashv^c\text{”})$, where $U^c = U$, $F^c = U \backslash F$ (the superstructure in U), and "$\dashv^c$" graphically coincides with "$\vdash$". The universe and its co-universe are said to be "dual" to each other. Notice, that a universe dual to a universe U is not just a universe with the symmetric presentation, but also with the superstructure in U as its foundation.

In a customary manner (as for example, in category theory), a "dual notion" with prefix "co-" added to its name is defined for each notion. The superscript "c" will be used in the denotation of a dual notion as above in the definition of the notion "co-universe" or, for example, in notation $[X]^c$ for the co-transitive closure of a subcollection X.

Obviously, the correlation $[F]^c = [F^c]$ takes place, which in words sounds like this: "the co-transitive closure of the foundation is equal to the transitive closure of the superstructure". Therefore, $[F]^c = U$ and $[F^c] = U$ are two equivalent conditions. In words, both have the meaning: "the foundation generates the universe".

This is the right place to explain why the term "foundation" is better than the term "basis" for a universe. In mathematics, the term "basis" is customarily used as a synonym for "generating basis" or "basis of generators". Therefore, mathematicians would consider as implied the statement that "the basis of a universe generates the universe" – a statement which in general case might be wrong! On the other hand, the term "foundation" does not have this connotation of "generation" and this is why it was preferred. Still, when the foundation really generates the universe, like in case of an *axiomatic* theory, or dually, a set theory with atoms, there is no reason to avoid using the term "basis" for the foundation of a universe.

The notions as direct product, homomorphic image, etc. for universes are defined in the customary manner, but since these are not used in this paper, they will not be formulated here. The only notion in addition to those already introduced which is needed here is that of a "sub-universe" and this is defined here like this: a universe U is said to be a sub-universe of the universe V, if the support, the foundation, and the reduction relationship of U (treated as a collection of ordered pairs) are included in the support, foundation and reduction relationship of V, respectively.

5 The universe of structures

The universes of ideas and the universes of objects are structures of the same type, and the distinction between them can be made only in

terms of various relations between them. These two kinds of universes are "structures", and one would like to look into the reduction relation *between structures*, before anything else.

If x and y are two *structures*, then the expression "x $\dashv$ y" is conveniently read as "x is a *reduct* of y". The term "reduct" used here comes from universal algebra, where an algebra A is said to be a reduct of an algebra B, if the signature of A is a subset of the signature of B. This can be imagined as "reducing" the B to A by "neglecting" or "ignoring" some of its operations. An intuitive synonym for "reduct" is "rudiment" or "rudimentary structure". In some cases, the expression "x $\dashv$ y" is conveniently read as "x is more *elementary* than y", and in one of such cases, when reduction is membership, even more conveniently: "x is an *element* of y".

The collections can be treated as "final reducts" of mathematical structures, since these have a collection as their support. Next, an explanation follows why the term "universe" is customarily used both for a collection and a structure which has this collection as its support. When the set theorists consider a universe of sets as an "internal model" of a set theory, no doubt they also take into account the membership relation which governs the sets in that universe. Thus, they treat such a "universe" as a structure and not as a "class". But they refer to such a structure as "class" by making use of a linguistic device called "metonymy" – naming a whole by the name of a part. Thus, the reference to a structure by its support, which is the "*final* reduct" of the structure can be treated as a result of applying a "conceptual metonymy" device.

It is by applying the conceptual metonymy device, that other reducts of a universe are also called "universes". One of such reducts is of the kind (U, "$\dashv$") obtainable by ignoring the foundation. Such a universe will be said to be "foundation-free" or (in some cases) "base-free". The most representative example of such a universe is the universe $(V, \text{“}\in\text{”})$ of discourse of ZF, where "V" is the standard notation of the class of all pure sets (i.e. sets built out of the empty set). One cannot just "identify" (consider "the same") the pair $(U, \text{“}\dashv\text{”})$ and the triple $(U, \emptyset, \text{“}\dashv\text{”})$, since "*without* foundation" is not the same as "*with* an

empty foundation". Instead, one can use the conceptual metonymy device and call "universe", or more precisely "foundation-free universe", the pair $(U, \text{“}\dashv\text{”})$.

There is yet another kind of reduct of a universe – a reduct obtained by discarding the reduction relation to obtain the ordered pair $P = (U, F)$. Such a universe can be treated as a "problem", where U is the collection of "possible solutions", and F – the set of "actual solutions", of the problem P. This type of "universes-problems" was proposed by Kolmogorov as an alternative interpretation of intuitionism. Finally, the reduct obtained by discarding both the reduction and the basis is a collection – thus, by using the conceptual metonymy device, the collections will be also referenced as "universes".

A *proper* definition of universes as structures, a definition accounting for the reducts of universes so that the metonymy device is *not* needed, *cannot* be formulated in the language of set theory other than by re-defining the notion of relation in a complicated manner. But such a "re-definition" can create risks of ontological and terminological inconsistency. There is, though, an approach, which offers a convenient device for the presentation of universes as structures – the approach presented in [6] which uses the notion "quasiary relation". Roughly, a quasiary relation is a relation with optional correlates. It also sounds plausible, that the notion of "conceptual metonymy" can be explicated in terms of quasiary relations.

An important question is whether any type of structures is reducible to structures of type "universe". The author did not research this, but there is a result of Quine which sounds to give an affirmative reply to this question. Namely, Quine showed that the combinatory logic of Moses Schoenfinkel can be interpreted as a logic of relations (rather than functions) [8]. This result can serve as a basis for the belief, that all possible kinds of structures can be represented as universes.

The final remark in this section is for terminologic purposes. Since there are two objects, which are more "rudimentary" than a binary relation, then the relation together with these two objects can be treated as a "generalized relation" and this is exactly a "universe". Thus, we obtain an important intuitive definition: *A universe is a "generalized*

binary relation". This treatment is needed for unification of terminology (say, the notion "well-founded" is commonly defined for relations, and here – for universes).

6 Atoms and axioms – dual irreducible objects

Recall that in this paper, the term "axiom" is considered as generalizing the terms "basic notions" and "axioms" of an axiomatic theory. In universics, such "axioms" and the set-theoretic "atoms" are treated as dual, which permits "projecting" the properties of ones to the others. The atoms and the axioms can be called "marginal objects", the atoms – "initial objects", and the axioms – "final objects". Next, the atoms are discussed and the aquired knowledge is applied to axioms.

A set theory with atoms uses a predicate symbol, customarily "$Atom(x)$", in addition to the membership symbol, and it postulates that the atoms make up a set (not a class). In such a theory the atoms are objects of a sort different from that of the sets, the atoms need to be considered as making up the foundation of the universe of discourse of the theory, since there is no other way to distinguish between different sorts. Here, as "atoms" the "regular atoms", those also called "urelements", are referenced. The urelements do not have elements, and are different from the empty set.

A "Quine atom" is an object q which equals to its singleton $\{q\}$, or in other words, a Quine atom is a set (!) "on which the membership relation is reflexive". Thus, a Quine atom is not a proper "atom" – it is a singleton, a special kind of set. Hence, a set theory whose all atoms are Quine atoms is a "pure set theory". If q is a Quine atom in a set theory, then there exists an infinite chain in the universe of discourse of this theory: $q \in q \in \ldots$. A Quine atom is a non-well-founded object, and the transfinite induction principle cannot be proved in a set theory with Quine atoms.

By analogy with set theory, the definitions for two kinds of atoms in arbitrary universes are these: $Urelement(x) \stackrel{\text{def}}{=} \neg\exists y(y \dashv x)$ and $QuineAtom(x) \stackrel{\text{def}}{=} \forall x((y \dashv x) \leftrightarrow y = x)$. In any mainstream "set

theory with atoms", the atoms are always of exactly one of these two sorts, and there is no need to introduce a general term for them. But in universics, the following definition of "atoms", which are either "urelements" or a "Quine atoms", makes sense:

$$Atom(x) \stackrel{\text{def}}{=} \forall y((y \dashv x) \to y = x).$$

There is no requirement that the foundation of a universe consists only of atoms, but if this is the case, then the elements of the foundation are "pairwise incomparable", i.e. the following condition is satisfied:

$$(\forall x, y \in F)((x \dashv y)\&(y \dashv x) \to x = y).$$

Such a "foundation" is called here "basis", or "orthogonal basis", and this complies with general mathematical terminologic practices (here the modifier "orthogonal" is used rather for "emphasis"). The foundation of universes of discourse of set theories with atoms is a "basis".

And now, these notions will be used in the "dual presentation" to clarify the terminology regarding the axioms of an axiomatic theory. The logical terminology is not the same "brushed up" one as that of set theory. Really, a statement, which can be deduced from "axioms", cannot be said to be "axiom" – at most it can be called "intended axiom". But in practice, expressions like "set of independent axioms" are often used, and "axioms", which are deducible from other axioms are accepted for an axiomatic theory. A better term for them is "fundamental statements".

In order to treat the "axioms" of an axiomatic theory and the "atoms" as duals, the term "axiomatic set theory" needs to be treated as "axiomatic set theory with 'independent axiom set' " and to maintain the terminology of universics consistent, this convention is here adopted.

7 What is foundational completeness?

Foundationally complete can be the "axiomatic theories" and "$\vdash$" is the convenient symbol for reduction of the theories treated as universes.

The expression "$x \vdash y$" will be read here as "x is more fundamental than y", and the formula "$F(x)$" as "x is fundamental".

Definition. Suppose $(T, A, \text{“}\vdash\text{”})$ is an axiomatic theory and this theory is a sub-universe of the universe $(U, F, \text{“}\vdash\text{”})$. Then the theory T is said to be *foundationally complete* in the universe U, if $[A] = F$.

If in this definition, "U" is a well-founded universe (see below), then the equality $[A] = U$ is definitely true, but using this condition in the definition would limit it to only well-founded universes. Thus, as it was formulated, this definition provides for the most large applicability of this notion, including, to the *non*-well-founded universes.

To get a better perception of this notion, a couple of examples of "wordings", which are close in meaning with the expression "foundationally complete", is in place (to give "precise" examples is impossible, since this is an "explication" and not a definition of a precise notion):

(a) if U is the collection of all assertions considered as true in intuitive set theory, and T is an axiomatic theory, then T is "foundationally complete in U", if T is said to be an "axiomatic theory *of sets*";

(b) if U is the collection of all "mathematical theorems", and T is an axiomatic theory, then T is "foundationally complete in U", if the theory T is said to be "foundations for mathematics".

8 Axiomatic universe theories

Similarly to set theory which is both an informal theory and a collection of "axiomatic set theories" focused on various explications of the conception "set", universics is an informal theory of universes and a collection of axiomatic "universe theories". Various axiomatic theories of Grothendieck universes can serve as examples of the latter kind. In order that universics obtains a practical use, alongside serving as a framework of "universe theories", it must also take over from concrete theories strict treatment of some of the most general subjects. In this paper, "fundamentality" and "well-founded-ness" are considered to be among such subjects, and these are treated here in terms of axiomatic theories. These theories describe the most general features of universes,

and their axioms and axiom schemes are referenced as "principles" to be distinguished from the axioms and axioms schemes of more "special" theories based on them.

The language of universe theories introduced in this paper will use a 1^{st} order predicate language with a unary predicate symbol "F", where "$F(x)$" is interpreted in a universe with a foundation F' as "$x \in F'$" – a formula read as "x is fundamental" (or as "x is foundational" for a universe *of theories*, where some theories can be "foundational"), as well as the binary predicate symbol "$\dashv$" and its "symmetrical" symbol "$\vdash$". Obviously, the theory in this language of all universes – denote it as "**U**" – cannot contain any non-logical axioms, since for any such "axiom" there exists a universe which falsifies it. Thus, only theories describing a class of universes narrower than U are interesting.

Universics explicates the property "to be fundamental' of objects and ideas (notions, assertions, but also *theories*), via the theory **F** with the principle (F) below as its only axiom:

$$(\forall\ \mathrm{x},\ \mathrm{y})\ ((\mathrm{x}\ \dashv\ \mathrm{y})\ \&\ \mathrm{F(y)}\ \rightarrow\ \mathrm{F(x)})\quad (F).$$

The theory **F**, obviously, describes all fundamental universes, in particular, the universes of all mainstream set theories, where the "foundation axiom" in any form is either postulated or deducible, but also the universe of discourse of non-well-founded theories, including Aczel's set theory with its "anti-foundation" axiom. Thus, the "fundamentality" is treated in universics in so wide manner that most (or maybe all) set theories proposed for the foundation of mathematics are "fundamental". Obviously, of particular interest is the fundamentality principle for the universe *of theories*, where some theories are "fundamental" whereas others are not.

In universics, "well-founded-ness" property is explicated via the theory **R** with the following axiom scheme (R) as its sole axioms:

$$(\forall \mathrm{x}(\mathrm{F(x)}\ \rightarrow\ \mathrm{P(x)})\ \&\ \forall \mathrm{x}(\forall \mathrm{y}((\mathrm{y}\ \dashv \mathrm{x})\ \rightarrow \mathrm{P(x)})\ \rightarrow \mathrm{P(x)}))\rightarrow \forall \mathrm{x}\mathrm{P(x)}\ (R).$$

In this "principle", P is a formula and "x" is a variable which may or may not enter in P, together forming a "property". The theory **R**

describes the universes said to be "*reductive*". To get the first idea about these universes, one can analyze "how they look" in particular cases.

If the reduction relation is membership, then by replacing the symbol "$\dashv$" with the symbol "ϵ" in (R), one can easily discover that the reduction principle is actually the well known epsilon-induction principle generalized also to describe universes with atoms or other non-well founded objects residing in its foundation.

If the reduction is deduction, then in the particular case, when $F(x)$ is the property "x is an axiom" and $P(x)$ is the property "x is true", the reduction principle states a semantic characteristics of axiomatic theories. In the most general setting, the meanings of reduction principle are much wider, and other meanings can be obtained as the result of profound research.

9 Well-founded universes

The notion of well-foundedness in most general treatment is a property of relations (https://en.wikipedia.org/wiki/Well-founded_relation), where the collection, on which a relation is defined, is also taken into account. One can say that this property is currently defined only for the foundation-free universes, and they would obviously expect that this property can be extended to arbitrary universes.

For the foundation-free universes, the induction (epsilon-induction) cannot start from the "induction basis", which is empty, and one would naturally also want to generalize induction to its most general form with "induction basis" as foundation of any universe. This would be the generalization of induction to arbitrary "generalized relations", i.e. to arbitrary universes. In universics, the conjunction $(F)\&(R)$ stands for the "generalized induction principle", and the theory with these two principles as its sole axioms describes the class of all reductive fundamental universes. This class is treated here as the "largest span" of induction principle.

Alongside the axiomatic characterization of reductive fundamental universes, one would expect that these can also be characterized in

terms of their structure. Such a characterization is known for the well-founded relations: a relation R on a collection U is well-founded, if and only if, it contains no countable infinite descending chains, that is, there is no infinite sequence $x_0, x_1, x_2, ...$ of elements of U such that $x_{n+1} R\, x_n$ for every natural number n. Considering that this is a characterization of well-founded foundation-free universes, and terming a foundation-free universe with this "chain condition" a "Noetherian universe", one can expect a similar characterization for any reductive fundamental universe, and this will be given below. One may wonder weather the term "Noetherian" was correctly used here, or the term "Artinian" should have been chosen. Actually, the term "Noetherian" is correctly chosen, because this definition can be equivalently formulated in terms of *encreasing* chains of "*ideals*" like in ring theory. This was not done because the "ideals" in arbitrary universes would have no other uses in current paper.

A universe $(U, F, \text{“}\dashv\text{”})$ is called *quasi-Noetherian* if any countable infinite descending chain in U continues from some point in F, that is, if $x_0, x_1, x_2, ...$ is a sequence of elements of U, such that $x_{n+1} \dashv x_n$, for every natural number n, then there exists a natural number n such that $x_m \,\epsilon\, F$, for any m, where $m \geq n$. Obviously, any quasi-Noetherian foundation-free universe is Noetherian.

Theorem. A universe U is both fundamental and reductive, if and only if, U is a quasi-Noetherian universe.

The proof of this theorem, even though formulated in slightly different terms, can be found in [5], published in the Proceeding of the the first "Mathematical Foundations of Informatics", which can be downloaded at this link: http://www.mfoi.eu/workshop2015/proceedings.pdf.

10 Conclusions

The following below conclusions about the use of universics impose themselves from the foregoing presentation.

(1) Since the deduction is an apparatus of metamathematics, and the language of set theory is generally considered as the language of

metamathematics – two "duals" in universics, this approach sounds to offer a convenient language for metamathematical analysis.

(2) An important application of universics can be the "projection" of the results from one domain of research to another.

(3) Usiversics can serve as a foundational theory, presumably on par with category theory. Both these theories use maximal collections, i.e. proper classes, but, unlike category theory, universics uses only one sort of entities based on such collections, the "universes", and does not invoke (or rather rarely invokes) any second sort of entities (like "morphisms" of category theory). Due to using only one sort of entities, and due to the similarity of its methods with those used in the intuitive set theory, universics can appear to be more intuitive than category theory to those "working mathematicians", who think that the diagrams of category theory "hide the intuition".

11 Future research and an open problem

Various directions of research can be easily indicated proceeding from the conclusions in previous section. But also, there is an "open problem", which prominently imposes itself – one, the solution of which would indicate how important for the foundations are the fundamental reductive universes, outline the "validity span" of principles (F) and (R), and show whether or not their validity is correlated with the predicativity of definition of set via comprehension principle.

Open Problem. Is there an axiomatic set theory with an impredicative comprehensition in which the principle $(F)\&(R)$ is not deducible?

References

[1] J. Vaananen, "Multiverse set theories and absolutely undecidable propositions." in *Interpreting Goedel, Critical Essays,* J. Kennedy ed., Cambridge University Press,

2014, ch. 9, pp. 180–208. ISBN: 9781107002661. DOI: http://dx.doi.org/10.1017/CBO9780511756306.013

[2] J. D. Hamkins, "The set-theoretic multiverse: A natural context for set theory," *Annals of the Japan Association for Philosophy of Science*, vol. 19, no. 1, pp. 37–55, 2011,

[3] I. Drugus, "Universics: a Common Formalization Framework for Brain Informatics and Semantic Web," in *Web Intelligence and Intelligent Agents,* Z.-U.-H. Usmani ed., InTech Publishers, 2010, ch. 4, pp. 55–78. ISBN: 978-953-7619-85-5.

[4] I. Drugus, "Universics: an Approach to Knowledge based on Set theory," in *International Conference on Knowledge Engineering, Principles and Techniques, KEPT 2009, Selected papers,* Cluj-Napoca, Romania, 2009, pp. 193–200.

[5] I. Drugus, "Universics: an Axiomatic Theory of Universes. Part 1 and 2," *Workshop on Foundations of Informatics FOI-2015*, Chisinau, Moldova, 2015, pp. 118–153.

[6] M. S. Nikitchenko and S. S. Shkilniak, "Algebras of quasiary relations," *Theoretical and Applied Aspects of Program Systems Development, TAAPSD'2014: 11th international conference,* Kiev, Ukraine, 2014. pp. 174–181.

[7] D. Lewis, *Parts of classes,* Basil: Blackwell, 1991, p. 112. ISBN-13: 978-0631176558.

[8] W. V. Quine, "A reinterpretation of Schönfinkel's logical operators," *Bull. Amer. Math. Soc.,* vol. 42, no. 2, pp. 87–89, 1936.

Ioachim Drugus

Received March 14, 2016

Institute of Mathematics and Computer Science
Academy of Sciences of Moldova
5 Academiei str., Chisinau, MD-2028, Moldova
E–mail: ioachim.drugus@math.md

Computer Science Journal of Moldova, vol.**24**, no.1(70), 2016

Many-Sorted First-Order Composition-Nominative Logic as Institution

Alexey Chentsov

Abstract

In the paper the institution for many-sorted first-order composition-nominative logic (CNL) is considered. The difference from the author's previous paper on this topic is richer logical system in question due to addition of operations and sorts, and also a slightly weakened constraint on signature morphisms regarding the set of names. The satisfaction condition is proven. Some directions for further research are outlined.

Keywords: Institution theory, many-sorted nominative data, irrefutability.

1 Introduction

Composition-nominative logics (CNL) are program-oriented algebra-based logics [1]–[3]. Many-sorted algebras of partial mappings form a semantic base of CNL. Mappings are defined over classes of nominative data considered in integrity of their intensional and extensional components [2]. The hierarchy of nominative data induces a hierarchy of CNLs. Properties of composition-nominative logics are quite well-studied [1], [3], [4]. Still there is a need to relate the results obtained for these logics to other logics. This can be achieved using such theoretical tools as institutions [5], [6].

Institutions are a unified framework that allows studying properties of logical systems in abstract way independently of notation [5], [7]. Institutions capture a lot of common features of different logics. So considering the logical system one is interested in presenting it as institution and finding out what specificity the obtained institution has.

This paper continues work started in [8], [9]. It aims to construct the institution for many-sorted first-order CNL. This is done in usual fashion when all necessary elements of the corresponding institution are gradually defined starting from category of signatures and ending with checking of satisfaction condition. The difference from one-sorted case is additional structure of sorts. It primarily affects variables and terms. Most compositions remain intact. However, some sort-awareness yet should be considered.

2 Indexed families of sets

In order to identify sorts in the system, we use indexed families of sets and functions. There are two approaches to the definition of the indexed families. The first one is conventional and most commonly used in the literature. Its systematic account can be found in [6]. The second approach is based on fibers. The reasoning behind it is presented in [10]. Some results concerning the connection between the approaches are listed in [11]. In this section we only recall necessary concepts and work out notation convention.

Definition 1. *Given a set of sorts S, an* S-sorted set $\mathcal{B}$ *is an object of the category* $\mathbb{S}et^S$. *Usually it is denoted* $(B_s)_{s\in S}$. *An* S-sorted map *is a morphism of the category* $\mathbb{S}et^S$.

It is known that the category $\mathbb{S}et^S$ is equivalent to the slice category $\mathbb{S}et/S$ [12], [13, sec. 7.9]. Where convenient, we use slice category constructions. It is stylistically closer to single-sorted case (provides easy transition by forgetting the sorts). It also allows to save writing by avoiding subscripts. The difference between two categories is that fibers of the object of $\mathbb{S}et/S$ are always disjoint while sets in the indexed family have no such restriction.

Consider an S-sorted set $\mathcal{A} = (A_s)_{s\in S}$. If sets A_s are pairwise disjoint, then there is a total function $T_A\colon A \to S$, where $A = \bigcup\!\!\!\cdot_{s\in S} A_s$. Dot in the middle of symbol for union emphasizes that arguments are pairwise disjoint. Thus pair (A, T_A) determines indexed family. If the disjointness condition does not hold we can use coproduct $A =$

$\coprod_{s\in S} A_s$. There is a canonical map $T_A\colon A \to S$ such that $T_A(i_s(a)) = s$ for all $s \in S$, $a \in A_s$, where i_s is a coproduct injection. That is the following diagram commutes

$$\begin{array}{ccc} A_s & \xrightarrow{i_s} & \coprod_{s\in S} A_s \\ {\scriptstyle !_{A_s}}\downarrow & & \downarrow{\scriptstyle T_A} \\ 1 & \xrightarrow{s} & S \end{array}$$

In both cases, we use slice category to represent $\mathcal{A}$. We write $\mathcal{A} = (A, T_A)$. In this representation S-sorted map from (A, T_A) to (B, T_B) is a function $f\colon A \to B$ such that the following diagram commutes.

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ & {\scriptstyle T_A}\searrow \quad \swarrow{\scriptstyle T_B} & \\ & S & \end{array}$$

It is usually quite straightforward to recover presentation in $\mathbb{S}et^S$ from $\mathbb{S}et/S$ representation.

For a given *reindexing* $\varphi\colon S \to S'$, there are reindexing of S-sorted and S'-sorted sets described as "change of base" [13, sec. 9.7]. For any S-sorted set $\mathcal{A} = (A, T_A) = (A_s)_{s\in S}$, the corresponding S'-sorted set is $\varphi(\mathcal{A}) = (A, \varphi \circ T_A)$. Its fibers are defined as follows:

$$\varphi(\mathcal{A}) = \left(\coprod_{\varphi(s)=s'} A_s \right)_{s'\in S'} .$$

If $\mathcal{A}' = (A, T'_A)$ is an S'-sorted set, then we have an S-sorted set $\varphi^*(\mathcal{A}') = (A_{\varphi(s)})_{s\in S}$ defined by pullback along φ. This transition can be demonstrated by the following pullback diagram:

$$\begin{array}{ccc} \coprod_{s\in S} A_{\varphi(s)} & \longrightarrow & A \\ {\scriptstyle T_A}\downarrow & \lrcorner & \downarrow{\scriptstyle T'_A} \\ S & \xrightarrow[\varphi]{} & S' \end{array}$$

The transitions have functorial behavior and can be applied to S-sorted maps as well.

3 Syntactic part

3.1 Language

Definition 2. *A* signature *of many-sorted first-order composition-nominative logic is a tuple $\Sigma = (S, \mathcal{V}, P, \mathcal{F})$, where S is a set of sorts, $\mathcal{V}$ an S-sorted set of names, P a set of predicate symbols and $\mathcal{F}$ an S-sorted set of operation symbols.*

Sentences of the language, called formulas, are constructed using symbols from the signature and a number of special *composition* symbols. Composition symbols form a tuple

$$C = \big(\vee, \neg, \{\exists x\}_{x\in V}, \{S^{v_1\dots v_n} \mid \bar v \in V^n, v_i \neq v_j \text{ for } i \neq j\},\\ \{`x\}_{x\in V}, \{S_F^{v_1\dots v_n} \mid \bar v \in V^n, v_i \neq v_j \text{ for } i \neq j\}\big).$$

Here traditional compositions: $\neg$ – negation, $\vee$ – disjunction, $\exists x$ – existential quantifier. Composition $`x$ is called *denomination*. Compositions $S^{v_1\dots v_n}$, $S_F^{v_1\dots v_n}$ are *substitutions in formula* and *in term* respectively. $\bar v$ denotes sequence $v_1 \dots v_n$. There is a uniqueness constraint on names v_i in substitution: $v_i = v_j$ only if $i = j$. Usually composition symbols are not explicitly included into signature because they are fixed and fully determined by $\mathcal{V}$.

First, we define the S-sorted set of terms $\mathcal{T}(\Sigma) = (\mathrm{Ter}, T)$. The definition is mutually inductive for terms and their typing (here we use notation similar to [14])

$$\begin{array}{rcll} \tau & ::= & \alpha & : \ T_F(\alpha) \\ & & `x & : \ T_V(x) \\ & & S_F^{v_1\dots v_n}(t; t_1 \dots t_n) & : \ T(t). \end{array} \qquad (1)$$

Here $\alpha \in F$, $x, v_i \in V$, $i = \overline{1,n}$, t, t_i are terms. The terms t_i satisfy condition $T_V(v_i) = T(t_i)$ for all $i = \overline{1,n}$. Sorts after semicolons

determine $T(\tau)$. The following notation for substitution is used:

$$[v_1 \mapsto t_1, \ldots, v_n \mapsto t_n]\, t = [\,\bar{v} \mapsto \bar{t}\,]\, t = S_F^{v_1 \ldots v_n}(t; t_1 \ldots t_n).$$

The class of Σ-*sentences* is based on class of terms and defined inductively:

$$\begin{array}{rcl} \Phi & ::= & \pi \\ & & \neg\Psi \\ & & \Psi \vee \Psi' \\ & & \exists x \Psi \\ & & S^{v_1 \ldots v_n}(\Psi; t_1 \ldots t_n), \end{array} \tag{2}$$

where $\pi \in P$, $x, v_i \in V$; Ψ and Ψ' are formulas, $S^{v_1 \ldots v_n}$ – substitution in formula. Once again there are typing constraints: $T(t_i) = T_V(v_i)$ for all $i = \overline{1,n}$. We use notation

$$[v_1 \mapsto t_1, \ldots, v_n \mapsto t_n]\, \Phi = [\,\bar{v} \mapsto \bar{t}\,]\, \Phi = S^{v_1 \ldots v_n}(\Phi; t_1 \ldots t_n).$$

Implication, conjunction and universal quantifier are defined conventionally as follows

$$\begin{aligned} \Phi \wedge \Psi &= \neg(\neg\Phi \vee \neg\Psi) \\ \Phi \to \Psi &= \neg\Phi \vee \Psi \\ \forall x \Phi &= \neg\exists x \neg\Phi \\ R^{v_1 \ldots v_n}_{x_1 \ldots x_n} \Phi &= [v_1 \mapsto {\text{`}}x_1, \ldots, v_n \mapsto {\text{`}}x_n]\, \Phi. \end{aligned}$$

Composition $R^{v_1 \ldots v_n}_{x_1 \ldots x_n}$ is called *renomination* and usually abbreviated as $R^{\bar{v}}_{\bar{x}}$. Uniqueness constraint transfers to the set of *upper* names v_i of renomination.

3.2 Signature morphisms and sentence translation

Definition 3. *A* morphism *of signatures is*

$$\varphi = (\varphi_S, \varphi_V, \varphi_P, \varphi_F) \colon (S, \mathcal{V}, P, \mathcal{F}) \to (S', \mathcal{V}', P', \mathcal{F}'),$$

where $\varphi_P\colon P \to P'$, $\varphi_S\colon S \to S'$ *is a reindexing,* $\varphi_F\colon \varphi_S(\mathcal{F}) \to \mathcal{F}'$ *an* S'*-sorted map,* $\varphi_V\colon \varphi_S(\mathcal{V}) \to \mathcal{V}'$ *an injective* S'*-sorted map. In other words the following diagrams commute:*

$$\begin{array}{ccc} V & \xrightarrow{\varphi_V} & V' \\ {\scriptstyle T_V}\downarrow & & \downarrow{\scriptstyle T'_V} \\ S & \xrightarrow[\varphi_S]{} & S' \end{array} \qquad \begin{array}{ccc} F & \xrightarrow{\varphi_F} & F' \\ {\scriptstyle T_F}\downarrow & & \downarrow{\scriptstyle T'_F} \\ S & \xrightarrow[\varphi_S]{} & S' \end{array}$$

Name component φ_V of signature morphism is restricted to 1-1 mapping to avoid name clashes in substitution (renomination) composition and to be able to extend Mod to a functor.

Our category $\mathbb{S}ig$ is simply a category of signatures and signature morphisms defined above.

Now we can extend action of signature morphism to the Σ-sentences defined in (2), i.e. define $\mathrm{Sen}(\varphi)\colon \mathrm{Sen}(S, \mathcal{V}, P, \mathcal{F}) \to \mathrm{Sen}(S', \mathcal{V}', P', \mathcal{F}')$ inductively on structure of the sentence as follows

$$\begin{aligned} \mathrm{Sen}(\varphi)(\alpha) &= \varphi_F(\alpha) \\ \mathrm{Sen}(\varphi)(\text{`}x) &= \text{`}\varphi_V(x) \\ \mathrm{Sen}(\varphi)([\,\bar{v} \mapsto \bar{t}\,]\,t') &= [\varphi_V(\bar{v}) \mapsto \mathrm{Sen}(\varphi)(\bar{t})]\,\mathrm{Sen}(\varphi)(t') \\ \mathrm{Sen}(\varphi)(\pi) &= \varphi_P(\pi) \\ \mathrm{Sen}(\varphi)(\Phi \vee \Psi) &= \mathrm{Sen}(\varphi)(\Phi) \vee \mathrm{Sen}(\varphi)(\Psi) \\ \mathrm{Sen}(\varphi)(\neg\Phi) &= \neg\,\mathrm{Sen}(\varphi)(\Phi) \\ \mathrm{Sen}(\varphi)(\exists x\Phi) &= \exists\varphi_V(x)\,\mathrm{Sen}(\varphi)(\Phi) \\ \mathrm{Sen}(\varphi)([\,\bar{v} \mapsto \bar{t}\,]\,\Phi) &= [\varphi_V(\bar{v}) \mapsto \mathrm{Sen}(\varphi)(\bar{t})]\,\mathrm{Sen}(\varphi)(\Phi). \end{aligned}$$

Here $\xi(\bar{l})$ denotes componentwise application of a function ξ to a list $\bar{l} = l_1, \ldots, l_n$, i.e. the list $\xi(l_1), \ldots, \xi(l_n)$.

Proposition 1. *The following diagram commutes:*

$$\begin{array}{ccc} \mathrm{Ter} & \xrightarrow{\mathrm{Sen}(\varphi)} & \mathrm{Ter}' \\ {\scriptstyle T}\downarrow & & \downarrow{\scriptstyle T'} \\ S & \xrightarrow[\varphi_S]{} & S' \end{array}$$

Proof. By induction on term structure. Let us check congruence rule.

$$\begin{aligned} T'\left(\mathrm{Sen}(\varphi)([\,\bar{v} \mapsto \bar{t}\,]\,t')\right) &= T'\left([\varphi_V(\bar{v}) \mapsto \mathrm{Sen}(\varphi)(\bar{t})]\,\mathrm{Sen}(\varphi)(t')\right) \\ &= T'\left(\mathrm{Sen}(\varphi)(t')\right) = \varphi_S(T(t')) \\ &= \varphi_S(T([\,\bar{v} \mapsto \bar{t}\,]\,t')). \end{aligned}$$

Here we used induction hypothesis for t', and assumed typing constraint for $[\varphi_V(\bar{v}) \mapsto \mathrm{Sen}(\varphi)(\bar{t})]\,\mathrm{Sen}(\varphi)(t')$. Let us prove the latter. By induction hypothesis, the properties of signature morphism and correctness of original term we have

$$T'(\mathrm{Sen}(\varphi)(\bar{t})) = \varphi_S(T(\bar{t})) = \varphi_S(T_V(\bar{v})) = T'_V(\varphi_V(\bar{v})). \qquad \square$$

As a result, $\mathrm{Sen}(\varphi)$ is correctly defined w.r.t. sorts.

Proposition 2. $\mathbb{S}ig$ *is a category.* Sen *is a functor* $\mathbb{S}ig \to \mathbb{S}et$.

In a context where Sen is known, expression $\mathrm{Sen}(\varphi)(\Phi)$ is usually abbreviated as simply $\varphi(\Phi)$.

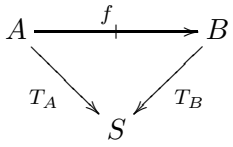
4 Models and model homomorphisms

4.1 Many-sorted nominative data

The basis for semantics of various composition-nominative logics is formed by nominative sets, quasiary predicates and operations. Let $A \neq \emptyset$ be some set, V be the set of names. A (*partial*) *nominative set* is a partial mapping from V to A, the class of all such mappings is denoted ${}^{V}A$. In this context the set A is called the set of values, ${}^{V}A$ – the set of nominative sets or set of *states*. Nominative sets can be also called *nominative data*. By analogy with single-sorted case, we define many-sorted nominative data.

Definition 4. *A* partial S-sorted map $f\colon (A, T_A) \nrightarrow (B, T_B)$ *is a partial map* $f\colon A \nrightarrow B$ *such that the following diagram commutes in a*

weak sense



i.e. $T_B \circ f = T_A\big|_{\operatorname{dom} f}$.

S-sorted sets and S-sorted partial maps form a category S-$\mathbb{S}et_{part}$.

Definition 5. *Let $\mathcal{V}$, $\mathcal{A}$ be S-sorted sets. An* S-sorted $\mathcal{V}$-nominative set *is a partial S-sorted map $d\colon \mathcal{V} \nrightarrow \mathcal{A}$.*

Let $\mathcal{V} = (V_s)_{s\in S}$, $\mathcal{A} = (A_s)_{s\in S}$. The class of all S-sorted $\mathcal{V}$-nominative sets is also an S-sorted set ${}^{\mathcal{V}}\mathcal{A}$ defined as follows:

$$ {}^{\mathcal{V}}\mathcal{A} = ({}^{V_s}A_s)_{s\in S}. $$

If we ignore sorts, then S-sorted nominative set d becomes simply a partial function $d\colon V \nrightarrow A$, where $V = \bigcup_{s\in S} V_s$, $A = \coprod_{s\in S} A_s$. In this sense there is an embedding

$$ {}^{\mathcal{V}}\mathcal{A} \hookrightarrow {}^{V}A\,. $$

Sometimes we prefer to work with such representation of $d \in {}^{\mathcal{V}}\mathcal{A}$ rather than $(d_s)_{s\in S}$.

We use the following notation in regard to partiality. Let $f\colon A \nrightarrow B$, $a, a' \in A$, $b \in B$. We write $f(a)\uparrow$ if $a \notin \operatorname{dom} f$, otherwise (if $a \in \operatorname{dom} f$) we write $f(a)\downarrow$. Here $\operatorname{dom} f = f^{-1}(B) = \{x \mid (x,y) \in f \text{ for some } y\}$ is the *domain of definition* of f. In the latter case $f(a)\downarrow$ can be used as well as the value of f on a, e.g. $f(a)\downarrow = b$. Also we use symbol $\cong$ for *strong equality* that makes allowance for undefined value, namely

$$ f(a) \cong f(a') \text{ if } f(a)\downarrow = f(a')\downarrow \text{ or } (f(a)\uparrow \text{ and } f(a')\uparrow). $$

Two partial functions f and g are equal if and only if $f(x) \cong g(x)$ for all x.

The elements of nominative data are pairs of the form $v \mapsto a$. Expression $v \mapsto a \in_n d$ denotes $d(v)\downarrow = a$. Given $v \in V_s$, $a \in A_s$

for some $s \in S$, expression $v \mapsto a$ in the context of ${}^{V}A$ means $v \mapsto i_s(a)$, where $i_s \colon A_s \to A$ is a canonical injection. Nominative sets are constructed using set-builder notation with square brackets.

Let us introduce the unary operation $r^{v_1 \ldots v_n}_{x_1 \ldots x_n} \colon {}^{\mathcal{V}}\mathcal{A} \to {}^{\mathcal{V}}\mathcal{A}$ of finite renomination of nominative set, where $T_V(v_i) = T_V(x_i)$ for all $i = \overline{1,n}$. First, we specify an S-sorted map $\sigma^{v_1 \ldots v_n}_{x_1 \ldots x_n} : \mathcal{V} \to \mathcal{V}$ associated with it:

$$\sigma^{v_1 \ldots v_n}_{x_1 \ldots x_n}(v) = \begin{cases} x_i & \text{if } v = v_i. \\ v & \text{otherwise.} \end{cases}$$

Then $r^{v_1 \ldots v_n}_{x_1 \ldots x_n} d = d \circ \sigma^{v_1 \ldots v_n}_{x_1 \ldots x_n}$, where $\circ$ denotes the composition of partial functions.

We require three more operations, *single name binding*, for $d \in {}^{\mathcal{V}}\mathcal{A}$, $u \in V_s$, $a \in A_s$, $s \in S$

$$d \triangledown u \mapsto a = d\big|_{V \setminus \{u\}} \dot\cup\, [u \mapsto a].$$

Here $\big|_W$ denotes conventional restriction of function domain to W and dot in $\dot\cup$ emphasizes that the union is disjoint. *Finite name binding*, for $d \in {}^{\mathcal{V}}\mathcal{A}$, distinct names $v_i \in V_{s_i}$, $a_i \in A_{s_i}$, $s_i \in S$ for $i = \overline{1,n}$

$$d \triangledown [v_i \mapsto a_i \mid i = \overline{1,n}] = d\big|_{V \setminus \{v_i\}_{i=\overline{1,n}}} \dot\cup\, [v_i \mapsto a_i \mid i = \overline{1,n}].$$

Finally *overriding*, for $d_1, d_2 \in {}^{V}A$

$$d_1 \triangledown d_2 = d_1\big|_{V \setminus \operatorname{dom} d_2} \dot\cup\, d_2.$$

Construction ${}^{\mathcal{V}}\mathcal{A}$ demonstrates bifunctorial behavior in the following sense. Let $\sigma \colon \mathcal{V} \nrightarrow \mathcal{V}'$ be a partial S-sorted map, and $h \colon \mathcal{A} \to \mathcal{A}'$ be an S-sorted map. They induce several total maps between nominative set domains: function ${}^{\sigma}\mathcal{A} \colon {}^{\mathcal{V}'}\mathcal{A} \to {}^{\mathcal{V}}\mathcal{A}$ that maps nominative set $d \in {}^{\mathcal{V}'}\mathcal{A}$ to nominative set $d \circ \sigma$, function ${}^{\mathcal{V}}h \colon {}^{\mathcal{V}}\mathcal{A} \to {}^{\mathcal{V}}\mathcal{A}'$ that maps $d \in {}^{\mathcal{V}}\mathcal{A}$ to $h \circ d$, and function ${}^{\sigma}h \colon {}^{\mathcal{V}'}\mathcal{A} \to {}^{\mathcal{V}}A'$ defined as $d \mapsto h \circ d \circ \sigma$. Notice that functions induced by change of set of values and set of names commute under composition:

$${}^{\mathcal{V}}h \circ {}^{\sigma}\mathcal{A} = {}^{\sigma}h = {}^{\sigma}\mathcal{A}' \circ {}^{\mathcal{V}'}h. \tag{3}$$

In these terms we have

$$r^{v_1\ldots v_n}_{x_1\ldots x_n} d = d \circ \sigma^{v_1\ldots v_n}_{x_1\ldots x_n} = {}^{\sigma^{\bar{v}}_{\bar{x}}}\mathcal{A}(d).$$

Ignoring the sorts does not change the functorial behavior of ${}^{\mathcal{V}}\mathcal{A}$.

4.2 Quasiary predicates and operations

Let $Bool = \{\top, \bot\}$ be a Boolean set. The *quasiary predicate* over S-sorted set of names $\mathcal{V}$ and S-sorted set of values $\mathcal{A}$ is a partial Boolean-valued function: ${}^{\mathcal{V}}\mathcal{A} \nrightarrow Bool$. The quasiary predicates over set of names $\mathcal{V}$ and set of values $\mathcal{A}$ are called $(\mathcal{V}, \mathcal{A})$-quasiary predicates for short. Let $Pr^{\mathcal{V}}_{\mathcal{A}} = \{p \mid p\colon {}^{\mathcal{V}}\mathcal{A} \nrightarrow Bool\}$.

The truth and falsity domains of $p \in Pr^{\mathcal{V}}_{\mathcal{A}}$ are respectively $\top(p) = \{d \mid p(d)\downarrow = \top\} = p^{-1}(\{\top\})$, $\bot(p) = p^{-1}(\{\bot\})$.

Definition 6. *The* extension *of a partial predicate p is a pair of its truth and falsity domains:* $\|p\| = (\top(p), \bot(p))$.

Notice that sets in the extension of a predicate are necessarily disjoint. There is a 1-1 correspondence between extensions and partial predicates. Also there is a natural ordering of extensions:

$$\|p\| \subseteq \|p'\| \text{ if } \top(p) \subseteq \top(p') \text{ and } \bot(p') \subseteq \bot(p).$$

Definition 7. *A predicate p is* irrefutable *if* $\bot(p) = \emptyset$.

Like the domain of nominative data ${}^{\mathcal{V}}\mathcal{A}$, the construction $Pr^{\mathcal{V}}_{\mathcal{A}}$ also has bifunctorial behavior. Given partial S-sorted map $\sigma\colon \mathcal{V} \nrightarrow \mathcal{V}'$ and total S-sorted map $h\colon \mathcal{A} \to \mathcal{A}'$, there are total maps $Pr^{\mathcal{V}}_{h}\colon Pr^{\mathcal{V}}_{\mathcal{A}'} \to Pr^{\mathcal{V}}_{\mathcal{A}}$, $Pr^{\sigma}_{\mathcal{A}}\colon Pr^{\mathcal{V}}_{\mathcal{A}} \to Pr^{\mathcal{V}'}_{\mathcal{A}}$, $Pr^{\sigma}_{h}\colon Pr^{\mathcal{V}}_{\mathcal{A}'} \to Pr^{\mathcal{V}'}_{\mathcal{A}}$ realized as follows. Let $p \in Pr^{\mathcal{V}}_{\mathcal{A}'}$, $q \in Pr^{\mathcal{V}}_{\mathcal{A}}$, then

$$Pr^{\mathcal{V}}_{h}(p) = p \circ {}^{\mathcal{V}}h$$
$$Pr^{\sigma}_{\mathcal{A}}(q) = q \circ {}^{\sigma}\mathcal{A}$$
$$Pr^{\sigma}_{h}(p) = p \circ {}^{\sigma}h.$$

Once again, notice that maps induced by change of set of values and set of names commute under composition

$$Pr_h^{\mathcal{V}'} \circ Pr_{\mathcal{A}'}^{\sigma} = Pr_h^{\sigma} = Pr_{\mathcal{A}}^{\sigma} \circ Pr_h^{\mathcal{V}}. \tag{4}$$

Analogously to the quasiary predicates, we consider *quasiary operations* over S-sorted set of names $\mathcal{V}$ and S-sorted set of values $\mathcal{A}$ as partial functions: ${}^{\mathcal{V}}\mathcal{A} \nrightarrow A_s$ for some $s \in S$. The quasiary operations over set of names $\mathcal{V}$ and set of values $\mathcal{A}$ are called $(\mathcal{V}, \mathcal{A})$-quasiary operations for short. Let $Fn_{\mathcal{A},s}^{\mathcal{V}} = \{ f \mid f \colon {}^{\mathcal{V}}\mathcal{A} \nrightarrow A_s \}$, $\mathcal{F}n_{\mathcal{A}}^{\mathcal{V}} = \left(Fn_{\mathcal{A},s}^{\mathcal{V}} \right)_{s \in S}$.

Similarly to the domain of nominative data ${}^{\mathcal{V}}\mathcal{A}$ and class of quasiary predicates $Pr_{\mathcal{A}}^{\mathcal{V}}$ construction $\mathcal{F}n_{\mathcal{A}}^{\mathcal{V}}$ also demonstrates functorial behavior, but this time only by parameter $\mathcal{V}$. Given a partial S-sorted map $\sigma \colon \mathcal{V} \nrightarrow \mathcal{V}'$, there is a total S-sorted map $Fn_{\mathcal{A}}^{\sigma} \colon \mathcal{F}n_{\mathcal{A}}^{\mathcal{V}} \to \mathcal{F}n_{\mathcal{A}}^{\mathcal{V}'}$ realized as follows: $Fn_{\mathcal{A},s}^{\sigma}(f) = f \circ {}^{\sigma}\mathcal{A}$, where $f \in Fn_{\mathcal{A},s}^{\mathcal{V}}$. Elements of $Fn_{\mathcal{A},s}^{\mathcal{V}}$ can also be thought as functorial algebras in S-$\mathbb{S}et_{part}$ (for the functor $(H_s)_{s \in S}$ such that $H_s(\mathcal{A}) = {}^{\mathcal{V}}\mathcal{A}$, and $H_{s'}(\mathcal{A}) = \emptyset$ for $s' \in S \setminus \{s\}$) [15, p. 142-143].

Given $d \in {}^{\mathcal{V}}\mathcal{A}$, distinct $v_i \in V$, $f_i \in Fn_{\mathcal{A},s_i}^{\mathcal{V}}$, such that $T_V(v_i) = s_i$, $i = \overline{1,n}$, there is a nominative data $[v_1 \mapsto f_1(d), \ldots, v_n \mapsto f_n(d)] \in {}^{\mathcal{V}}\mathcal{A}$ defined as follows

$$v \mapsto a \in_n [v_1 \mapsto f_1(d), \ldots, v_n \mapsto f_n(d)] \text{ if } \exists i \in \overline{1,n}. v = v_i, f_i(d)\!\downarrow = a.$$

For short $[v_1 \mapsto f_1(d), \ldots, v_n \mapsto f_n(d)]$ is written as $[\bar{v} \mapsto \bar{f}(d)]$. Let us introduce *substitution* operation $[v_1 \mapsto f_1, \ldots, v_n \mapsto f_n]$ for nominative sets:

$$[v_1 \mapsto f_1, \ldots, v_n \mapsto f_n]\, d = d\big|_{V \setminus \{v_i\}} \nabla [\bar{v} \mapsto \bar{f}(d)].$$

For short $[v_1 \mapsto f_1, \ldots, v_n \mapsto f_n]\, d$ is written as $[\, \bar{v} \mapsto \bar{f}\,]\, d$.

4.3 Models

The sets $\mathcal{F}n_{\mathcal{A}}^{\mathcal{V}}$, $Pr_{\mathcal{A}}^{\mathcal{V}}$ are used as a carrier sets for most composition-nominative logics. The terms are interpreted as quasiary operations

and formulas as quasiary predicates. Compositions have fixed interpretation for CNLs and are defined as follows

$$\begin{aligned}
{}'x(d) &\cong d_{T_V(x)}(x).\\
\|p \vee q\| &= (\top(p) \cup \top(q), \bot(p) \cap \bot(q))\\
\|\neg p\| &= (\bot(p), \top(p))\\
\|\exists x p\| &= \big(\{d \mid d\triangledown x \mapsto a \in \top(p) \text{ for some } a \in A_{T_V(x)}\},\\
&\qquad \{d \mid d\triangledown x \mapsto a \in \bot(p) \text{ for all } a \in A_{T_V(x)}\}\big)
\end{aligned} \tag{5}$$

$$[\,\bar v \mapsto \bar f\,]\, g(d) \cong g([\,\bar v \mapsto \bar f\,]\, d)$$

$$[\,\bar v \mapsto \bar f\,]\, p(d) \cong p([\,\bar v \mapsto \bar f\,]\, d).$$

Here $d \in {}^{\mathcal{V}}\mathcal{A}$, $x, v_i \in V$, $g \in Fn^{\mathcal{V}}_{\mathcal{A},T_V(x)}$, $f_i \in Fn^{\mathcal{V}}_{\mathcal{A},T_V(v_i)}$, $i = \overline{1,n}$, $p, q \in Pr^{\mathcal{V}}_{\mathcal{A}}$; $[\,\bar v \mapsto \bar f\,]\, g(d)$ denotes $[v_1 \mapsto f_1, \ldots, v_n \mapsto f_n]\, g(d)$, likewise $[\,\bar v \mapsto \bar f\,]\, p(d)$ denotes $[v_1 \mapsto f_1, \ldots, v_n \mapsto f_n]\, p(d)$. In these terms

$$R^{\bar v}_{\bar x} p(d) \cong [\,\bar v \mapsto {}'\bar x\,]\, p(d) \cong p([\,\bar v \mapsto {}'\bar x\,]\, d) \cong p \circ {}^{\sigma^{\bar v}_{\bar x}}\mathcal{A}(d).$$

That is

$$R^{\bar v}_{\bar x} p = p \circ {}^{\sigma^{\bar v}_{\bar x}}\mathcal{A} = Pr^{\sigma^{\bar v}_{\bar x}}_{\mathcal{A}}(p).$$

Definition 8. *A* first-order algebra of $(\mathcal{V}, \mathcal{A})$-quasiary predicates *is a tuple* $(Pr, \mathcal{F}n, \mathcal{A}; Comp)$*, where* $Comp$ *are compositions defined in* (5) *and sets* $Pr \subseteq Pr^{\mathcal{V}}_{\mathcal{A}}$*,* $\mathcal{F}n \subseteq \mathcal{F}n^{\mathcal{V}}_{\mathcal{A}}$ *are closed under compositions.*

Definition 9. *Given a signature* $\Sigma = (S, \mathcal{V}, P, \mathcal{F})$*, a* Σ-model *of many-sorted first-order composition-nominative logic is a quadruple* $(Pr, \mathcal{F}n, \mathcal{A}, I)$ *such that* $(Pr, \mathcal{F}n, \mathcal{A}; Comp)$ *forms a first-order* $(\mathcal{V}, \mathcal{A})$*-quasiary predicates algebra and* $I = (I_P, I_F)$*, where* $I_P \colon P \to Pr$ *and* $I_F \colon \mathcal{F} \to \mathcal{F}n$ *are total and* S*-sorted total maps respectively.*

Interpretation of formulas and terms in a model is straightforward. The details are presented in section 6.

4.4 Model homomorphisms

Consider the conventional case of first-order logic. Model homomorphisms are functions $h \colon A \to B$ with operation and predicate preser-

vation property. For each arity n due to contravariant powerset functor there is an induced map $\mathcal{P}_n(h)\colon \mathcal{P}(B^n) \to \mathcal{P}(A^n)$ between n-ary predicates. Preservation of n-ary predicate symbol $\pi \in P_n$ means $M_\pi \subseteq \mathcal{P}_n(h)(M'_\pi)$.

An analogous construction for quasiary case is presented in subsection 4.2. Let $h\colon \mathcal{A} \to \mathcal{A}'$ be a total S-sorted map. Consider total map $Pr_h^{\mathcal{V}}\colon Pr_{\mathcal{A}'}^{\mathcal{V}} \to Pr_{\mathcal{A}}^{\mathcal{V}}$ induced by h. Let us check its properties in regards to algebraic structure.

Proposition 3. *Function $Pr_h^{\mathcal{V}}$ preserves disjunction, negation and renomination compositions. If h is surjective, it also preserves existential quantifier composition.*

Proof. Let $p \in Pr_{\mathcal{A}'}^{\mathcal{V}}$, then

$$\top(Pr_h^{\mathcal{V}}(p)) = \{d \mid {}^{\mathcal{V}}h(d) \in \top(p)\} = \left({}^{\mathcal{V}}h\right)^{-1}(\top(p)).$$

Therefore

$$\|Pr_h^{\mathcal{V}}(p)\| = \left(\left({}^{\mathcal{V}}h\right)^{-1}(\top(p)), \left({}^{\mathcal{V}}h\right)^{-1}(\bot(p))\right) = \left({}^{\mathcal{V}}h\right)^{-1}\|p\|.$$

Let $p, q \in Pr_{\mathcal{A}'}^{\mathcal{V}}$, then

$$\begin{aligned}\|Pr_h^{\mathcal{V}}(\neg p)\| &= \left({}^{\mathcal{V}}h\right)^{-1}(\bot(p), \top(p)) = \|\neg Pr_h^{\mathcal{V}}(p)\|,\\ \|Pr_h^{\mathcal{V}}(p \vee q)\| &= \left({}^{\mathcal{V}}h\right)^{-1}(\top(p) \cup \top(q), \bot(p) \cap \bot(q))\\ &= \|Pr_h^{\mathcal{V}}(p) \vee Pr_h^{\mathcal{V}}(q)\|,\end{aligned}$$

where preservation of unions and intersections by preimage is used.

For the renomination composition we use commutativity (4):

$$Pr_h^{\mathcal{V}}(R_{\bar{x}}^{\bar{v}}p) = Pr_h^{\mathcal{V}} \circ Pr_{\mathcal{A}'}^{\sigma_{\bar{x}}^{\bar{v}}}(p) = Pr_{\mathcal{A}}^{\sigma_{\bar{x}}^{\bar{v}}} \circ Pr_h^{\mathcal{V}}(p) = R_{\bar{x}}^{\bar{v}} Pr_h^{\mathcal{V}}(p).$$

Finally, if $h\colon \mathcal{A} \to \mathcal{A}'$ is surjective, then

$$\begin{aligned}
\top(Pr_h^{\mathcal{V}}(\exists x p)) &= \left\{ d \in {}^{\mathcal{V}}\mathcal{A} \mid (h \circ d)\nabla x \mapsto a' \in \top(p) \text{ for some } a' \in A'_{T(x)} \right\} \\
&= \left\{ d \mid (h \circ d)\nabla x \mapsto h_s(a) \in \top(p) \text{ for some } a \in A_{T(x)} \right\} \\
&= \left\{ d \mid h \circ (d\nabla x \mapsto a) \in \top(p) \text{ for some } a \in A_{T(x)} \right\} \\
&= \top(\exists x Pr_h^{\mathcal{V}}(p)). \\
\bot(Pr_h^{\mathcal{V}}(\exists x p)) &= \left\{ d \mid (h \circ d)\nabla x \mapsto a' \in \bot(p) \text{ for all } a' \in A'_{T(x)} \right\} \\
&= \left\{ d \in {}^{\mathcal{V}}\mathcal{A} \mid (h \circ d)\nabla x \mapsto h_s(a) \in \bot(p) \text{ for all } a \in A_{T(x)} \right\} \\
&= \bot(\exists x Pr_h^{\mathcal{V}}(p)).
\end{aligned}$$

That is $Pr_h^{\mathcal{V}}(\exists x p) = \exists x Pr_h^{\mathcal{V}}(p)$. □

Since there is no direct transformation between $\mathcal{F}n_{\mathcal{A}}^{\mathcal{V}}$ and $\mathcal{F}n_{\mathcal{A}'}^{\mathcal{V}}$, we cannot establish similar property for arbitrary substitution but we can do it for some subset of operations.

Definition 10. *Given a total function $h\colon \mathcal{A} \to \mathcal{A}'$, an operation $f \in Fn_{\mathcal{A},s}^{\mathcal{V}}$ is* h-related to *operation $f' \in Fn_{\mathcal{A}',s}^{\mathcal{V}}$ if the following diagram commutes.*

$$\begin{array}{ccc}
{}^{\mathcal{V}}\mathcal{A} & \xrightarrow{f} & A_s \\
{\scriptstyle {}^{\mathcal{V}}h}\downarrow & & \downarrow{\scriptstyle h_s} \\
{}^{\mathcal{V}}\mathcal{A}' & \xrightarrow[f']{} & A'_s
\end{array}$$

The next proposition summarizes interaction between h and operation compositions.

Proposition 4. *For arbitrary map $h\colon \mathcal{A} \to \mathcal{A}'$ and name $x \in V$ composition $`x \in Fn_{\mathcal{A},T_V(x)}^{\mathcal{V}}$ is $h$-related to $`x \in Fn_{\mathcal{A}',T_{V(x)}}^{\mathcal{V}}$. If $g \in Fn_{\mathcal{A},s}^{\mathcal{V}}$, $f_i \in Fn_{\mathcal{A},s_i}^{\mathcal{V}}$ are h-related to $g' \in Fn_{\mathcal{A},s}^{\mathcal{V}}$, $f'_i \in Fn_{\mathcal{A}',s_i}^{\mathcal{V}}$, then substitution $[\,\bar{v} \mapsto \bar{f}\,]\,g$ is h-related to $[\,\bar{v} \mapsto \bar{f}'\,]\,g'$. If $f_i \in Fn_{\mathcal{A},s_i}^{\mathcal{V}}$ are h-related to $f'_i \in Fn_{\mathcal{A}',s_i}^{\mathcal{V}}$, $p \in Pr_{\mathcal{A}'}^{\mathcal{V}}$, then*

$$Pr_h^{\mathcal{V}}([\,\bar{v} \mapsto \bar{f}'\,]\,p) = [\,\bar{v} \mapsto \bar{f}\,]\,Pr_h^{\mathcal{V}}(p). \tag{6}$$

Proof. If $d \in {}^{\mathcal{V}}\mathcal{A}$, then $h_s(`x(d)) \cong h_s(d_s(x)) \cong {}^{\mathcal{V}}h(d)_s(x) \cong `x({}^{\mathcal{V}}h(d))$, where $s = T_V(x)$. For the second property we prove the commutativity of the diagram

$$\begin{array}{ccccc}
{}^{\mathcal{V}}\mathcal{A} & \xrightarrow{[\,\bar{v}\mapsto\bar{f}\,]} & {}^{\mathcal{V}}\mathcal{A} & \xrightarrow{g} & A_s \\
{\scriptstyle {}^{\mathcal{V}}h}\downarrow & & \downarrow{\scriptstyle {}^{\mathcal{V}}h} & & \downarrow{\scriptstyle h_s} \\
{}^{\mathcal{V}}\mathcal{A}' & \xrightarrow[[\,\bar{v}\mapsto\bar{f}'\,]]{} & {}^{\mathcal{V}}\mathcal{A}' & \xrightarrow[g']{} & A'_s
\end{array}$$

The right rectangle is commutative by condition. Let $d \in {}^{\mathcal{V}}\mathcal{A}$, $v \in V$. Suppose $v \neq v_i$ for all $i = \overline{1,n}$. Then

$${}^{\mathcal{V}}h([\,\bar{v} \mapsto \bar{f}\,]\,d)(v) \cong h([\,\bar{v} \mapsto \bar{f}\,]\,d(v)) \cong h(d(v)) \cong [\,\bar{v} \mapsto \bar{f}'\,]\,({}^{\mathcal{V}}h(d))(v).$$

Otherwise, if $v = v_i$ for some $i \in \overline{1,n}$, then

$$\begin{aligned}
{}^{\mathcal{V}}h([\,\bar{v} \mapsto \bar{f}\,]\,d)(v) &\cong h([\,\bar{v} \mapsto \bar{f}\,]\,d(v_i)) \cong h(i_s(f_i(d))) \cong i'_s(f'_i({}^{\mathcal{V}}h(d))) \\
&\cong [\,\bar{v} \mapsto \bar{f}'\,]\,{}^{\mathcal{V}}h(d)(v),
\end{aligned}$$

where $s = T_V(v_i)$ and $i_s\colon A_s \to A$, $i'_s\colon A'_s \to A'$ are canonical injections. This gives us commutativity of left rectangle. As a result outer rectangle is also commutative, i.e. the second property holds.

Let $p \in Pr^{\mathcal{V}}_{\mathcal{A}'}$, then

$$\begin{aligned}
Pr^{\mathcal{V}}_h([\,\bar{v} \mapsto \bar{f}'\,]\,p) &= p \circ [\,\bar{v} \mapsto \bar{f}'\,] \circ {}^{\mathcal{V}}h \\
&= p \circ {}^{\mathcal{V}}h \circ [\,\bar{v} \mapsto \bar{f}\,] = [\,\bar{v} \mapsto \bar{f}\,]\,Pr^{\mathcal{V}}_h(p).
\end{aligned}$$

Here we used the commutativity of left rectangle once again. □

Thus we only need to formalize preservation of predicates by map h. There are several ways to accomplish this. Here we do it similarly to the conventional case using the extensions of quasiary predicates.

Definition 11. *A $(S, \mathcal{V}, P, (F_s)_{s\in S})$-model homomorphism $h\colon (Pr, \mathcal{F}n, \mathcal{A}, I) \to (Pr', \mathcal{F}n', \mathcal{A}', I')$ is a total S-sorted map $h\colon \mathcal{A} \to \mathcal{A}'$ such that $Pr^{\mathcal{V}}_h(Pr') \subseteq Pr$, $I_{F,s}(\alpha)$ is h-related to $I'_{F,s}(\alpha)$ for all $\alpha \in F_s$, $s \in S$, and $\|I_P(\pi)\| \subseteq \|Pr^{\mathcal{V}}_h(I'_P(\pi))\|$ for all $\pi \in P$.*

Proposition 5. *$(S, \mathcal{V}, P, \mathcal{F})$-models and $(S, \mathcal{V}, P, \mathcal{F})$-model homomorphisms form a category* $\mathrm{Mod}(S, \mathcal{V}, P, \mathcal{F})$.

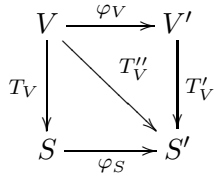
If this notion of homomorphism is too strict, other options include different relations between predicate extensions [8].

5 Model transformation

Now we need to figure out the change of model under signature morphism. Signature morphism has several components. Each of them cause some change of the model. We consider them one-by-one starting with predicate and operation symbols component, then following with change of names and ending with the change of sorts.

The simplest is the change of operation and predicate symbols. It only affects the interpretation functions for operation and predicate symbols. In the new model they become $(I'_P \circ \varphi_P, I'_F \circ \varphi_F)$. Due to properties of φ they are correct interpretations for the set of sorts S.

Consider the following commutative diagram



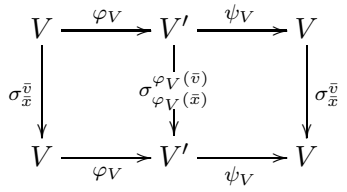
It shows that model transformation is performed sequentially: first, according to the right triangle and then, according to the left triangle.

5.1 Change of names

Recall that name component of signature morphism is a 1-1 S-sorted map $\varphi_V : \varphi_S(\mathcal{V}) \to \mathcal{V}'$. Due to injectivity of φ_V there is a partial map $\psi_V : \mathcal{V}' \nrightarrow \varphi_S(\mathcal{V})$ such that $\varphi_V \circ \psi_V = \mathrm{id}_{\varphi_V(\varphi_S(\mathcal{V}))}$, $\psi_V \circ \varphi_V = \mathrm{id}_{\varphi_S(\mathcal{V})}$. It induces a total function ${}^{\psi_V}\mathcal{A} : {}^{\varphi_S(\mathcal{V})}\mathcal{A} \to {}^{\mathcal{V}'}\mathcal{A}$. Notice that ${}^{\psi_V}\mathcal{A}(d)(v')\uparrow$ for all $v' \in V' \setminus \varphi_V(V)$. There are also total functions $Pr^{\psi_V}_{\mathcal{A}} : Pr^{\mathcal{V}'}_{\mathcal{A}} \to Pr^{\varphi_S(\mathcal{V})}_{\mathcal{A}}$, $Fn^{\psi_{\mathcal{V}}}_{\mathcal{A},s'} : Fn^{\mathcal{V}'}_{\mathcal{A},s'} \to Fn^{\varphi_S(\mathcal{V})}_{\mathcal{A},s'}$. They are required to be able to

jump from $\mathcal{V}'$-quasiary predicate model to $\mathcal{V}$-quasiary predicate model as Mod-functor implies. Before working out change of the model let us see how ψ_V affects the extension of quasiary predicate and how it interacts with compositions.

Lemma 6. *The following diagram commutes in the category of sets and partial mappings*



Proof. Outer rectangle commutes because $\psi_V \circ \varphi_V = \mathrm{id}_V$. Notice that if $v \in V' \setminus \varphi_V(V)$, then value for both paths of right rectangle are undefined since $\psi(v)\uparrow$, $\sigma^{\varphi_V(\bar{v})}_{\varphi_V(\bar{x})}(v) = v$. Therefore right rectangle commutes. Left rectangle commutes because $\operatorname{dom}\psi_V = \varphi_V(V)$ and ψ_V is injective. □

Proposition 7. *Let $\varphi_V \colon \varphi_S(\mathcal{V}) \to \mathcal{V}'$ be a name component of signature morphism. Then $Pr^{\psi_V}_{\mathcal{A}} \colon Pr^{\mathcal{V}'}_{\mathcal{A}} \to Pr^{\varphi_S(\mathcal{V})}_{\mathcal{A}}$, $Fn^{\psi_{\mathcal{V}}}_{\mathcal{A},s'} \colon Fn^{\mathcal{V}'}_{\mathcal{A},s'} \to Fn^{\varphi_S(\mathcal{V})}_{\mathcal{A},s'}$ preserve compositions in the following sense. Let $p', q' \in Pr^{\mathcal{V}'}_{\mathcal{A}}$, $x, v_j, x_j, u_i \in V$, $g' \in Fn^{\mathcal{V}'}_{\mathcal{A},s'}$, $f'_i \in Fn^{\mathcal{V}'}_{\mathcal{A},T''_V(u_i)}$, then*

$$
\begin{aligned}
Pr^{\psi_V}_{\mathcal{A}}(\neg p') &= \neg Pr^{\psi_V}_{\mathcal{A}}(p') \\
Pr^{\psi_V}_{\mathcal{A}}(p' \vee q') &= Pr^{\psi_V}_{\mathcal{A}}(p') \vee Pr^{\psi_V}_{\mathcal{A}}(q') \\
Pr^{\psi_V}_{\mathcal{A}}(R^{\varphi_V(\bar{v})}_{\varphi_V(\bar{x})} p') &= R^{\bar{v}}_{\bar{x}} Pr^{\psi_V}_{\mathcal{A}}(p') \\
Pr^{\psi_V}_{\mathcal{A}}(\exists \varphi_V(x) p') &= \exists x Pr^{\psi_V}_{\mathcal{A}}(p') \\
Fn^{\psi_V}_{\mathcal{A},T''_V(x)}(\text{`}\varphi_V(x)) &= \text{`}x \\
Fn^{\psi_V}_{\mathcal{A},s'}([\,\varphi_V(\bar{u}) \mapsto \bar{f}'\,]\, g') &= [\bar{u} \mapsto Fn^{\psi_V}_{\mathcal{A},T''_V(\bar{u})}(\bar{f}')]\, Fn^{\psi_V}_{\mathcal{A},s'}(g') \\
Pr^{\psi_V}_{\mathcal{A}}([\,\varphi_V(\bar{u}) \mapsto \bar{f}'\,]\, p') &= [\bar{u} \mapsto Fn^{\psi_V}_{\mathcal{A},T''_V(\bar{u})}(\bar{f}')]\, Pr^{\psi_V}_{\mathcal{A}}(p').
\end{aligned}
$$

Proof. The proof is similar to the proof of proposition 3. We directly check the properties and switch to extensions of predicates where needed. Suppose that $p', q' \in Pr^{\mathcal{V}'}_{\mathcal{A}}$. Then

$$\top(Pr^{\psi_V}_{\mathcal{A}}(p')) = \{d \in {}^{\varphi_S(\mathcal{V})}\mathcal{A} \mid {}^{\psi_V}\mathcal{A}(d) \in \top(p')\} = {}^{\psi_V}\mathcal{A}^{-1}(\top(p')),$$

where ${}^{\psi_V}\mathcal{A}^{-1}(D)$ is a preimage of D under map ${}^{\psi_V}\mathcal{A}$. The same goes for the falsity domain. Therefore

$$\|Pr^{\psi_V}_{\mathcal{A}}(p')\| = \left({}^{\psi_V}\mathcal{A}^{-1}(\top(p')), {}^{\psi_V}\mathcal{A}^{-1}(\bot(p'))\right) = {}^{\psi_V}\mathcal{A}^{-1}(\|p'\|).$$

Respectively

$$\begin{aligned}
\|Pr^{\psi_V}_{\mathcal{A}}(\neg p')\| &= {}^{\psi_V}\mathcal{A}^{-1}(\bot(p'), \top(p')) = \|\neg Pr^{\psi_V}_{\mathcal{A}}(p')\|, \\
\|Pr^{\psi_V}_{\mathcal{A}}(p' \vee q')\| &= {}^{\psi_V}\mathcal{A}^{-1}(\top(p') \cup \top(q'), \bot(p') \cap \bot(q')) \\
&= \|Pr^{\psi_V}_{\mathcal{A}}(p') \vee Pr^{\psi_V}_{\mathcal{A}}(q')\|.
\end{aligned}$$

Here we used the properties of the preimage of a function.

For the existential quantifier let $p' \in Pr^{\mathcal{V}'}_{\mathcal{A}}$, then

$$\begin{aligned}
\top(\exists x Pr^{\psi_V}_{\mathcal{A}}(p')) &= \left\{d \mid (d\triangledown x \mapsto a) \circ \psi_V \in \top(p') \text{ for some } a \in A_{T''_V(x)}\right\} \\
&= \left\{d \mid d \circ \psi_V \triangledown \varphi_V(x) \mapsto a \in \top(p') \text{ for some } a \in A_s\right\} \\
&= \top(Pr^{\psi_V}_{\mathcal{A}}(\exists \varphi_V(x) p')).
\end{aligned}$$

Here we used the definition of ψ_V and the following property of nominative sets. For $d \in {}^{\mathcal{V}}\mathcal{A}$, partial function $\sigma\colon \mathcal{V}' \nrightarrow \mathcal{V}$ we have

$$\begin{aligned}
(d\triangledown x \mapsto a) \circ \sigma &= \left(d\big|_{V\setminus\{x\}} \cup [x \mapsto a]\right) \circ \sigma \\
&= \left(d \circ \sigma\big|_{\sigma^{-1}(V)\setminus\sigma^{-1}(\{x\})} \cup \left[x' \mapsto a \mid \sigma(x')\!\downarrow = x\right]\right) \\
&= d \circ \sigma \triangledown \left[x' \mapsto a \mid \sigma(x')\!\downarrow = x\right].
\end{aligned}$$

Repeating for falsity domain and combining we derive

$$\exists x Pr^{\psi_V}_{\mathcal{A}}(p') = Pr^{\psi_V}_{\mathcal{A}}(\exists \varphi_V(x) p').$$

For renomination by lemma 6 we immediately have

$$R^{\bar{v}}_{\bar{x}} Pr^{\psi_V}_{\mathcal{A}}(p') = Pr^{\sigma^{\bar{v}}_{\bar{x}}} \circ Pr^{\psi_V}_{\mathcal{A}}(p') = Pr^{\psi_V}_{\mathcal{A}}(R^{\varphi_V(\bar{v})}_{\varphi_V(\bar{x})} p').$$

Now let us consider denomination. For $s' = T''_V(x)$ we have

$$Fn^{\psi_V}_{\mathcal{A},s'}(`\varphi_V(x))(d) \cong {}^{\psi_V}A(d)_{s'}(\varphi_V(x)) \cong d_{s'}(\psi_V(\varphi_V(x)) \cong `x(d).$$

Let $g' \in Fn^{\mathcal{V}'}_{\mathcal{A},s'}$, $f'_i \in Fn^{\mathcal{V}'}_{\mathcal{A},T''_V(u_i)}$, $u_i \in V$, $i = \overline{1,n}$. Notice that

$$\begin{aligned}([\bar{u} \mapsto Fn^{\psi_V}_{\mathcal{A},T''_V(\bar{u})}(\bar{f}')]\, d) \circ \psi_V &= \left(d\big|_{V\setminus\{u_i\}} \nabla [\bar{u} \mapsto \bar{f}'(d \circ \psi_V)]\right) \circ \psi_V \\ &= [\,\varphi_V(\bar{u}) \mapsto \bar{f}'\,]\,(d \circ \psi_V).\end{aligned}$$

Here we used equality

$$[\bar{u} \mapsto \bar{f}'(d')] \circ \psi_V = [v \mapsto f'_i(d') \mid \psi_V(v)\downarrow = u_i] = [\varphi_V(\bar{u}) \mapsto \bar{f}'(d')].$$

Then

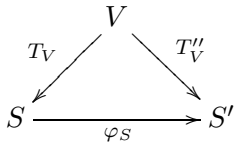
$$\begin{aligned}Fn^{\psi_V}_{\mathcal{A},s'}([\,\varphi_V(\bar{u}) \mapsto \bar{f}'\,]\, g')(d) &\cong [\,\varphi_V(\bar{u}) \mapsto \bar{f}'\,]\, g'(d \circ \psi_V) \\ &\cong g'([\,\varphi_V(\bar{u}) \mapsto \bar{f}'\,]\, d \circ \psi_V) \\ &\cong g'(([\bar{u} \mapsto Fn^{\psi_V}_{\mathcal{A},T''_V(\bar{u})}(\bar{f}')]\, d) \circ \psi_V) \\ &\cong [\bar{u} \mapsto Fn^{\psi_V}_{\mathcal{A},T''_V(\bar{u})}(\bar{f}')]\, Fn^{\psi_V}_{\mathcal{A},s'}(g')(d).\end{aligned}$$

And similarly

$$Pr^{\psi_V}_{\mathcal{A}}([\,\varphi_V(\bar{u}) \mapsto \bar{f}'\,]\, p')(d) \cong [\bar{u} \mapsto Fn^{\psi_V}_{\mathcal{A},T''_V(\bar{u})}(\bar{f}')]\, Pr^{\psi_V}_{\mathcal{A}}(p')(d). \quad \square$$

5.2 Change of base

We start with commutative triangle



According to it $(V, T''_V) = \varphi(\mathcal{V})$. A new S-sorted set of values is obtained from S'-sorted $\mathcal{A}$ using pullback functor $\varphi_S^*(\mathcal{A}) = (A_{\varphi_S(s)})_{s\in S}$. Fundamental for the construction of quasiary predicates and operations of new model is to understand the connection between nominative data of the two models. Here we use 1-1 correspondence

$$\frac{d\colon \mathcal{V} \twoheadrightarrow \varphi_S^*(\mathcal{A})}{d^{\#}\colon \varphi_S(\mathcal{V}) \twoheadrightarrow \mathcal{A}}$$

realized as follows:

$$d^{\#}_{s'} = \bigcup_{\varphi_S(s)=s'}^{\cdot} d_s$$
$$d_s = d^{\#}_{\varphi_S(s)}\Big|_{V_s}.$$

It is actually a conventional adjunction $\varphi_S \dashv \varphi_S^*$ but extended to partial S-sorted maps [13, sec. 9.7].

Let $p \in Pr_{\mathcal{A}}^{\varphi_S(\mathcal{V})}$, $s \in S$, $f \in Fn_{\mathcal{A},\varphi_S(s)}^{\varphi_S(\mathcal{V})}$. Then there are $p^{\#} \in Pr_{\varphi_S^*(\mathcal{A})}^{\mathcal{V}}$, $f^{\#} \in Fn_{\varphi_S^*(\mathcal{A}),s}^{\mathcal{V}}$ defined as

$$p^{\#}(d) \cong p(d^{\#})$$
$$f^{\#}(d) \cong f(d^{\#}).$$

Notice there is an instance of $f^{\#}$ for each s' such that $\varphi_S(s') = \varphi_S(s)$.

By construction, the transition $d \mapsto d^{\#}$ preserves the operation of domain restriction and disjoint union of nominative sets. Respectively

$$(d_1 \triangledown d_2)^{\#} = d_1^{\#} \triangledown d_2^{\#}$$
$$(d \triangledown x \mapsto a)^{\#} = d^{\#} \triangledown x \mapsto a$$
$$(d \triangledown [\bar{v} \mapsto \bar{a}])^{\#} = d^{\#} \triangledown [\bar{v} \mapsto \bar{a}]$$
$$[\,\bar{u} \mapsto \bar{f}^{\#}(d)\,]^{\#} = [\,\bar{u} \mapsto \bar{f}(d^{\#})\,].$$

Proposition 8. *The correspondence $p \mapsto p^\#$, $(f,s) \mapsto f^\#$ preserves compositions in the following sense:*

$$\begin{aligned}
(p \vee q)^\# &= p^\# \vee q^\# \\
(\neg p)^\# &= \neg p^\# \\
(R^{\bar v}_{\bar x} p)^\# &= R^{\bar v}_{\bar x} p^\# \\
(\exists x p)^\# &= \exists x p^\# \\
(\text{`}x)^\# &= \text{`}x \\
([\,\bar u \mapsto \bar f\,]\, g)^\# &= [\bar u \mapsto \bar f^\#]\, g^\# \\
([\,\bar u \mapsto \bar f\,]\, p)^\# &= [\bar u \mapsto \bar f^\#]\, p^\#.
\end{aligned}$$

Proof. We immediately check

$$\begin{aligned}
\|(p \vee q)^\#\| &= (\{d \mid d^\# \in \top(p \vee q)\}, \{d \mid d^\# \in \bot(p \vee q)\}) \\
&= (\top(p^\#) \cup \top(q^\#), \bot(p^\#) \cap \bot(q^\#)) = \|p^\# \vee q^\#\|. \\
\|(\neg q)^\#\| &= (\{d \mid d^\# \in \bot(q)\}, \{d \mid d^\# \in \top(q)\}) = \|\neg q^\#\|. \\
\top((\exists x p)^\#) &= \{d \mid d^\# \triangledown x \mapsto a \in \top(p) \text{ for some } a \in A_{T''_V(x)}\} \\
&= \{d \mid (d \triangledown x \mapsto a)^\# \in \top(p) \text{ for some } a \in A_{\varphi_P(T_V(x))}\} \\
&= \top(\exists x p^\#). \\
\bot((\exists x p)^\#) &= \{d \mid (d \triangledown x \mapsto a)^\# \in \bot(p) \text{ for all } a \in A_{\varphi_P(T_V(x))}\} \\
&= \bot(\exists x p^\#). \\
[\,\bar u \mapsto \bar f\,]\, d^\# &= d^\# \triangledown [\,\bar u \mapsto \bar f(d^\#)\,] \\
&= (d \triangledown [\,\bar u \mapsto \bar f^\#(d)\,])^\# = ([\,\bar u \mapsto \bar f^\#\,]\, d)^\#. \\
([\,\bar u \mapsto \bar f\,]\, p)^\#(d) &\cong p([\,\bar u \mapsto \bar f\,]\, d^\#) \\
&\cong p(([\,\bar u \mapsto \bar f^\#\,]\, d)^\#) \cong [\bar u \mapsto \bar f^\#]\, p^\#(d). \\
([\,\bar u \mapsto \bar f\,]\, g)^\#(d) &\cong g([\,\bar u \mapsto \bar f\,]\, d^\#) \\
&\cong g(([\,\bar u \mapsto \bar f^\#\,]\, d)^\#) \cong [\bar u \mapsto \bar f^\#]\, g^\#(d). \\
(\text{`}x)^\#(d) &\cong d^\#_{T''_V(x)}(x) \cong \bigdot\!\bigcup_{\varphi_S(s) = \varphi_S(T_V(x))} d_s(x) \cong \text{`}x(d).
\end{aligned}$$

$$(R^{\bar v}_{\bar x} p)^{\#}(d) \cong p(d^{\#} \circ \sigma^{\bar v}_{\bar x}) \cong p((d \circ \sigma^{\bar v}_{\bar x})^{\#}) \cong R^{\bar v}_{\bar x} p^{\#}(d). \qquad \square$$

Suppose $Pr \subseteq Pr^{\varphi_S(\mathcal{V})}_{\mathcal{A}}$, then we denote $Pr^{\#} = \{p^{\#} \mid p \in Pr\}$.

5.3 Reduct functor

Next we provide combined model transformation.

Proposition 9. *Let $\varphi\colon (S, \mathcal{V}, P, (F_s)_{s\in S}) \to (S', \mathcal{V}', P', \mathcal{F}')$ be a signature morphism and $M' = (Pr', (Fn'_{s'})_{s'\in S'}, \mathcal{A}, I')$ be a $(S', \mathcal{V}', P', \mathcal{F}')$-model. Then there is a $(S, \mathcal{V}, P, (F_s)_{s\in S})$-model*

$$Mod(\varphi)(M') = (Pr, (Fn_s)_{s\in S}, \varphi^*_S(\mathcal{A}), (I_P, I_F)), \tag{7}$$

where

$$\begin{aligned} Pr &= Pr^{\psi_V}_{\mathcal{A}}(Pr')^{\#}, \quad Fn_s = Fn^{\psi_V}_{\mathcal{A},\varphi_S(s)}(Fn'_{\varphi_S(s)})^{\#}, \\ I_P(\pi) &= Pr^{\psi_V}_{\mathcal{A}}(I'_P(\varphi_P(\pi)))^{\#} \text{ for } \pi \in P, \\ I_{F,s}(\alpha) &= Fn^{\psi_V}_{\mathcal{A},\varphi_S(s)}(I'_{F,\varphi_S(s)}(\varphi_F(\alpha)))^{\#} \text{ for } \alpha \in F_s. \end{aligned}$$

Proof. Considerations above show that Pr is closed under quasiary predicates compositions. For instance, assume that $p_{1,2} = Pr^{\psi_V}_{\mathcal{A}}(p'_{1,2})^{\#} \in Pr$. Then $p_1 \vee p_2 = Pr^{\psi_V}_{\mathcal{A}}(p'_1 \vee p'_2)^{\#} \in Pr$. Class $(Fn_s)_{s\in S}$ is also closed under operation composition. For instance, suppose that $v_i \in V$, $f_i = Fn^{\psi_V}_{\mathcal{A},T''_V(v_i)}(f'_i)^{\#} \in Fn_{T_V(v_i)}$, $i = \overline{1,n}$, $g = Fn^{\psi_V}_{\mathcal{A},\varphi_S(s)}(g')^{\#} \in Fn_s$. Then

$$[\,\bar v \mapsto \bar f\,]\, g = Fn^{\psi_V}_{\mathcal{A},\varphi_S(s)}([\,\varphi_V(\bar v) \mapsto \bar f'\,]\, f')^{\#} \in Fn_s.$$

Thus $(Pr, (Fn_s)_{s\in S}, \varphi^*_S(\mathcal{A}); Comp(\mathcal{V}, \varphi^*_S(\mathcal{A})))$ is indeed a $(\mathcal{V}, \mathcal{A})$-quasiary predicate algebra. The following diagram

$$P \xrightarrow{\varphi_P} P' \xrightarrow{I'_P} Pr' \xrightarrow{Pr^{\psi_V}_{\mathcal{A}}} Pr^{\psi_V}_{\mathcal{A}}(Pr') \xrightarrow{\#} Pr$$

shows that I_P has proper type $P \to Pr$. The same way $I_{F,s}\colon F_s \to Fn_s$. $\square$

Corollary 10. *Given signatures* $\Sigma = (S, \mathcal{V}, P, \mathcal{F})$, $\Sigma' = (S', \mathcal{V}', P', \mathcal{F}')$, *morphism* $\varphi\colon \Sigma \to \Sigma'$ *and* Σ'*-models* $M' = (Pr', \mathcal{F}n', \mathcal{A}, I')$, $M'_1 = (Pr'_1, \mathcal{F}n'_1, \mathcal{A}_1, I'_1)$, *any* Σ'*-model homomorphism* $h\colon M' \to M'_1$ *induces* Σ*-model homomorphism* $\varphi^*_S(h)\colon \mathrm{Mod}(\varphi)(M') \to \mathrm{Mod}(\varphi)(M'_1)$.

Proof. There are three conditions to check here: that map $Pr^{\mathcal{V}}_{\varphi^*_S(h)}$ satisfies image condition, that it preserves predicates from P and that map $\varphi^*_S(h)$ preserves operations from $\mathcal{F}$.

By definition of $(S', \mathcal{V}', P', \mathcal{F}')$-model homomorphism, commutativity (4) and properties of images we have

$$Pr^{\psi_V}_{\mathcal{A}}(Pr')^{\#} \subseteq Pr^{\psi_V}_{\mathcal{A}}(Pr^{\mathcal{V}'}_h(Pr'_1))^{\#} = Pr^{\mathcal{V}}_{\varphi^*_S(h)}(Pr^{\psi_V}_{\mathcal{A}_1}(Pr'_1)^{\#}).$$

Here we also used properties of adjunction:

$$Pr^{\mathcal{V}}_h(p)^{\#}(d) \cong p(h \circ d^{\#}) \cong p((\varphi^*_S(h) \circ d)^{\#}) \cong Pr^{\mathcal{V}}_{\varphi^*_S(h)}(p^{\#})(d).$$

Similarly, for any $\pi \in P$ we have

$$\|Pr^{\psi_V}_{\mathcal{A}}(I'_P(\pi'))^{\#}\| \subseteq \|Pr^{\mathcal{V}}_{\varphi^*_S(h)}(Pr^{\psi_V}_{\mathcal{A}_1}(I'_{1P}(\pi'))^{\#})\|,$$

where $\pi' = \varphi_P(\pi)$.

Finally, for any $\alpha \in F_s$ we have

$$\begin{aligned}
\varphi^*_S(h)_s(I_{F,s}(\alpha)(d)) &\cong h_{\varphi_S(s)} \circ I'_{F,\varphi_S(s)}(\alpha') \circ {}^{\psi_V}\mathcal{A}(d^{\#}) \\
&\cong I'_{1F,\varphi_S(s)}(\alpha') \circ {}^{\mathcal{V}'}h \circ {}^{\psi_V}\mathcal{A}(d^{\#}) \\
&\cong I'_{1F,\varphi_S(s)}(\alpha') \circ {}^{\psi_V}\mathcal{A}_1(h \circ d^{\#}) \\
&\cong I'_{1F,\varphi_S(s)}(\alpha') \circ {}^{\psi_V}\mathcal{A}_1((\varphi^*_S(h) \circ d)^{\#}) \\
&\cong I_{1F,s}(\alpha) \circ {}^{\mathcal{V}}\varphi^*_S(h)(d).
\end{aligned}$$

where $\alpha' = \varphi_F(\alpha) \in F'_{\varphi_S(s)}$. Here we used preservation of α' by h, adjunction and commutativity for induced maps (3). □

Corollary 11. *The construction given by* (7) *and*

$$\mathrm{Mod}(\varphi)(h) = \varphi^*_S(h)$$

extends Mod *to a functor* $\mathbb{S}ig \to \mathbb{C}at$.

When Mod is known, the model resulting from the application of the reduct functor $\mathrm{Mod}(\varphi)$ to M, i.e. $\mathrm{Mod}(\varphi)(M)$, is often abbreviated as $M\big|_{\varphi}$.

6 Satisfaction relation

First, we extend the interpretation to all terms and formulas. Due to the definition of quasiary predicate algebras it is quite easy. Let $\Sigma = (S, \mathcal{V}, P, \mathcal{F})$, $t \in \mathrm{Ter}(\Sigma)$, $\Phi \in \mathrm{Sen}(\Sigma)$, $M = (Pr, (Fn_s)_{s\in S}, \mathcal{A}, I) \in |\mathrm{Mod}(\Sigma)|$. We define $M(\Phi) \in Pr$, $M_{T(t)}(t) \in Fn_{T(t)}$ inductively:

$$\begin{aligned}
M_s(\alpha) &= I_{F,s}(\alpha), \text{ where } \alpha \in F_s \\
M_{T_V(x)}(\text{`}x) &= \text{`}x \\
M_{T(t')}([\,\bar v \mapsto \bar t\,]\, t') &= [\bar v \mapsto M_{T(\bar t)}(\bar t)]\, M_{T(t')}(t') \\
M(\pi) &= I_P(\pi) \\
M(\Phi \vee \Psi) &= M(\Phi) \vee M(\Psi) \\
M(\neg\Phi) &= \neg M(\Phi) \\
M(\exists x \Phi) &= \exists x M(\Phi) \\
M([\,\bar v \mapsto \bar t\,]\, \Phi) &= [\bar v \mapsto M_{T(\bar t)}(\bar t)]\, M(\Phi).
\end{aligned}$$

In the right-hand side we use the interpretation of composition symbols given in subsection 4.3. The interpretation is respected by homomorphisms in the following sense.

Proposition 12. *Let $h\colon M \to M_1$ be a Σ-model homomorphism, where $M = (Pr, \mathcal{F}n, \mathcal{A}, I)$, $M_1 = (Pr_1, \mathcal{F}n_1, \mathcal{A}_1, I_1)$ are Σ-models. Then $M(t)$ is h-related to $M'(t)$ for any Σ-term t.*

Proof. By induction on term structure and proposition 4. □

Corollary 13. *Let $h\colon M \to M_1$ be a Σ-model homomorphism, where $M = (Pr, \mathcal{F}n, \mathcal{A}, I)$, $M_1 = (Pr_1, \mathcal{F}n_1, \mathcal{A}_1, I_1)$. Then for arbitrary Σ-terms t_i, $i = \overline{1,n}$ and predicate $p_1 \in Pr_1$, the following holds*

$$Pr_h^{\mathcal{V}}([\bar v \mapsto M_1(\bar t)]\, p_1) = [\bar v \mapsto M(\bar t)]\, Pr_h^{\mathcal{V}}(p_1).$$

Definition 12. *A formula* $\Phi \in \mathrm{Sen}(\Sigma)$ *is* satisfied *by* Σ*-model* $M = (Pr, \mathcal{F}n, \mathcal{A}, I)$*, if the predicate* $M(\Phi)$ *is irrefutable, i.e.* $\bot(M(\Phi)) = \emptyset$*. This is denoted by* $M \models \Phi$*.*

Let us see how change of notation affects interpretation of a formula.

Proposition 14. *Given a* Σ*-term* t*, formula* $\Phi \in \mathrm{Sen}(\Sigma)$*, signature* $\Sigma' = (S', \mathcal{V}', P', (F'_s)_{s' \in S'})$*,* Σ'*-model* $M' = (Pr', \mathcal{F}n', \mathcal{A}, I')$ *and signature morphism* $\varphi\colon \Sigma \to \Sigma'$*, the following holds:*

$$Fn^{\psi_V}_{\mathcal{A},\varphi_S(s)}(M'_{\varphi_S(s)}(\varphi(t)))^{\#} = M'\big|_{\varphi,s}(t), \text{ where } s = T(t)$$
$$Pr^{\psi_V}_{\mathcal{A}}(M'(\varphi(\Phi)))^{\#} = M'\big|_{\varphi}(\Phi).$$

Proof. By induction on structure of term t and formula Φ respectively. Let $\alpha \in F_s$ be an operation symbol. Then

$$\begin{aligned} Fn^{\psi_V}_{\mathcal{A},\varphi_S(s)}(M'_{\varphi_S(s)}(\varphi_F(\alpha)))^{\#} &= Fn^{\psi_V}_{\mathcal{A},\varphi_S(s)}(I'_{F,\varphi_S(s)}(\varphi_F(\alpha)))^{\#} \\ &= M'\big|_{\varphi,s}\varphi_F(\alpha). \\ Fn^{\psi_V}_{\mathcal{A},\varphi_S(T_V(x))}(M'_{\varphi_S(T_V(x))}(\text{`}\varphi_V(x)))^{\#} &= Fn^{\psi_V}_{\mathcal{A},\varphi_S(T_V(x))}(\text{`}\varphi_V(x))^{\#} \\ &= \text{`}x^{\#} = M'\big|_{\varphi,T_V(x)}(\text{`}x). \end{aligned}$$

In the latter we used propositions 7 and 8. For the next case in addition to them we use induction hypothesis. Let t, t_i be Σ-terms, $v_i \in V$, $T_V(v_i) = T(t_i) = s_i$, $T(t) = s$, $\varphi_S(s) = s'$, $i = \overline{1,n}$. Then

$$\begin{aligned} Fn^{\psi_V}_{\mathcal{A},s'}(M'(\varphi([\,\bar{v} \mapsto \bar{t}\,]\,t)))^{\#} &= Fn^{\psi_V}_{\mathcal{A},s'}\left([\varphi_V(\bar{v}) \mapsto M'_{\bar{s}}(\varphi(\bar{t}))]\,M'(\varphi(t))\right)^{\#} \\ &= [\bar{v} \mapsto M'\big|_{\varphi,\bar{s}}(\bar{t})]\,M'\big|_{\varphi,s}(t') \\ &= M'\big|_{\varphi,s}([\,\bar{v} \mapsto \bar{t}\,]\,t'). \end{aligned}$$

The same can be done for formulas. Let $\pi \in P$ be a predicate symbol, then

$$Pr^{\psi_V}_{\mathcal{A}}(M'(\varphi_P(\pi)))^{\#} = Pr^{\psi_V}_{\mathcal{A}}(I'_P(\varphi_P(\pi)))^{\#} = M'\big|_{\varphi}(\pi).$$

For the rest of cases we use propositions 7, 8, induction hypothesis and just proven property for terms:

$$\begin{aligned}
Pr^{\psi_V}_{\mathcal{A}}(M'(\varphi(\Phi \vee \Psi)))^{\#} &= Pr^{\psi_V}_{\mathcal{A}}\left(M'(\varphi(\Phi))^{\#} \vee M'(\varphi(\Psi))\right)^{\#} \\
&= Pr^{\psi_V}_{\mathcal{A}}\left(M'(\varphi(\Phi))\right)^{\#} \vee Pr^{\psi_V}_{\mathcal{A}}\left(M'(\varphi(\Psi))\right)^{\#} \\
&= M'\big|_{\varphi}(\Phi) \vee M'\big|_{\varphi}(\Psi) = M'\big|_{\varphi}(\Phi \vee \Psi). \\
Pr^{\psi_V}_{\mathcal{A}}(M'(\varphi(\neg\Phi)))^{\#} &= \neg\left(Pr^{\psi_V}_{\mathcal{A}}(M'(\varphi(\Phi)))^{\#}\right) \\
&= \neg M'\big|_{\varphi}(\Phi) = M'\big|_{\varphi}(\neg\Phi). \\
Pr^{\psi_V}_{\mathcal{A}}(M'(\varphi(\exists x\Phi)))^{\#} &= Pr^{\psi_V}_{\mathcal{A}}(\exists\varphi_V(x)M'(\varphi(\Phi)))^{\#} \\
&= \exists x Pr^{\psi_V}_{\mathcal{A}}(M'(\varphi(\Phi)))^{\#} = M'\big|_{\varphi}(\exists x\Phi).
\end{aligned}$$

The case for substitution in formula basically repeats case for substitution in term. □

Corollary 15. *Given formula $\Phi \in \mathrm{Sen}(\Sigma)$, Σ'-model M' and signature morphism $\varphi\colon \Sigma \to \Sigma'$, then*

$$M' \models \varphi(\Phi) \text{ if and only if } M'\big|_{\varphi} \models \Phi.$$

Proof. By previous proposition $\bot(M'\big|_{\varphi}(\Phi))^{\#} = {}^{\psi_V}\mathcal{A}^{-1}(\bot(M'(\varphi(\Phi))))$, where $\mathcal{A}$ is the carrier of M'. Due to properties of images the satisfaction condition holds. □

By proposition 2, and corollaries 11, 15 we have

Theorem 1. *Constructed* $(\mathbb{S}ig, \mathrm{Sen}, \mathrm{Mod}, \models)$ *form an institution.*

This result finishes construction of institution **FOCNL** for many-sorted first-order composition-nominative logic.

7 Conclusion

This paper proves that many-sorted first-order composition-nominative logic forms an institution. For this all necessary constituents of institution are provided. Homomorphisms between models of many-sorted

first-order CNL are introduced. This construction can be considered as an extension of the institution for (pure) first-order CNL [8], [9]. It can be developed further to accommodate programming logics like [4]. Other line of research suggests studying the distinctive features of obtained institutions compared to more conventional ones.

References

[1] M. Nikitchenko and S. Shkilniak, *Mathematical logic and theory of algorithms*. Kyiv, Ukraine: Publishing house of Taras Shevchenko National University of Kyiv, 2008, (in Ukrainian).

[2] M. Nikitchenko and A. Chentsov, "Basics of intensionalized data: Presets, sets, and nominats," *Computer Science Journal of Moldova*, vol. 20, no. 3(60), pp. 334–365, 2012. [Online]. Available: http://www.math.md/publications/csjm/issues/v20-n3/11122/

[3] M. Nikitchenko and V. Tymofieiev, "Satisfiability in composition-nominative logics," *Central European Journal of Computer Science*, vol. 2, no. 3, pp. 194–213, Oct. 2012. [Online]. Available: http://dx.doi.org/10.2478/s13537-012-0027-3

[4] A. Kryvolap, M. Nikitchenko, and W. Schreiner, "Extending Floyd-Hoare logic for partial pre- and postconditions," in *Information and Communication Technologies in Education, Research, and Industrial Applications*, ser. Communications in Computer and Information Science. Springer, 2013, vol. 412, pp. 355–378. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-03998-5_18

[5] R. Diaconescu, *Institution-independent Model Theory*, ser. Studies in Universal Logic, J.-Y. Béziau, Ed. Birkhäuser Basel, 2008.

[6] D. Sannella and A. Tarlecki, *Foundations of Algebraic Specification and Formal Software Development*, ser. Monographs in Theoretical Computer Science. An EATCS Series. Springer-Verlag Berlin Heidelberg, 2012.

[7] T. Mossakowski, J. Goguen, R. Diaconescu, and A. Tarlecki, "What is a logic?" in *Logica Universalis: Towards a General*

Theory of Logic, J.-Y. Béziau, Ed. Birkhäuser Basel, 2007, pp. 111–133.

[8] A. Chentsov and M. Nikitchenko, "Institution for pure first-order composition-nominative logic," in *Proc. Workshop Foundations of Informatics (FOI-2015)*, Aug. 2015, pp. 50–63. [Online]. Available: http://foi.math.md/proceedings.pdf

[9] ——, "Composition-nominative logics as institutions," in *Handbook 5th World Congr. and School Universal Logic (UniLog 2015)*, Jun. 2015, pp. 370–371. [Online]. Available: http://www.uni-log.org/hunilog2015.pdf

[10] C. McLarty, *Elementary Categories, Elementary Toposes*, ser. Oxford Logic Guides, Book 21. Clarendon Press, 1992, ch. 11, pp. 99–106.

[11] J. Stell, "A framework for order-sorted algebra," in *Algebraic Methodology and Software Technology*, ser. Lecture Notes in Computer Science, 2002, vol. 2422, pp. 396–410. [Online]. Available: http://dx.doi.org/10.1007/3-540-45719-4_27

[12] W. Phoa, "An introduction to fibrations, topos theory, the effective topos and modest sets," Lab. Found. Comp. Sci., Univ. of Edinburgh, Edinburgh, Tech. Rep. ECS-LFCS-92-208, 1992.

[13] S. Awodey, *Category Theory*, ser. Oxford Logic Guides, Book 52. Oxford University Press, 2006.

[14] B. Pierce, *Types and Programming Languages*. Massachusetts: MIT Press, 2002.

[15] D. Rydeheard and R. Burstall, *Computational Category Theory*. New York, London: Prentice Hall, 1988.

Alexey Chentsov Received December 14, 2015

Alexey Chentsov
Taras Shevchenko National University of Kyiv
01601, Kyiv, Volodymyrska st, 60
Phone: +38044 2590511
E–mail: chentsov@ukr.net

Computer Science Journal of Moldova, vol.**24**, no.1(70), 2016

Some applications of quasigroups in cryptology

N.A. Moldovyan A.V. Shcherbacov V.A. Shcherbacov

Abstract

In the paper we present based on quasigroups new deniable encryption method, generalisation of Markovski stream cipher, and generalisation of El-Gamal enciphering system.

Keywords: cryptology, quasigroup, algorithm, stream, deniable, encryption, El-Gamal scheme.

MSC 2000: 94A60, 20N05, 11S05

1 Deniable-encryption mode for block ciphers

This paper is the extended version of the paper [1].

Deniable encryption (DE) is a method for generating ciphertexts that can be alternatively decrypted providing security against so called coercive attacks [2] for which it is assumed that after ciphertext has been sent the adversary has possibility to force both the sender and the receiver to open the plaintext corresponding to the ciphertext and the encryption key. In the case of block ciphering the DE can be provided with simultaneous encryption of the secret and fake messages using the secret and fake keys, correspondingly. While being coerced the sender and receiver of the ciphertext open the fake key and fake message and declare they have used the probabilistic encryption [3]. Earlier in paper [4] it had been proposed a method for simultaneous encryption of two messages based on solving a system of two linear equations. In this section we propose design of the DE mode for using block ciphers, which is based on the mentioned method.

Definition 1. *Binary groupoid* $(G, \circ)$ *is an isotopic image of a binary groupoid* $(G, \cdot)$*, if there exist permutations* α, β, γ *of the set* G *such that* $x \circ y = \gamma^{-1}(\alpha x \cdot \beta y)$ *[5].*

Suppose E_V be a block encryption algorithm with n-bit input data block and the value used as encryption key. All existing n-bit data blocks can be considered as elements of some quasigroup with the operation $*$ defined as follows:

$$K * i = E_V(K \oplus E_V(i)),$$

where $\oplus$ is the XOR operation; K and i are n-bit vectors. This quasigroup is an isotope of the group $(G, \oplus)$, where G is the set of all n-bit vectors. Here E_V is a permutation of the symmetric group S_G.

Evidently, for all possible values i and $Q \neq K$ we have

$$E_V(Q \oplus E_V(i)) \neq E_V(K \oplus E_V(i)). \tag{1}$$

Using this property of the quasigroup and two different keys K and $Q \neq K$ one can define simultaneous encryption of two different messages $T = (t_1, t_2, \ldots, t_i, \ldots, t_z)$ and $M = (m_1, m_2, \ldots, m_i, \ldots, m_z)$, where $z < 2^n$; t_i and m_i are n-bit data blocks, as generation of the single ciphertext $C = (c_1, c_2, \ldots, c_i, \ldots, c_z)$ containing $(2n)$-bit ciphertext blocks $c_i = (c_i', c_i'')$, where c_i' and c_i'' are n-bit values, computed from the following system of equations in the field $GF(2^n)$:

$$\begin{cases} c_i' + A_i c_i'' \equiv B_i + m_i \bmod \eta(x) \\ c_i' + G_i c_i'' \equiv H_i + t_i \bmod \eta(x), \end{cases} \tag{2}$$

where $\eta(x)$ is some specified irreducible binary polynomial of the degree n; the n-bit values A_i, B_i, G_i, and H_i are computed using the random n-bit initialization vector V (this value is not secret) as follows:

$$\begin{aligned} &A_i = E_V(K \oplus E_V(i)); G_i = E_V(Q \oplus E_V(i)); \\ &B_i = E_K(A_i); H_i = E_Q(G_i). \end{aligned} \tag{3}$$

While solving (2) the values A_i, B_i, G_i, and H_i are considered as binary polynomials of the degree $s < n$. Due to condition (1) it holds the unequality $A_i \neq G_i$, therefore the system (2) always has the single solution, i.e. the proposed deniable-encryption procedure is defined correctly.

Let us agree that the secret message (key) is the value T (Q) and the fake message (key) is the value M (K). If the coercer forces the sender and receiver of the secret message T to open the ciphertext C and the encryption key, then they open the fake key K and the fake message M and declare using the probabilistic block-encryption mode implemented with the block cipher E. In terms of paper [3] the declared encryption algorithm is called the associated encryption algorithm.

In the case of the proposed deniable-encryption method the last algorithm is described as consecutive probabilistic encryption of the data blocks m_i for each value $i = 1, 2, \ldots, z$ performing the following steps:

1. Generate a random initialization vector V and compute the values $A_i = E_V(K \oplus E_V(i))$ and $B_i = E_K(A_i)$.

2. Generate a random binary polynomial $\rho_i(x) \neq A_i$ of the degree $s < n$.

3. Compute the unknowns c'_i and c''_i from the following system of equations in $GF(2^n)$:

$$\begin{cases} c'_i + A_i c''_i \equiv B_i + m_i \bmod \eta(x) \\ c'_i + \rho_i c''_i \equiv 1 \bmod \eta(x), \end{cases} \tag{4}$$

Evidently, for some sequence of the values $\rho_1(x), \rho_2(x), \ldots, \rho_z(x)$ the message M is transformed with the key K into the given ciphertext C.

To distinguish the use of the deniable encryption with the system (2) from the probabilistic encryption with the system (4) the potential coercive attacker should compute the key Q. The last problem is computationally difficult, if E is a secure block cipher, for example, AES [6] with 128-bit key and $n = 128$. Restoring the secret message from the ciphertext is performed as decryption of each ciphertext block $c_i = (c'_i, c''_i)$, $i = 1, 2, \ldots, z$, as follows:

1. Using the secret key Q compute the values $G_i = E_V(Q \oplus E_V(i))$ and $H_i = E_Q(G_i)$.

2. Compute the plaintext data block $t_i = c'_i + G_i c''_i - H_i \bmod \eta(x)$.

The fake decryption of the ciphertext is performed as follows ($i = 1, 2, \ldots, z$):

1. Using the fake key K compute the values $A_i = E_V(K \oplus E_V(i))$ and $B_i = E_K(A_i)$.

2. Compute the plaintext data block $m_i = c'_i + A_i c''_i - B_i \bmod \eta(x)$.

2 Stream deniable-encryption scheme

The method described in Section 1 can be used for constructing stream deniable encryption algorithms. Formally, for small values n (for example $n = 8$) that method represents consecutive deniable encryption of the pairs of symbols of two different input texts. The sequence of the pairs (A_i, B_i) represents the key stream used for encrypting the message M. Respectively, the sequence of the pairs (G_i, H_i) represents the key stream used for encrypting the message T. However one should take into account that for small values n these key streams contain too short periods, therefore such encryption method is not secure. To overcome this problem one can propose the following modification of the formulas for computing the values A_i, B_i, G_i, and H_i:

$$\begin{aligned} &A_i = E_{V||i}(k_j), G_i = E_{V||i}(q_j), \\ &B_i = E_K(A_i), H_i = E_Q(G_i), \end{aligned} \tag{5}$$

where $\|$ denotes concatenation operation; V is 64-bit initialization vector; i is 64-bit counter; k_j are 8-bit subkeys of the key $K = (k_0, k_1, \ldots, k_7)$; q_j are 8-bit subkeys of the key $Q = (q_0, q_1, \ldots, q_7)$, $j = (i-1) \bmod 8$. In the last formulas the block encryption function E operates with 8-bit data block and 128-bit key.

While generating the keys K and Q it is to be fulfilled the condition $k_j \neq q_j$ for $j = 0, 1, \ldots, 7$.

It should be mentioned that the value $V||i$ is used as variable key for computing the values A_i and B_i, i.e. the key is to be reset for each transformed pair of the 8-bit symbols of the texts M and T. This feature makes preferable to use block ciphers E with simple key scheduling in order to get higher performance of the stream deniable encryption [7]. In the case of hardware implementation of the function

E one can use, for example, the design based on controlled substitution-permutation networks [8].

3 Stream cipher based on binary quasigroups

Here we give more detailed description of algorithm which was proposed in [9]. This algorithm simultaneously uses two cryptographical procedures: enciphering using generalisation of Markovski stream algorithm [10] and enciphering using a system of orthogonal operations.

We also give some realisation of this algorithm based on T-quasigroups, more precise, on medial quasigroups. Necessary information about quasigroups and some its applications in cryptography can be found in [5], [9], [11].

Below we denote the action of the left (right, middle) translation in the power a of a binary quasigroup (Q, g_1) on the element u_1 by the symbol ${}_{g_1}T^a_{l_1}(u_1)$. And so on. Here l_1 means leader element. See [9]–[11] for details.

Algorithm 1. *Enciphering. Initially we have plaintext* $u_1, u_2, \ldots, u_6$.

$$
\begin{aligned}
&\textit{Step } 1.\\
&{}_{g_1}T^a_{l_1}(u_1) = v_1\\
&{}_{g_2}T^b_{l_2}(u_2) = v_2\\
&F^c_1(v_1, v_2) = (v'_1, v'_2)\\
&\textit{Step } 2.\\
&{}_{g_3}T^d_{v'_1}(u_3) = v_3\\
&{}_{g_4}T^e_{v'_2}(u_4) = v_4\\
&F^f_2(v_3, v_4) = (v'_3, v'_4)\\
&\textit{Step } 3.\\
&{}_{g_5}T^g_{v'_3}(u_5) = v_5\\
&{}_{g_6}T^h_{v'_4}(u_6) = v_6\\
&F^i_3(v_5, v_6) = (v'_5, v'_6).
\end{aligned}
\tag{6}
$$

We obtain ciphertext $v'_1, v'_2, \ldots, v'_6$.

Deciphering. Initially we have ciphertext $v'_1, v'_2, \ldots, v'_6$.

$$
\begin{aligned}
&\textit{Step } 1.\\
&F_1^{-c}(v'_1, v'_2) = (v_1, v_2)\\
&{}_{g_1}T_{l_1}^{-a}(v_1) = u_1\\
&{}_{g_2}T_{l_2}^{-b}(v_2) = u_2\\
&\textit{Step } 2.\\
&F_2^{-f}(v'_3, v'_4) = (v_3, v_4)\\
&{}_{g_3}T_{v'_1}^{-d}(v_3) = u_3\\
&{}_{g_4}T_{v'_2}^{-e}(v_4) = u_4\\
&\textit{Step } 3.\\
&F_3^{-i}(v'_5, v'_6) = (v_5, v_6)\\
&{}_{g_5}T_{v'_3}^{-g}(v_5) = u_5\\
&{}_{g_6}T_{v'_4}^{-h}(v_6) = u_6
\end{aligned}
\tag{7}
$$

We obtain plaintext $u_1, u_2, \ldots, u_6$.

From Algorithm 1 we obtain classical Markovski algorithm, if we take only one quasigroup, one kind of quasigroup translations (left translations) any of which is taken in power = 1, and, finally, if system of orthogonal operations (crypto-procedure F) is not used. Some generalisations of Algorithm 1 are given in [12].

4 T-quasigroup based stream cipher

We give a numerical example of encryption Algorithm 1 based on T-quasigroups (more exactly, on medial quasigroups) [12]. Notice that the number 257 is prime. The form of parastrophes of T-quasigroups, for example, of quasigroup $(A, \overset{(13)}{*})$ can be found in [12], [13, p. 39].

Example 1. *Take the cyclic group* $(Z_{257}, +) = (A, +)$.

1. *Define T-quasigroup* $(A, *)$ *by the form* $x*y = 2\cdot x+131\cdot y+3$ *with a leader element* l*, say,* $l = 17$*. Denote the mapping* $x \mapsto x * l$ *by the letter* g_1*, i.e.* $g_1(x) = x * l$ *for all* $x \in A$*.*

   In order to find the mapping g_1^{-1} *we find the form of operation* $\overset{(13)}{*}$*. We have* $x \overset{(13)}{*} y = 129\cdot x+63\cdot y+127$*,* $f^{-1}x = x \overset{(13)}{*} l$*. Then* $g_1^{-1}(g_1(x)) = g_1^{-1}(x * l) = (x * l) \overset{(13)}{*} l{=}x$*.*

   In some sense quasigroup $(A, \overset{(13)}{*})$ *is the "right inverse quasigroup" to quasigroup* $(A, *)$*. Notice that from results of article [13, Theorem 16] it follows that* $(A, *)\perp(A, \overset{(13)}{*})$*.*

2. *Define T-quasigroup* $(A, \circ)$ *by the form* $x \circ y = 10 \cdot x + 81 \cdot y + 53$ *with a leader element* l*, say,* $l = 71$*. Denote the mapping* $x \mapsto l{*}x$ *by the letter* g_2*, i.e.* $g_2(x) = l \circ x$ *for all* $x \in A$*.*

   In order to find the mapping g_2^{-1} *we find the form of operation* $\overset{(23)}{\circ}$*. We have* $x \overset{(23)}{\circ} y = 149 \cdot x + 165 \cdot y + 250$*.*

3. *Define a system of two parastroph orthogonal T-quasigroups* $(A, \cdot)$ *and* $(A, \overset{(23)}{\cdot})$ *in the following way*

$$\begin{cases} x \cdot y = 3 \cdot x + 5 \cdot y + 6 \\ x \overset{(23)}{\cdot} y = 205 \cdot x + 103 \cdot y + 153. \end{cases}$$

   Denote quasigroup system $(A, \cdot, \overset{(23)}{\cdot})$ *by* $F(x, y)$*, since this system is a function of two variables.*

   In order to find the mapping $F^{-1}(x, y)$ *we solve the system of linear equations*

$$\begin{cases} 3 \cdot x + 5 \cdot y + 6 = a \\ 205 \cdot x + 103 \cdot y + 153 = b. \end{cases}$$

   We have $\Delta = 55$*,* $1/\Delta = 243$*,* $x = 100\cdot a+70\cdot b+255$*,* $y = 43\cdot a+215 \cdot b$*. Therefore we have, if* $F(x, y) = (a, b)$*, then* $F^{-1}(a, b) =$

$(100 \cdot a + 70 \cdot b + 255, 43 \cdot a + 215 \cdot b)$, *i.e.*

$$\begin{cases} x = 100 \cdot a + 70 \cdot b + 255 \\ y = 43 \cdot a + 215 \cdot b. \end{cases}$$

We have defined the mappings g_1, g_2, F *and now we can use them in Algorithm 1.*

Let 212; 17; 65; 117 *be a plaintext. We take the following values in formula (6):* $a = b = d = e = f = 1; c = 2$. *Below we use Gothic font to distinguish leader elements, i.e., the numbers* $\mathfrak{17}$ *and* $\mathfrak{71}$ *are leader elements. Then*

Step 1.

$g_1(212) = 212 * \mathfrak{17} = 2 \cdot 212 + 131 \cdot 17 + 3 = 84$

$g_2(17) = \mathfrak{71} \circ 17 = 10 \cdot 71 + 81 \cdot 17 + 53 = 84$

$F(84; 84) = (3 \cdot 84 + 5 \cdot 84 + 6; 205 \cdot 84 + 103 \cdot 84 + 153) = (164; 68)$

$F(164; 68) = (3 \cdot 164 + 5 \cdot 68 + 6; 205 \cdot 164 + 103 \cdot 68 + 153) =$ ***(67; 171)***

Step 2.

$g_1(65) = 65 * 67 = 2 \cdot 65 + 131 \cdot 67 + 3 = 172$

$g_2(117) = 171 \circ 117 = 10 \cdot 171 + 81 \cdot 117 + 53 = 189$

$F(172; 189) = (3 \cdot 172 + 5 \cdot 189 + 6; 205 \cdot 172 + 103 \cdot 189 + 153) =$ ***(182; 139)***

We obtain the following ciphertext 67; 171; 182; 139.

For deciphering we use formula (7).

Step 1.

$F^{-1}(67; 171) = (100 \cdot 67 + 70 \cdot 171 + 255, 43 \cdot 67 + 215 \cdot 171) = (164; 68)$

$F^{-1}(164; 68) = (100 \cdot 164 + 70 \cdot 68 + 255, 43 \cdot 164 + 215 \cdot 68) = (84; 84)$

$g_1^{-1}(84) = 84 \overset{(13)}{*} 17 = 129 \cdot 84 + 63 \cdot 17 + 127 =$ ***212***

$g_2^{-1}(84) = 71 \overset{(23)}{\circ} 84 = 149 \cdot 71 + 165 \cdot 84 + 250 =$ ***17***

Step 2.

$F^{-1}(182; 139) = (100 \cdot 182 + 70 \cdot 139 + 255, 43 \cdot 182 + 215 \cdot 139) = (172; 189)$

$g_1^{-1}(172) = 172 \overset{(13)}{*} 67 = 129 \cdot 172 + 63 \cdot 67 + 127 =$ ***65***

$$g_2^{-1}(189) = 171 \overset{(23)}{\circ} 189 = 149 \cdot 171 + 165 \cdot 189 + 250 = \boldsymbol{117}$$

A program using freeware version of programming language Pascal was developed. First experiments demonstrate that encoding-decoding is executed sufficiently fast.

We plan to continue researches in this direction, namely we plan to estimate time complexity of encryption and decryption for the proposed algorithms, as well as give formal justification for computational security of ones. The authors thank Referee for this suggestion.

Remark 1. *Proper binary groupoids are more preferable than linear quasigroups by construction of the mapping $F(x, y)$ in order to make encryption more safe, but in this case decryption may be slower than in linear quasigroup case and definition of these groupoids needs more computer (or some other device) memory. The same remark is true for the choice of the function g. Maybe a golden mean in this choice problem is to use linear quasigroups over non-abelian, especially simple, groups.*

Remark 2. *In this cipher there exists a possibility of protection against standard statistical attack. For this scope it is possible to denote more often used letters or pair of letters by more than one integer or by more than one pair of integers.*

5 El Gamal cryptosystem

We recall El Gamal cryptosystem [14]. Let $(Z_p, +)$ be a cyclic group of residues of big (say 200 to 300 digits) prime order relative to addition of residues, a be a generator of the group $(Z_{p-1}, \cdot) \cong Aut(Z_p, +)$ $(gcd(a, p-1) = 1)$.

Alices keys are as follows:

Public Key p, a, and a^m, $m \in \mathbb{N}$.

Private Key m.

Encryption

To send a message $b \in (Z_{p-1}, \cdot)$ Bob computes a^r and a^{mr} for a random $r \in \mathbb{N}$ (sometimes the number r is called an ephemeral key).

The ciphertext is $(a^r, a^{mr} \cdot b)$.

Decryption

Alice knows m, so if she receives the ciphertext $(a^r, a^{mr} \cdot b)$, she computes a^{-rm} from a^r and then she computes b from $a^{mr} \cdot b$ using the formula $a^{-mr} \cdot a^{mr} \cdot b = b$.

6 De-symmetrisation of Markovski algorithm

We give an analogue of El Gamal encryption system based on Markovski algorithm [1], [15].

Let (Q, f) be a binary quasigroup and $T = (\alpha, \beta, \gamma)$ be its isotopy.

Alices keys are as follows:

Public Key is (Q, f), T, $T^{(m,n,k)} = (\alpha^m, \beta^n, \gamma^k)$, $m, n, k \in \mathbb{N}$, and Markovski algorithm.

Private Key m, n, k.

Encryption

To send a message $b \in (Q, f)$ Bob computes $T^{(r,s,t)}$, $T^{(mr,ns,kt)}$ for a random $r, s, t \in \mathbb{N}$ and $(T^{(mr,ns,kt)}(Q, f))$.

The ciphertext is $(T^{(r,s,t)}, (T^{(mr,ns,kt)}(Q, f))b)$.

To obtain $(T^{(mr,ns,kt)}(Q, f))b$ Bob uses Markovski algorithm which is known to Alice.

Decryption

Alice knows m, n, k, so if she receives the ciphertext

$$(T^{(r,s,t)}, (T^{(mr,ns,kt)}(Q, f))b),$$

she computes $(T^{(mr,ns,kt)}(Q, f))^{-1}$ using $T^{(r,s,t)}$ and, finally, she computes b.

In this algorithm there can also be used isostrophy [16] instead of isotopy, Algorithm 1 instead of Markovski algorithm, n-ary ($n > 2$) quasigroups [9], [17] instead of binary quasigroups.

Generalisation of El Gamal scheme using a Moufang loop is given in [18]. In [19] it is proved that discrete logarithms problem in Moufang loops can be reduced to the same problem in finite simple fields.

Generalisation of El Gamal encryption system based on a quasiautomorphism of a quasigroup is presented in [18]. In this generalisation a generator element of the group Z_n is a quasiautomorphism (the third component of an autotopy) of a quasigroup.

7 Conclusion

In this paper we have presented stream deniable encryption algorithm, generalisation of quasigroup based Markovski algorithm, and based on Markovski algorithm and concept of isotopy generalisation of El Gamal encryption system.

Acknowledgment. The authors thank Referee for valuable suggestions.

References

[1] N. Moldovyan, A. Shcherbacov, and V. Shcherbacov, "On some applications of quasigroups in cryptology," in *Workshop on Foundations of Informatics, August 24-29, 2015, Chisinau, Proceedings*, Chisinau, 2015, pp. 331–341.

[2] R. Canetti, C. Dwork, M. Naor, and R.Ostrovsky, "Deniable Encryption," *Proceedings Advances in Cryptology CRYPTO 1997* (Lecture Notes in Computer Science, vol. 1294), pp. 90–104, 1997.

[3] A. Moldovyan and N. Moldovyan, "Practical method for bideniable public-key encryption," *Quasigroups and related systems*, vol. 22, pp. 277–282, 2014.

[4] A. Moldovyan, N. Moldovyan, and V. A. Shcherbacov, "Bideniable public-key encryption protocol secure against active coercive adversary," *Buletinul Academiei de Stiinte a Republicii Moldova. Matematica*, no. 3, pp. 23–29, 2014.

[5] V. Belousov, *Foundations of the Theory of Quasigroups and Loops*. Moscow: Nauka, 1967, (in Russian).

[6] J. Pieprzyk, T. Hardjono, and J. Seberry, *Fundumentals of Computer Security*. Berlin: Springer-Verlag, 2003.

[7] N. Moldovyan, "On cipher design based on switchable controlled operations," *International Journal of Network Security*, vol. 7, no. 3, pp. 404–415, 2008.

[8] N. Moldovyan and A. Moldovyan, *CData-driven block ciphers for fast telecommunication systems*. New York: Talor&Francis Group, 2008.

[9] V. Shcherbacov, "Quasigroups in cryptology," *Comput. Sci. J. Moldova*, vol. 17, no. 2, pp. 193–228, 2009.

[10] V. Shcherbacov and N. Moldovyan, "About one cryptoalgorithm," in *Proceedings of the Third Conference of Mathematical Society of the Republic of Moldova dedicated to the 50th anniversary of the foundation of the Institute of Mathematics and Computer Science, August 19-23, 2014, Chisinau*. Chisinau: Institute of Mathematics and Computer Science, 2014, pp. 158–161.

[11] V. Shcherbacov, "Elements of quasigroup theory and some its applications in code theory," 2003. [Online]. Available: www.karlin.mff.cuni.cz/drapal/speccurs.pdf.

[12] V. Shcherbacov, "Quasigroup based crypto-algorithms," pp. 1–23, 2012, arXiv:1201.3016.

[13] G. Mullen and V. Shcherbacov, "On orthogonality of binary operations and squares," *Bul. Acad. Stiinte Repub. Mold., Mat.*, no. 2, pp. 3–42, 2005.

[14] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans inf Theo*, vol. 31, no. 4, pp. 469–472, 1985.

[15] V. Shcherbacov, "On generalisation of Markovski cryptoalgorithm", AAA89: Workshop on General Algebra, February 26-

March 1, 2015, Technische Universität Dresden, Technical Report, 2015, Technische Universität Dresden, Dresden, pp. 36–37.

[16] V. Shcherbacov, "On the structure of left and right F-, SM- and E-quasigroups," *J. Gen. Lie Theory Appl.*, vol. 3, no. 3, pp. 197–259, 2009.

[17] V. Belousov, *n-Ary Quasigroups*. Kishinev: Stiintsa, 1971, (in Russian).

[18] A. V. Gribov, "Algebraic non-associative structures and its applications in cryptology," Ph.D. dissertation, Moscow: Moscow State University, 2015, (in Russian).

[19] G. Maze, "Algebraic methods for constructing one-way trapdoor functions," Ph.D. dissertation, University of Notre Dame, 2003.

N. A. Moldovyan[1], A. V. Shcherbacov[2],
V. A. Shcherbacov[3],

Received October 30, 2015

[1] Professor, St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences
14 Liniya, 39, St.Petersburg, 199178
Russia
E–mail: mdn.spectr@mail.ru

[2] M.Sc., Theoretical Lyceum "C. Sibirschi"
Lech Kaczyski str. 4, MD-2028, Chişinău
Moldova
E–mail: admin@sibirsky.org

[3] Dr., Institute of Mathematics and Computer Science
Academy of Sciences of Moldova Academiei str. 5, MD−2028 Chişinău
Moldova
Email: scerb@math.md

Computer Science Journal of Moldova, vol.**24**, no.1(70), 2016

Stream Deniable-Encryption Algorithms

N.A. Moldovyan A.A. Moldovyan D.N. Moldovyan
V.A. Shcherbacov

Abstract

A method for stream deniable encryption of secret message is proposed, which is computationally indistinguishable from the probabilistic encryption of some fake message. The method uses generation of two key streams with some secure block cipher. One of the key streams is generated depending on the secret key and the other one is generated depending on the fake key. The key streams are mixed with the secret and fake data streams so that the output ciphertext looks like the ciphertext produced by some probabilistic encryption algorithm applied to the fake message, while using the fake key. When the receiver or/and sender of the ciphertext are coerced to open the encryption key and the source message, they open the fake key and the fake message. To disclose their lie the coercer should demonstrate possibility of the alternative decryption of the ciphertext, however this is a computationally hard problem.

Keywords: cryptology, algorithm, stream, deniable, encryption.

MSC 2000: 94A60, 11S05.

1 Introduction

This paper is an extended version of the article [1].

The notion of deniable encryption (DE) was introduced by Canetti et al. in 1997 [2] as property of cryptographic protocols and algorithms to resist the so called coercive attacks that are performed by some adversary (coercer) that intercepts the ciphertext and has power to force

sender or/and receiver to open both the sent message and the encryption key. If the sender encrypts the secrete message using public key of the receiver of the message, then we have the case of the public-key deniable encryption schemes. If the encryption of the secrete message is performed using a shared secret key, then we have the case of the shared-key deniable encryption schemes.

The public-key DE protocols are applicable for preventing vote buying in the internet-voting systems [3] and for providing security of multiparty computations [4]. The shared-key DE algorithms represent interest for information protection in computer and telecommunication systems. In literature the following cryptoschemes are considered: sender-deniable [2], [3] (coercer attacks the sender of the ciphertext), receiver-deniable [4] (coercer attacks the receiver of the ciphertext), and bi-deniable [5] (coercer attacks the both parties of the secure communication session) cryptoschemes. The encryption scheme is deniable, if it provides possibility to the sender or/and to the receiver to open a fake message and a fake key instead of the secret ones so that disclosing their lie is a computationally infeasible problem for the coercer. Practical methods for bi-deniable public-key encryption have been proposed in [6], [7].

Fast methods for block deniable encryption are described in [8]. Those methods implement deniable encryption as simultaneous transformation of two different messages, secret and fake ones, using two keys, secret and fake ones, into the single ciphertext. In the paper [8] it has been also introduced the notion of the computational indistinguishability of the DE from the probabilistic encryption. The DE algorithm is considered as possessing such property, if it produces the ciphertext that can be also produced by some probabilistic-encryption algorithm used for ciphering the fake message with the fake key and some random input. The stream DE algorithms proposed in [8] and [9] are indistinguishable from some probabilistic encryption algorithms, however those algorithms are very slow. At present no practical and fast algorithms for shared-key stream DE are described in the literature, such algorithms are very attractive for practical application to provide information protection in computer and telecommunication systems

though.

The present paper proposes a method and algorithm for sufficiently fast stream bi-deniable encryption. Computational indistinguishability from a probabilistic stream encryption is used as a design criterion. The paper is organized as follows. Section 2 presents the design criteria. Section 3 and 4 present method and algorithm for stream bi-deniable encryption, correspondingly. Section 5 discusses the proposed algorithm. Section 6 concludes the paper.

2 Design criteria

For designing a shared-key DE algorithm the following criteria have been used:

- the algorithm should implement the stream encryption;
- the used encryption method should provide possibility of the independent decryption of each symbol of the produced ciphertext; this criterion takes into account possible practical applications in the cloud-computing technologies for processing data contained in encrypted files having large size;
- the method should implement the DE procedure as simultaneous encryption of the secret and fake messages using the secret and fake keys;
- the output ciphertext generated by the algorithm should be computationally indistinguishable from the ciphertext produced by some probabilistic ciphering a fake message with a fake key;
- the algorithm should provide sufficiently high encryption speed;
- the algorithm should provide bi-deniability;
- one should provide possibility of the independent recovering of the secret and fake messages, using secret or fake key, correspondingly.

3 Encryption method

One can consider text files as sequence of small data blocks having fixed size, i.e. as sequence of bit strings with which symbols are coded. Thus,

for encrypting a file or a message it is possible to apply formally the fast bi-deniable block-encryption method proposed in [7]. To encrypt a secret message $T = (T_1, T_2, \ldots, T_i, \ldots, T_n)$ represented as sequence of the b-bit data blocks T_i ($b = 32, 64, 128$, or 256), in that method it is supposed to generate a fake message $M = (M_1, M_2, \ldots, M_i, \ldots, M_n)$, where M_i are the b-bit data blocks, having the same size as the secret one and then to encrypt simultaneously all pairs of the data blocks T_i and M_i ($i = 1, 2, \ldots, n$) as follows:

1. Using some known secure block cipher with b-bit input data block, encrypt the data block M_i into the b-bit block C_{M_i} of intermediate ciphertext in accordance with the formula

$$C_{M_i} = E_K(M_i), \tag{1}$$

where E is the used block cipher; K is the fake key.

2. Encrypt the data block T_i into the b-bit block C_{T_i} of intermediate ciphertext in accordance with the formula

$$C_{T_i} = E_Q(T_i), \tag{2}$$

where Q is the secret key.

3. Compute the ith $(2b)$-bit block of the output ciphertext C_i as $(2b)$-bit binary polynomial satisfying the system of congruences

$$\begin{cases} C_i \equiv C_{M_i} \bmod \mu(x) \\ C_i \equiv C_{T_i} \bmod \lambda(x), \end{cases} \tag{3}$$

where binary polynomial $\mu(x) = 1||\mu'(x)$, $||$ denotes the concatenation operation; $\mu'(x)$ is the binary polynomial, which is given by the right b bits of the fake key K (i.e. the right b bits of the secret key K are interpreted as binary polynomial); binary polynomial $\lambda(x) = 1||\lambda'(x)$; $\lambda'(x)$ is the binary polynomial, which is given by the right b bits of the secret key Q.

In the method described in [8] the keys K and Q are generated as a pair of random bit strings such that polynomials $\mu'(x)$ and $\lambda'(x)$ are mutually irreducible, therefore the last system of congruences has

unique solution $C_i < \lambda(x)\mu(x)$ and can be computed as follows:

$$C_i = [C_{M_i}\lambda(x)(\lambda^{-1}(x) \bmod \mu(x)) + \\ + C_{T_i}\mu(x)(\mu^{-1}(x) \bmod \lambda(x))] \bmod \mu(x)\lambda(x).$$

In the case of small values of the data blocks the described method is insecure, for example, in the case of simultaneous encryption of the files $T = (t_1, t_2, \ldots, t_i, \ldots, t_n)$ and $M = (m_1, m_2, \ldots, m_i, \ldots, m_n)$, where t_i and m_i are symbols having size $b \leq 16$ bits. To overcome this problem we propose to modify the key for each value $i = 1, 2, \ldots, n$. Due to such modification it becomes possible to simplify computation of the blocks C_{M_i} and C_{T_i}, if the sequences of the modified values of the fake and secret key are generated in the form of some pseudorandom sequence that is computationally indistinguishable from the uniform random sequence. Besides, we propose to use unique fake and secret key sequences for encryption of each secret message T. Thus, we have come to idea to generate fake (Γ) and secret (Γ') key sequences using the block cipher E in accordance with the following formulas

$$E_K(i||V) \bmod 2^{2b} = (\alpha_i||\beta_i) \quad \text{and} \quad E_Q(i||V) \bmod 2^{2b} = (\alpha'_i||\beta'_i),$$

where α_i, β_i, α'_i, and β'_i are b-bit strings such that binary polynomials $\mu_i(x) = 1||\beta_i$ and $\lambda_i(x) = 1||\beta'_i$ are mutually irreducible; V is the 64-bit initialization vector generated at random for each encrypted message or file (the value V is not secret, therefore V can be transmitted via insecure channel).

The sequences Γ and Γ' can be written as follows:

$$\Gamma = \{(\alpha_1||\beta_1), (\alpha_2||\beta_2), \ldots, (\alpha_i||\beta_i), \ldots, (\alpha_n||\beta_n)\} \text{ and}$$
$$\Gamma' = \{(\alpha'_1||\beta'_1), (\alpha'_2||\beta'_2), \ldots, (\alpha'_i||\beta'_i), \ldots, (\alpha'_n||\beta'_n)\}.$$

The elements $(\alpha_i||\beta_i)$ and $(\alpha'_i||\beta'_i)$ of these sequences are to be used to encrypt simultaneously the couple of symbols t_i and m_i. Instead of formulas (1) and (2) one can use the following transformation of the ith symbol of the fake and secret messages, respectively:

$$c_{m_i} = m_i \oplus \alpha_i \quad (4)$$
$$c_{t_i} = t_i \oplus \alpha'_i, \quad (5)$$

where $\oplus$ is the XOR operation. The b-bit symbols c_{m_i} and c_{t_i} of the intermediate ciphertext are to be mixed into the single $(2b)$-bit symbol c_i of the output ciphertext in accordance with the following formula

$$\begin{aligned} c_i = [c_{m_i}\lambda_i(x)(\lambda_i^{-1}(x) \bmod \mu_i(x)) + \\ + c_{t_i}\mu_i(x)(\mu_i^{-1}(x) \bmod \lambda_i(x))] \bmod \mu_i(x)\lambda_i(x), \end{aligned} \tag{6}$$

where $\mu_i(x) = 1||\beta_i$ and $\lambda_i(x) = 1||\beta'$ are mutually irreducible binary polynomials. Formula (6) defines solution of the following system of congruences

$$\begin{cases} c_i \equiv c_{m_i} \bmod \mu_i(x) \\ c_i \equiv c_{t_i} \bmod \lambda_i(x). \end{cases} \tag{7}$$

System (7) defines the following formulas for computing the symbols c_{m_i} and c_{t_i} from c_i:

$$c_{m_i} = c_i \bmod \mu_i(x), \tag{8}$$

$$c_{t_i} = c_i \bmod \lambda_i(x). \tag{9}$$

Then the ith symbols t_i and m_i of the source texts T and M are computed using the values α_i and α'_i with the following formulas ($i = 1, 2, \ldots, n$):

$$m_i = c_{m_i} \oplus \alpha_i, \tag{10}$$

$$t_i = c_{t_i} \oplus \alpha'_i. \tag{11}$$

4 The stream deniable encryption algorithm

Suppose we have a secure block cipher E with 128-bit input data block and 128-bit key K. Using the method described in Section 3 (in which it is supposed that two parties of the communication session share the secret 128-bit key Q and the fake 128-bit key K) we have constructed the following algorithm for performing the stream DE of the secret message T:

INPUT: the secret message $T = (t_1, t_2, \ldots, t_i, \ldots, t_n)$ and encryption keys K and Q.

1. Generate a fake message M having the same length as the message T.

2. Generate a random value of the 64-bit initialization vector V.

3. For $i = 1$ to n do the following steps.

3.1. Using the procedure **Form_**$\alpha\beta$ generate the ith elements $(\alpha_i||\beta_i)$ and $(\alpha'_i||\beta'_i)$ of the key sequences Γ and Γ'.

3.2. Compute the b-bit symbols c_{m_i} and c_{t_i} of the intermediate ciphertext using formulas (4) and (5).

3.3. Compute the $(2b)$-bit symbol c_i of the output ciphertext as solution of the system of two linear congruences (7), which is defined by formula (6).

4. Compose the output ciphertext $C = (c_1, c_2, \ldots, c_i, \ldots, c_n)$.

OUTPUT: the ciphertext $C = (c_1, c_2, \ldots, c_i, \ldots, c_n)$ and the initialization vector V.

The procedure **Form_**$\alpha\beta$ used at step 3 is described as follows:

INPUT: two 128-bit keys K and Q and two 64-bit values i and V.

1. Compute the value $(\alpha_i||\beta_i) = E_K(i||V) \bmod 2^{2b}$, where E is some specified 128-bit block cipher; α_i and β_i are b-bit strings; the value $E_K(i||V)$ is considered as binary number.

2. Compose the bit string $\mu_i = (1||\beta_i)$.

3. Compute the value $(\alpha'_i||\beta'_i) = E_Q(i||V) \bmod 2^{2b}$.

4. Compose the bit string $\lambda_i = (1||\beta'_i)$.

5. Considering the bit strings μ_i and λ_i as binary polynomials $\mu_i(x)$ and $\lambda_i(x)$ of the degree b, respectively, compute the greatest common divisor $D = gcd(\mu_i(x), \lambda_i(x))$.

6. If $D \neq 1$, then increment $\beta'_i \leftarrow \beta'_i + 1 \bmod 2^b$ (here the bit string β'_i is considered as binary number) and go to step 4, otherwise STOP.

OUTPUT: two $(2b)$-bit elements $(\alpha_i||\beta_i)$ and $(\alpha'_i||\beta'_i)$ of the key sequences Γ and Γ'.

Decryption of the ciphertext C produced by the proposed DE algorithm requires using the value V assigned to C (i.e. sent together with the ciphertext C) and both the secret and fake keys. The following algorithm describes the decryption procedure.

Algorithm for decrypting the secret message.

INPUT: the ciphertext $C = (c_1, c_2, \ldots, c_i, \ldots, c_n)$, the encryption key Q, the fake key K, and the initialization vector V.

1 For $i = 1$ to n do the following steps.

1.1. Using the procedure **Form_**$\alpha\beta$ generate the ith element $(\alpha'_i||\beta'_i)$ of the key sequence Γ'.

1.2. Compute the b-bit symbol c_{t_i} of the intermediate ciphertext using the formula (9).

1.3. Compute the b-bit symbol t_i of the secret message using formula (11).

2. Compose the message $T = (t_1, t_2, \ldots, t_i, \ldots, t_n)$.

OUTPUT: the opened message T.

5 Discussion

5.1 Security against the two-side coercive attack

Suppose a coercive adversary intercepts the ciphertext and initialization vector sent by sender to receiver of secret message and then forces both the parties to open the message, the encryption and decryption algorithms, and the encryption key. The encryption algorithm proposed in Section 4 resists this attack, since the sender and the receiver are able to fulfill coercers demands without opening the secret message. For this purpose they open the following:

- the fake key K declared as the secret one;
- the fake message M declared as the secret one;
- probabilistic encryption algorithm that allegedly produced the ciphertext intercepted by the coercer;
- decryption algorithm that discloses the fake message from the cryptogram, while using the fake key.

To catch them in a lie, the coercer should show conclusively that the ciphertext contains another message. The last can be performed by guessing the secret key Q, however this method is impractical due to sufficiently large size of the value Q (128 bits).

Let us also consider the known-plaintext attack, i.e. suppose the coercer knows the secret message. If he is able to compute the secret key

Q, then he is able to prove that the sender and the receiver are cheating (the proving consists in opening the message T from the ciphertext C, while using the key Q). Suppose additionally that, using the known message T and the value V, the coercer is able to compute the key sequence Γ' and then all values $E_K(i||V)$, where $i = 1, 2, \ldots, n$ (see step 3 in description of the procedure **Form_**$\alpha\beta$).

In this case the assumption about possibility to compute the key Q from the known 128-bit input $i||V$ and output values $E_K(i||V)$ leads to conclusion about insecurity of the used block cipher E against the known-plaintext attack. However in the proposed DE algorithm it is used a secure block cipher, for example, AES that surely resists such attacks and is recommended by the standard ISO/IET 18033-3:2010 [10].

Thus, one can conclude the proposed DE algorithm provides bi-deniability. The probabilistic encryption algorithm to be opened to the coercer is described as follows.

Associated probabilistic stream encryption algorithm

INPUT: the message $M = (m_1, m_2, \ldots, m_i, \ldots, m_n)$ and the encryption key K.

1. Generate a random value of the 64-bit initialization vector V.

2. For $i = 1$ to n do the following steps.

2.1. Compute the value $(\alpha_i||\beta_i) = E_K(i||V) \bmod 2^{2b}$, where E is the specified 128-bit block cipher; α_i and β_i are b-bit strings; the value $E_K(i||V)$ is considered as binary number.

2.2. Compose the bit string $\mu_i = (1||\beta_i)$.

2.3. Generate randomly two b-bit strings ρ and η'.

2.4. Compose the bit string $\eta = (1||\eta')$.

2.5. Considering the bit strings μ_i and η as binary polynomials $\mu_i(x)$ and $\eta(x)$ of the degree b, respectively, compute the greatest common divisor $D = gcd(\mu_i(x), \eta(x))$.

2.6. If $D \neq 1$, then increment $\eta' \leftarrow \eta' + 1 \bmod 2^b$ (here the bit string η' is considered as binary number) and go to step 2.4, otherwise go to step 2.7.

2.7. Compute the b-bit symbol c_{m_i} of the intermediate ciphertext using formula (4).

2.8. Compute the $(2b)$-bit symbol c_i as solution of the following system of two linear congruences:

$$\begin{cases} c_i \equiv c_{m_i} \bmod \mu_i(x) \\ c_i \equiv \rho(x) \bmod \eta(x), \end{cases} \tag{12}$$

where the bit string c_{m_i} is considered as binary polynomial and $\rho(x)$ is the binary polynomial represented by the bit string ρ.

3. Compose the output ciphertext $C = (c_1, c_2, \ldots, c_i, \ldots, c_n)$.
OUTPUT: the ciphertext $C = (c_1, c_2, \ldots, c_i, \ldots, c_n)$ and the initialization vector V.

The value c_i at step 2.8 can be computed using the following formula:

$$\begin{aligned} c_i = [c_{m_i}\eta(x)(\eta^{-1}(x) \bmod \mu_i(x)) + \\ + \rho(x)\mu_i(x)(\mu_i^{-1}(x) \bmod \eta(x))] \bmod \mu_i(x)\eta(x). \end{aligned}$$

It is easy to see that for each symbol c_i of the ciphertext C there exist different bit strings η' and ρ satisfying system (12). Indeed, for given c_i and arbitrary η' such that $gcd(\mu_i(x), \eta(x)) = 1$ the value ρ satisfying (12) can be computed as binary polynomial $\rho(x) = c_i \bmod \eta(x)$, where the bit string c_i is considered as binary polynomial.

Thus, while using the encryption key K the associated probabilistic encryption algorithm can potentially encrypt the message M into the cryptogram C produced by the DE algorithm. Since it is computationally difficult to prove that the ciphertext C was produced by the DE process, but not by the probabilistic encryption, one can say the proposed DE algorithm is computationally indistinguishable from the associated probabilistic encryption algorithm.

The decryption algorithm to be opened to the coercer is described as follows.

Dishonest decryption algorithm

INPUT: the ciphertext $C = (c_1, c_2, \ldots, c_i, \ldots, c_n)$, the encryption key K, and the initialization vector V.

1. For $i = 1$ to n do the following steps.

1.1. Compute the value $(\alpha_i||\beta_i) = E_K(i||V) \bmod 2^{2b}$.

1.2. Compose the bit string $\mu_i = (1||\beta_i)$.

1.3. Compute the b-bit symbol c_{m_i} of the intermediate ciphertext using the formula (8).

1.4. Compute the b-bit symbol m_i using the formula (10).

2. Compose the message $M = (m_1, m_2, \ldots, m_i, \ldots, m_n)$.

OUTPUT: the opened message M.

5.2 Estimation of the encryption speed

For comparing the performance of the proposed algorithm with the stream DE algorithm described in [7] one can roughly assume that time complexity of computation of the value c_i in accordance with the formula (6) is equal to the time complexity of one block encryption operation. Besides, the time complexity of generation of the values $(\alpha_i||\beta_i) = E_K(i||V) \bmod 2^{2b}$ and $(\alpha'_i||\beta'_i) = E_Q(i||V) \bmod 2^{2b}$ is approximately equal to 1 and 2 block-encryption operations, correspondingly.

Thus, the time complexity of the encryption of one symbol of the secret message is equal to ≈ 4 block-encryption operations. Taking the last into account one can get the following formula for the encryption speed of the proposed algorithm:

$$S = \frac{1}{4} \cdot \frac{b}{128} S_E, \tag{13}$$

where b is the bit length of the symbols with which the secret message is written; S_E is the encryption speed of the block cipher E.

While implementing the DE method from [8] using the block cipher E, encryption of one symbol of the secret message takes on the average 2^{2b+1} operations of the block encryption and defines the following formula for estimating the speed:

$$S_{[7]} = \frac{b}{128} \cdot \frac{S_E}{2^{2b+1}}. \tag{14}$$

Comparing (13) with (14) one can state that the proposed stream DE algorithm is significantly faster (by 2^{2b-1} times) than algorithm by method in [8]. For example, in the case $b = 8$ the ratio S/S_E is equal to 2^{15}.

5.3 Probabilistic deniable encryption

The feasibility of practical use of the probabilistic encryption is related to its providing better statistical properties of ciphertext. This thesis provides credence to both the associated probabilistic stream encryption algorithm and the dishonest decryption algorithm that are to be presented to the coercive attacker together with the fake key. In the stream deniable encryption algorithm described in Section 4 no random values are used, i.e. it is deterministic after the fake message was generated. From practical point of view it is interesting that one can modify the proposed deniable encryption algorithm into the probabilistic deniable encryption one. Indeed, the probabilistic deniable encryption of the messages T and M can be performed as follows:

1. Generate a random value of the 64-bit initialization vector V.

2. For $i = 1$ to n do the following steps.

2.1. Using the procedure **Form_**$\alpha\beta$ generate the ith elements $(\alpha_i||\beta_i)$ and $(\alpha'_i||\beta'_i)$ of the key sequences Γ and Γ'.

2.2. Compute the b-bit symbols c_{m_i} and c_{t_i} of the intermediate ciphertext using formulas (4) and (5).

2.3. Generate at random two b-bit strings ρ and η'.

2.4. Compose the bit string $\eta = (1||\eta')$.

2.5. Considering the bit strings μ_i, λ_i, and η as binary polynomials $\mu_i(x)$, $\lambda_i(x)$, and $\eta(x)$ of the degree b, respectively, compute the greatest common divisors $D_1 = gcd(\mu_i(x), \eta(x))$ and $D_2 = gcd(\lambda_i(x), \eta(x))$.

2.6. If $D_1 \neq 1$ or $D_2 \neq 1$, then increment $\eta' \leftarrow \eta' + 1 \bmod 2^b$ (here the bit string η' is considered as binary number) and go to step 2.4, otherwise go to step 2.7.

2.7. Compute the $(3b)$-bit symbol c_i of the output ciphertext as

solution of the system of the following three linear congruences

$$\begin{cases} c_i \equiv c_{m_i} \bmod \mu_i(x) \\ c_i \equiv c_{t_i} \bmod \lambda_i(x) \\ c_i \equiv \rho(x) \bmod \eta(x), \end{cases}$$

which is defined by the formula

$$\begin{aligned} c_i = \lfloor & c_{m_i}\lambda_i(x)\eta(x)\left(\lambda_i^{-1}(x)\eta^{-1}(x) \bmod \mu_i(x)\right) + \\ & + c_{t_i}\mu_i(x)\eta(x)\left(\mu_i^{-1}(x)\eta^{-1}(x) \bmod \lambda_i(x)\right) + \\ & + \rho(x)\lambda_i(x)\mu_i(x)\left(\lambda_i^{-1}(x)\mu_i^{-1}(x) \bmod \eta(x)\right)\rfloor \bmod \lambda_i(x)\mu_i(x)\eta(x) \end{aligned}$$

3. Compose the output ciphertext $C = (c_1, c_2, \ldots, c_i, \ldots, c_n)$.

The associated probabilistic encryption algorithm connected with the last one is the same as that described in Subsection 5.1, except at step 2.3 there are generated $(2b)$-bit random values ρ and η'. The corresponding dishonest decryption algorithm is exactly the same as that described in Subsection 5.1.

6 Conclusion

It is proposed a method and algorithm for fast stream deniable encryption satisfying criterion of the computational indistinguishability from the stream probabilistic encryption. It has been shown that the DE algorithm resists two-side coercive attack. As compared with the stream DE algorithm presented in [8] the proposed one is significantly faster, the algorithm from [8] has one interesting advantage though. The advantage consists in using the same decryption algorithm for opening both the secret and the fake messages from the ciphertext. Such property is significant for providing security against coercive attacks combined with measuring duration of the decryption process. In our future research we plan to develop a fast stream DE method with the same algorithm that decrypts the secret and fake message.

References

[1] A. A. Moldovyan, D. N. Moldovyan, and V. A. Shcherbacov, "Stream deniable-encryption algorithm satisfying criterion of the computational indistinguishability from probabilistic ciphering," in *Workshop on Foundations of Informatics, August 24-29, 2015, Chisinau, Proceedings*, Chisinau, 2015, pp. 318–330.

[2] R. Canetti, C. Dwork, M. Naor, and R.Ostrovsky, "Deniable Encryption," *Proceedings Advances in Cryptology CRYPTO 1997* (Lecture Notes in Computer Science, vol. 1294), pp. 90–104, 1997.

[3] J. Howlader and S. Basu, "Sender-side public key deniable encryption scheme," in *Advances in Recent Technologies in Communication and Computing. Proceedings of the ARTCom '09 International Conference*, 2009, pp. 9–13, dOI: 10.1109/ARTCom.2009.107.

[4] B.Meng and J. Wang, "A receiver deniable encryption scheme," in *Proceedings of the 2009 International Symposium on Information Processing (ISIP09)*, Huangshan, P. R. China, Aug. 2009, pp. 254–257.

[5] A. O'Neil, C. Peikert, and B. Waters, "Bi-deniable public-key encryption," in *Advances in Cryptology CRYPTO 2011* (Lecture Notes in Computer Science, vol. 6841), Berlin: Springer Verlag, 2011, pp. 525–542.

[6] A. Moldovyan and N. Moldovyan, "Practical method for bi-deniable public-key encryption," *Quasigroups and related systems*, vol. 22, pp. 277–282, 2014.

[7] A. Moldovyan, N. Moldovyan, and V. A. Shcherbacov, "Bi-deniable public-key encryption protocol secure against active coercive adversary," *Buletinul Academiei de Stiinte a Republicii Moldova. Matematica*, no. 3, pp. 23–29, 2014.

[8] E. Morozova, Y. Mondikova, and N.A.Moldovyan, "Methods of Deniable Encryption with a Shared Key," *Informatsionno-upravliaiushchie sistemy (Information and Control Systems)*, no. 6(67), pp. 73–78, 2013, (in Russian).

[9] N. Moldovyan, A. R. Birichevskiy, and Y. Mondikova, "Deniable encryption based on block ciphers," *Informatsionno-upravliaiushchie sistemy(Information and Control Systems)*, no. 5(72), pp. 80–86, 2014, (in Russian).

[10] I. standard ISO/IEC 18033-3:2010, "Information technology – security techniques – encryption algorithms – part 3: Block ciphers," 2010.

N. A. Moldovyan[1] A. A. Moldovyan[2],
D. N. Moldovyan[3], V. A. Shcherbacov[4],

Received July 10, 2015

[1] Professor, St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences
14 Liniya, 39, St.Petersburg, 199178
Russia
E–mail: nmold@mail.ru

[2] Professor, ITMO University
Kronverksky pr., 10, St.Petersburg, 197101
Russia
E–mail: maa1305@yandex.ru

[3] Dr., St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences
14 Liniya, 39, St.Petersburg, 199178
Russia E–mail: mdn.spectr@mail.ru

[4] Dr., Institute of Mathematics and Computer Science
Academy of Sciences of Moldova Academiei str. 5, MD−2028 Chişinău
Moldova
E–mail: scerb@math.md

Computer Science Journal of Moldova, vol.**24**, no.1(70), 2016

Taxonomy of Strategic Games with Information Leaks and Corruption of Simultaneity

Valeriu Ungureanu

Abstract

We consider pseudo-simultaneous normal form games — strategic games with rules violated by information leaks and simultaneity corruption. We provide classification and construction of a game taxonomy based on applicable solution principles. Existence conditions are highlighted, formulated and analysed.

Keywords: Non-cooperative game, Nash equilibrium, simultaneous and sequential games, Stackelberg equilibrium, knowledge, information leak, corruption, taxonomy.

MSC 2010: 91A05, 91A06, 91A10, 91A43, 91A20, 91A26, 91A44, 91A65.

1 Introduction

Strategic or normal form game constitutes an abstract mathematical model of decision processes with two or more decision makers (players) [5], [6]. An important supposition of the game is that all the players choose their strategies simultaneously and confidentially, and that everyone determines his gain on the resulting profile. Reality is somewhat diverse. The rules of the games may be broken. Some players may cheat and know the choices of the other players. So, the rule of confidentiality and simultaneity is not respected. Is the essence of initial normal form game change in such games? Is still the Nash equilibrium principle applicable? Do we need other solution principles and other interpretations? How many types of games appear and may they be classified? Can we construct a taxonomy (classification) of these

games? Answers to these and other related questions are the objective of presented work.

Usually, the traditional research approach to games of such types relies on consideration of all possible players' best response mappings and analysis of all possible profiles [3]. There is a stable opinion about a high complexity of their analysis and solving [3].

We initiate an approach which sets rules of all possible games with information leaks and highlights their specific characteristics. The approach relies on knowledge vectors of the players and game knowledge net. A taxonomy (classification) of all possible games is done on the bases of the applicable solution principles. The name of every taxon (class) reflects the principle used for including respective games in the same taxon.

As a result of the taxonomy construction and establishing strict characteristics and rules for every taxon, we reveal simplicity of analysis and solving the games. It is an unexpected and impressive result.

For the beginning, let us remember that a Nash equilibrium (NE) sample and the entire Nash equilibrium set (NES) may be determined via intersection of the graphs of best response mappings — a method considered earlier in works [8]–[13]. The approach proves to be expedient for strategic games with information leaks and broken simultaneity as well. Initially, we expose the results for two-matrix games with different levels of knowledge. Then, we expose results for the general multi-matrix game. It is a useful approach both for simplicity of exposition, and for understanding the ideas and results.

1.1 Normal form game and axioms

Consider finite strategic (normal form) game

$$\Gamma = \left\langle N, \{S_p\}_{p\in N}, \left\{a^p_{\mathbf{s}} = a^p_{s_1 s_2 \ldots s_n}\right\}_{p\in N} \right\rangle,$$

where

$N = \{1, 2, ..., n\}$ is a set of players,
$S_p = \{1, 2, \ldots, m_p\}$ is a set of strategies of player $p \in N$,

$s_p \in S_p$ is a strategy of $p \in N$ player,
$\#S_p = m_p < +\infty$, $p \in N$, is a finiteness constraint,
$a^p_{\mathbf{s}} = a^p_{s_1 s_2 \ldots s_n}$ is a player's $p \in N$ pay-off function defined on Cartesian product $S = \underset{p \in N}{\times} S_p$, i.e. for every player $p \in N$ a n dimensional pay-off matrix $A^p[m_1 \times m_2 \times \cdots \times m_n]$ is defined.

For normal form game a system of axioms is stated.

Axiom 1.1. Rationality. *The players behave rationally. The rationality means that every rational player optimizes the value of his pay-off function.*

Axiom 1.2. Knowledge. *The players know the set of players, the strategy sets and the pay-off functions.*

Axiom 1.3. Simultaneity. *The players choose their strategy simultaneously and confidentially in a single-act (single stage) without knowing the chosen strategies of the other players.*

Axiom 1.4. Pay-off. *After all strategy selection the players compute their pay-off as the values of their pay-off functions on the resulting profile.*

Traditionally, simultaneous Nash games [5], [6] are based on these four axioms.

In Stackelberg game the axiom of simultaneity is replaced by the axiom of hierarchy.

Axiom 1.5. Hierarchy. *The players choose their strategies in a known order, e.g. the first player chooses his strategy and communicates it to the second player. The second player (follower) knows the strategy of the leader, chooses his strategy and communicates it to the third player and so on. The last player knows all the strategies of the precedent players and chooses his strategy the last.*

Both the Nash game, and the Stackelberg game, are based commonly on axioms of rationality, knowledge and pay-off. Additionally

and distinctively the Nash game [5] is based on axiom of simultaneity, while the Stackelberg game [4], [7] is based on axiom of hierarchy.

Finally, we can deduce that in Nash game all the players choose their strategies simultaneously and every player determines his gain as the value of his pay-off function on the resulting profile. But in Stackelberg game the players choose their strategies sequentially, in a known order and knowing the strategies chosen by the precedent players, and every player determines his gain as the value of his pay-off function on the resulting profile.

1.2 Axiom of simultaneity and its corruption

Both for Nash games, and for Stackelberg games, their own solution principles exist. If the axioms of the games are respected, these solution principles may be applied. Actually, the name of the games are chosen to reflect the solution concept which is applicable.

But, what does it happen when the axioms are violated by the corruption of some of their elements, e.g. some players may know chosen strategies of the other players in Nash games? May such games be examined by applying the same general solution principles (Nash and Stackeberg equilibria) or must new solution concepts be defined and applied?

To respond to these questions, we need to avoid the ambiguity. So, let us examine more exactly the process of decision making in conditions of information leaks, by establishing the axioms of the corrupt games. It is very convenient to consider in such case a manager of the games — a person (or persons) which organizes and manages the decision process in the games. Thereby, we can describe exactly the process of decision making in corrupt games, knowing the source of corruption[1].

At the **first pseudo-stage** of the decision process, the manager declares the players must choose their strategies. After players choose their strategies (intentions), the manager dishonestly from the view-

[1]Corruption: "the abuse of entrusted power to private gain" (Transparency International); "dishonest or fraudulent conduct by those in power, typically involving bribery" (Oxford Dictionary, 2014).

point of the rules of the strategy game may submit to some players information about chosen strategies (corruption and information leaks). With additional information, some of players may want to change their strategies and they can do this at the second pseudo-stage.

At the **second pseudo-stage**, the manager declares the players must submit immediately their choices. At this moment, the players may change their initial decisions. After possible changes in their intentions the players submit definitely their chosen strategies.

For an honest player, the decision process looks like a mono stage process. Only for the dishonest manager and for the players which obtain additional information, the decision process looks like a two stage process.

As an axiom of such a game the axiom of information leak may be stated as

Axiom 1.6. Information leak. *The decision process has two pseudo-stages.*

At the **first pseudo-stage**, *the information leak about player chosen strategies may occur.*

At the **second pseudo-stage**, *the players choose their strategies, some of them knowing eventually the strategies chosen by the other players.*

Definition 1.1. *Let us define a game with information leak or a corrupt game as a game for which four axioms are fulfilled: rationality, knowledge, pay-off and information leak.*

Remark 1.1. *Let us observe that three axioms of rationality, knowledge and pay-off, are common for Nash game, Stackelberg game and corrupt game (game with information leak).*

Remark 1.2. *Generally, the game with information leak is only a particular case of the corrupt game. We will use interchangeably this name unless we will define a more general context of corrupt game.*

Remark 1.3. *The game with information leak as it is defined above actually includes different types of games.*

To respond to the various questions which appear in the context of corrupt games we consider further the taxonomy (classification) of the all possible types of games, principles of solutions, solution existence conditions and algorithms for solutions determining. The exposition will start with two-matrix games, but firstly we must highlight shortly in this context the essence of so named theory of moves.

1.3 Theory of moves

We must observe that the games we consider in this work may be related to the theory of moves [2]. Nevertheless, it is an important difference — we consider only two pseudo-stages of the decision making process, while the theory of moves does not limit the number of moves to one fixed number. Moreover, theory of moves has initial axioms which are defined in a strict manner as the process of decision making, as the end condition. Additionally, those axioms differ from that accepted for games with information leaks.

The theory of moves is based on the concepts of thinking ahead, stable outcomes, outcomes induced when one player has "moving power", incomplete information, non-myopic concept of equilibrium, etc. The non-myopic equilibrium depends on some parameters, such as, e.g., initial state from which the process of moving starts and who moves the first. It is essential that all games have at least one non-myopic equilibrium. In the games we consider, there are different solution concepts and it is not guaranteed that the solutions exist. These thoughts we expose further.

2 Taxonomy of two-matrix games with information leaks

For two-matrix strategic games, we suppose the process of making decision occurs in two pseudo-stages, because of possible information leaks. At the first pseudo-stage, the players choose their strategies and by corruption, it is possible either for one of them, or for both players, to know the chosen (intention) strategy of the opponent. At the second

pseudo-stage, the players use the obtained information, choose their strategies and no more corruption is possible.

First, for such processes of decision making we can distinguish **simultaneous** and **sequential** two-matrix games.

Second, simultaneous two-matrix games may obtain some features of sequential games taking into consideration the obtained information/knowledge ($\gamma\nu\omega\sigma\eta$) possessed by each player in the process of realizing the game.

Remark 2.1. *We suppose initially, when the players begin the strategy selections, they start playing a Nash game, but in the process of strategy selections information leak may occur, the Nash game may degenerate and may change its essence.*

Remark 2.2. *In order to distinguish players without their numbers, we will refer to them as the player and his opponent. So, if the first player is referred to simply as the player, then the second player is referred to as the opponent, and vice versa.*

2.1 Knowledge and types of games

The knowledge of the players is associated with their knowledge vectors γ^A and γ^B.

2.1.1 Knowledge vectors

Essentially, the knowledge vectors have an infinite number of components $\gamma^A = \left(\gamma_0^A, \gamma_1^A, \dots\right)$ and $\gamma^B = \left(\gamma_0^B, \gamma_1^B, \dots\right)$, with components defined and interpreted as it follows.

- **Player's knowledge of the normal form components.** γ_0^A and γ_0^B are reserved to knowledge about the normal form of the game. Values $\gamma_0^A = 1$ and $\gamma_0^B = 1$ mean that the players have full information about the strategy sets and pay-off functions. It is the case we consider in this work, i.e. mutually $\gamma_0^A = 1$ and $\gamma_0^B = 1$, and these components of the knowledge vectors are simply omitted.

- **Player's knowledge of the opponent's chosen strategy.** Values $\gamma_1^A = 0$ and $\gamma_1^B = 0$ mean for each player, correspondingly, that he doesn't know the opponent's strategy. $\gamma_1^A = 1$ and $\gamma_1^B = 1$ mean for each player, correspondingly, that he knows the opponent's strategy. The combined cases are possible, too.

- **Player's knowledge of the opponent's knowledge of the player's chosen strategy.** $\gamma_2^A = 0$ and $\gamma_2^B = 0$ mean for each player, correspondingly, that he knows that the opponent doesn't know player's strategy. $\gamma_2^A = 1$ and $\gamma_2^B = 1$ mean for each player, correspondingly, that he knows that the opponent knows player's strategy. Evidently, the combined cases are possible, too. Remark that these components may be thought rather as the players beliefs, because such type of knowledge may be as true, as false. In this context it must be observed, that the values of γ_2^A and γ_2^B represent the knowledge/belief about the values of γ_1^B and γ_1^A, correspondingly.

- **The next components $\gamma_3^A, \gamma_4^A, \ldots$ and $\gamma_3^B, \gamma_4^B, \ldots$ of the knowledge vectors are omitted, initially.** Nevertheless, it must be remarked that the values of γ_i^A and γ_i^B represent the knowledge/belief about the values of γ_{i-1}^B and γ_{i-1}^A, correspondingly.

We distinguish the games with l levels of knowledge, for which all components of the knowledge vectors with indices greater than l are equal to 0.

Remark 2.3. *Remark, once again, that there are two pseudo-levels of decision making process. Information leaks may occur only at the first pseudo-level. The knowledge vectors may have any number $l \geq 1$ of components (levels of knowledge).*

2.1.2 Types of games

Depending on the values of the knowledge vectors, different types of games may be considered.

Proposition 2.1. *There are* 4^l *possible types of games* $\Gamma_{\gamma^A\gamma^B}$ *with* l *levels of knowledge.*

Proof. It is enough to emphasize the components of the knowledge vectors $\gamma^A = \left(\gamma_1^A, \ldots, \gamma_l^A\right)$ and $\gamma^B = \left(\gamma_1^B, \ldots, \gamma_l^B\right)$ and their possible values as 0 or 1. Accordingly, there are 4^i possible pairs of such vectors, i.e. 4^l possible games. □

2.1.3 Knowledge net

Knowledge net is defined as:

$$G = (V, E)\,,$$

where $V = I \cup J \cup \gamma_A \cup \gamma_B, E \subseteq V \times V$.

For the present we will limit ourselves to knowledge vectors.

2.2 Taxonomy Elements

If the information leaks occur only at the first 2 levels, then there are $4^2 = 16$ possible kinds of games with information leaks with $\gamma^A = \left(\gamma_1^A, \gamma_2^A\right)$ and $\gamma^B = \left(\gamma_1^B, \gamma_2^B\right)$, according to the above. From the solution principle perspective, some of them are similar and they may be included in common taxa (classes, families, sets).

Let us highlight the possible kinds of such games by the values of low index, where the first two digits are the values for knowledge vector components of the first player, and the following two digits are the values for knowledge vector components of the second player. We obtain the following taxonomy for two matrix games with information leaks on two levels:

1. Nash taxon: $\mathrm{NT} = \{\Gamma_{00\,00}, \Gamma_{11\,11}, \Gamma_{00\,11}, \Gamma_{11\,00}\}\,,$
2. Stackelberg taxon: $\mathrm{ST} = \{\Gamma_{01\,10}, \Gamma_{01\,11}, \Gamma_{10\,01}, \Gamma_{11\,01}\}\,,$
3. Maximin taxon: $\mathrm{MT} = \{\Gamma_{01\,01}\}\,,$
4. Maximin-Nash taxon: $\mathrm{MNT} = \{\Gamma_{00\,01}, \Gamma_{01\,00}\}\,,$

5. Optimum taxon: $\mathrm{OT} = \{\Gamma_{10\,10}\}$,
6. Optimum-Nash taxon: $\mathrm{ONT} = \{\Gamma_{00\,10}, \Gamma_{10\,00}\}$,
7. Optimum-Stackelberg taxon: $\mathrm{OST} = \{\Gamma_{10\,11}, \Gamma_{11\,10}\}$.

The generic name of each taxon is selected on the basis of the correspondent solution principles applied by the players: Nash equilibrium, Stackelberg equilibrium, Maximin principle, Optimum principle or two of them together. Even though the taxon may include some games, the name reflects solution principle or principles applied in all the games of the taxon. If the taxon is formed only by one element, its name is the same as for game.

Remark 2.4. *We choose the term taxon (plural — taxa) to name the set of the related games in order to highlight additionally their acquired pseudo-dynamics [1],[4],[7] and to avoid confusion with mathematically overcharged or too used terms of class, cluster, family, group or set.*

Let us investigate the solution principles for all these taxa.

3 Solution principles of two-matrix games with information leaks on two levels of knowledge

Consider a two-matrix $m \times n$ game Γ with matrices

$$A = (a_{ij}),\ B = (b_{ij}),\ i \in I, j \in J,$$

where $I = \{1, 2, \ldots, m\}$ is the set of strategies of the first player, and $J = \{1, 2, \ldots, n\}$ is the set of strategies of the second player.

We consider the games base on four axioms of rationality, knowledge, pay-off and information leak (axioms 1.1, 1.2, 1.4, 1.6). The players choose simultaneously their strategies and before submitting the results of their selections, the information leaks may occur. One or both of them may know the intention of the opponent. Let us suppose in such case, they may change only once their strategy according to the leaked information. So, the strategic games may be transformed

by acquiring additional information into two stage games. At the first stage they choose strategies but do not submit them because of acquiring additional information. At the second stage, according to the leaked information they may change the initial strategies and submit definitely new strategies adjusted to the obtained information. After such submission, the games end and both players determine the values of their pay-off functions.

Evidently, other types of games with information leaks may be considered. Firstly, we will limit ourselves only to such two-pseudo-stage games with information leaks on two levels of knowledge.

For every taxon we will firstly define it, after that we will argue its consistency.

3.1 Nash Taxon

Let us argue that for $\mathrm{NT} = \{\Gamma_{00\,00}, \Gamma_{11\,11}, \Gamma_{00\,11}, \Gamma_{11\,00}\}$ all its elements are Nash games, i.e. axioms 1.1–1.4 are characteristic, too, for these games and for them Nash equilibrium principle may be applied as a common solution principle.

Firstly, let us remember, that the process of decision making in the Nash game, denoted by $\mathrm{N}\Gamma$, is described in the following way. Simultaneously and confidentially, the first player selects the lines i^* of the matrices A and B, and the second player selects columns j^* of the same matrices. The first player gains $a_{i^*j^*}$, and the second player gains $b_{i^*j^*}$.

Evidently, $\Gamma_{00\,00}$ is a pure Nash game, i.e. $\Gamma_{00\,00} = \mathrm{N}\Gamma$. But, it is not difficult to understand that $\Gamma_{11\,11}$, $\Gamma_{00\,11}$, $\Gamma_{11\,00}$, are Nash games, too. So, the taxon (group) is formed by four Nash games, differing only by the knowledge/belief of the players.

If we call the player which applies a Nash equilibrium strategy as an atom (a Nash atom, Nash atomic player) and denote him as N, then the two-player Nash game may be denoted as N_2 (Nash game, Nash molecular game).

Remark 3.1. *We will name and denote in the same manner other types of players and games.*

3.1.1 Nash Equilibrium

The pair of strategies (i^*, j^*) forms a Nash equilibrium if

$$a_{i^*j^*} \geq a_{ij^*}, \forall i \in I,$$
$$b_{i^*j^*} \geq b_{i^*j}, \forall j \in J.$$

3.1.2 Set of Nash Equilibria

An equivalent Nash equilibrium definition may be formulated in terms of graphs of best response (optimal reaction) applications (mappings).

Let

$$\mathrm{Gr_A} = \left\{ (i,j) : \, j \in J, \, i \in \underset{k \in I}{\mathrm{Argmax}} \, a_{kj} \right\},$$

be the graph of best response application of the first player, and

$$\mathrm{Gr_B} = \left\{ (i,j) : \, i \in I, \, j \in \underset{k \in J}{\mathrm{Argmax}} \, b_{ik} \right\}.$$

be the graph of best response application of the second player.

$$\mathrm{NE} = \mathrm{Gr_A} \cap \mathrm{Gr_B}$$

forms the set of Nash equilibria.

3.1.3 Nash Equilibrium Existence

Proposition 3.1. *There are Nash games which do not have a Nash equilibrium.*

Proof. Examples of games which do not have a Nash equilibrium are commonly known. □

Remark, the games we consider are pure strategy games. It is a largely known result that every poly-matrix strategic game has Nash equilibria in mixed strategies. In this work we consider only pure strategy games.

3.2 Stackelberg Taxon

Stackelberg Taxon is defined as $\mathrm{ST} = \{\Gamma_{01\,10}, \Gamma_{01\,11}, \Gamma_{10\,01}, \Gamma_{11\,01}\}$. To argue the inclusion of each element in ST, let us remember the decision making process in the Stackelberg game.

Stackelberg two player game has two stages, from the start, and for Stackelberg game the axioms 1.1, 1.2, 1.4 and 1.5 are characteristic. At the first stage, the first player (leader) selects the lines i^* of the matrices A and B, and communicates his choice to the second player (follower). At the second stage, the second player (follower) knows the choice of the first player (leader) and selects columns j^* of the matrices A and B. The first player gains $a_{i^*j^*}$, and the second player gains $b_{i^*j^*}$. If the players change their roles as the leader and the follower, an another Stackelberg game is defined.

The Stackelberg game is denoted by SG_{12} if the first player is the leader and by SG_{21} if the second player is the leader.

$\Gamma_{01\,10}$ is a pure Stackelberg game $\mathrm{S}\Gamma_{12}$, i.e. $\Gamma_{01\,10} = \mathrm{S}\Gamma_{12}$, and $\Gamma_{10\,01}$ is a pure Stackelberg game $\mathrm{S}\Gamma_{21}$, i.e. $\Gamma_{10\,01} = \mathrm{S}\Gamma_{21}$. It is clear that $\Gamma_{01\,11} = \mathrm{S}\Gamma_{12}$ and $\Gamma_{11\,01} = \mathrm{S}\Gamma_{21}$.

3.2.1 Stackelberg Equilibrium

The pair of strategies $(i^*, j^*) \in \mathrm{Gr_B}$ forms a Stackelberg equilibrium if

$$a_{i^*j^*} \geq a_{ij}, \forall (i,j) \in \mathrm{Gr_B}.$$

If the players change their roles and the second player is the leader, then the pair of strategies $(i^*, j^*) \in \mathrm{Gr_A}$ forms a Stackelberg equilibrium if

$$b_{i^*j^*} \geq b_{ij}, \forall (i,j) \in \mathrm{Gr_A}.$$

3.2.2 Set of Stackelberg Equilibria

The sets of Stackelberg equilibria are generally different for Stackelberg games $\mathrm{S}\Gamma_{12}$ and $\mathrm{S}\Gamma_{21}$.

$$\mathrm{SE}_{12} = \underset{(i,j)\in \mathrm{Gr_B}}{\mathrm{Argmax}}\, a_{ij}$$

forms the set of Stackelberg equilibria in a Stackelberg game $\mathrm{S}\Gamma_{12} = \mathrm{S}_{12}$.

$$\mathrm{SE}_{21} = \underset{(i,j)\in \mathrm{Gr_A}}{\mathrm{Argmax}}\, b_{ij}$$

forms the set of Stackelberg equilibria in a Stackelberg game $\mathrm{S}\Gamma_{21} = \mathrm{S}_{21}$.

It is evident that the notions of Nash and Stackeberg equilibria are not identical. The respective sets of equilibria may have common elements, but the sets generally differ.

3.2.3 Stackelberg Equilibrium Existence

Proposition 3.2. *Every finite Stackelberg game has a Stackelberg equilibrium.*

Proof. The proof follows from the Stackelberg equilibrium definition and the finiteness of the player strategy sets. □

3.3 Maximin Taxon

Maximin Taxon contains only one element $\mathrm{MT} = \{\Gamma_{01\,01}\}$.

The decision making process in the Maximin game $\mathrm{M}\Gamma = \mathrm{M}_2$ follows the axioms 1.1–1.4 as for Nash game. Simultaneously and secretly, as in Nash Game, the first player selects the lines i^* of the matrices A and B, and the second player selects columns j^* of the same matrices. Unlike the Nash game, every player suspects that the opponent may know his choice, i.e. distinction of the Maximin game consists in player attitudes.

3.3.1 Maximin Solution Principle

Players compute the set of their pessimistic strategies.

$$\mathrm{MS_A} = \mathrm{Arg} \max_{i\in I} \min_{j\in J} a_{ij}$$

forms the set of pessimistic strategies of the first player.

$$\mathrm{MS_B} = \mathrm{Arg}\max_{j\in J}\min_{i\in I} b_{ij}$$

forms the set of pessimistic strategies of the second player.

Every element of Cartesian product $\mathrm{MS} = \mathrm{MS_A} \times \mathrm{MS_B}$ forms a maximin solution of Maximin Game $\mathrm{M\Gamma} = \mathrm{M}_2$.

3.3.2 Set of Maximin Solutions

$\mathrm{MS} = \mathrm{M_A} \times \mathrm{M_B}$ is the set of Maximin Solutions of the Maximin Game.

Proposition 3.3. *For matrices A and B the sets* NE, SE_{12}, SE_{21} *and* MS *are generally not identical.*

Proof. It is enough to mention that every Stackelberg game has Stackelberg equilibria and every Maximin game has the maximin solution, but the Nash game with the same matrices may do not have Nash equilibria. Even though the Nash game has equilibria, simple examples may be constructed which illustrate that Nash equilibrium is not identical with the Stackelberg equilibrium and the Maximin solution. □

3.3.3 Maximin Solution Existence

Proposition 3.4. *Every finite Maximin Game has maximin solutions.*

Proof. The proof follows from the finiteness of the strategy sets. □

3.4 Maximin-Nash Taxon

Maximin-Nash Taxon contains two elements:

$$\mathrm{MNT} = \{\Gamma_{00\,01}, \Gamma_{01\,00}\}\,.$$

Let us suppose, without loss of generality, that the players choose their strategies without knowing the opponent choice. However, one of them (and only one) has the belief that there is information leak about the chosen strategy. Let us denote such a game by $\mathrm{MN\Gamma} = \mathrm{MN}$ or $\mathrm{NM\Gamma} = \mathrm{NM}$.

3.4.1 Maximin-Nash Solution Principle

For defining the solution concept of such games, we can observe firstly that they may be seen as a constrained Nash Game $\Gamma_{00\,00}$, in which additionally must be applied the Maximin principle for the pessimistic player which suspects the corruption. So, for $\Gamma_{00\,01}$ we can define as the solution any element from:

$$\mathrm{NMS} = \mathrm{NE} \cap (\mathrm{I} \times \mathrm{MS_B}),$$

For $\Gamma_{01\,00}$ the solution is any element from:

$$\mathrm{MNS} = \mathrm{NE} \cap (\mathrm{MS_A} \times \mathrm{J}).$$

From the above definitions, it follows that a Maximin-Nash Solution is a Nash Equilibrium for which one of it's components (corresponding to the player which suspects corruption) is a Maximin strategy, too.

3.4.2 Set of Maximin-Nash Solutions

NMS is the set of solutions in game $\mathrm{NM} = \Gamma_{00\,01}$, and MNS is the set of solutions in game $\mathrm{MN} = \Gamma_{01\,00}$.

3.4.3 Maximin-Nash Solution Existence

Proposition 3.5. *If Maximin-Nash Game* MN *has a solution, then the Nash Game has a Nash equilibrium.*

Proof. The proof follows from the definition of the Maximin vs Nash solution. □

Generally, the reciprocal proposition is not true.

3.5 Optimum Taxon

Optimum Taxon is formed only by one element $\mathrm{OT} = \{\Gamma_{10\,10}\}$.

The player strategies are selected as it follows. Let us suppose, that the both players declare they play Nash game, but everyone cheats

and (by corruption and information leaks) knows the choices of the opponent. Such a game is denoted by $\mathrm{O}\Gamma = \mathrm{O}_2$.

To formalize this game we must highlight two pseudo-stages of the game. The first pseudo-stage when the players initially choose their strategies (i_0, j_0). And the second pseudo-stage when the players, after knowing (i_0, j_0), may choose their final strategies (i_1, j_1).

3.5.1 Optimum Profile

As everyone do not suspect opponent of cheating, but the both cheat, they play as followers, i.e., in the game O_2 the players act as followers.

The resulting profile is (i_1, j_1), where $i_1 \in \underset{i \in I}{\mathrm{Argmax}}\ a_{ij_0}$ and $j_1 \in \underset{j \in J}{\mathrm{Argmax}}\ b_{i_0 j}$.

As both i_1 and j_1 correspond to j_0 and i_0, correspondingly, the pair (i_1, j_1) is not a solution concept. It is a simple profile — an Optimum Profile.

3.5.2 Set of Optimum Profiles

For this game we can define only the set of Optimum Profiles:

$$\mathrm{O}_2\mathrm{P}(i_0, j_0) = \left(\underset{i \in I}{\mathrm{Argmax}}\ a_{ij_0},\ \underset{j \in J}{\mathrm{Argmax}}\ b_{i_0 j} \right).$$

3.5.3 Optimum Profiles Existence

We mentioned above that the OT taxon is based on Optimum Profile, which is generally not a solution concept. Nevertheless, we may conclude the Optimum Profile exists for every finite game, because of strategy finiteness.

3.6 Optimum-Nash Taxon

This Taxon has two symmetric elements $\mathrm{ONT} = \{\Gamma_{00\,10},\ \Gamma_{10\,00}\}$.

Let us suppose, the players declare they play Nash game, but one of them cheats and (by corruption and information leaks) knows the choice of the opponent. We denote this game by $\mathrm{ON}\Gamma = \mathrm{ON}$ or $\mathrm{NO}\Gamma = \mathrm{NO}$.

To formalize this game we highlight two stages of the game, as in the precedent case. The first stage is when the players initially choose their strategies (i_0, j_0). And the second stage is when the cheater changes his strategy as optimal to the opponent strategy. So, at the second stage the strategy (i_0, j_1) or (i_1, j_0) is realised.

3.6.1 Optimum-Nash Profile Principle

As in the case of the Maximin vs Nash Game, for defining the solution concept we can observe firstly that if they play Nash Game $\Gamma_{00\,00}$, i.e., they choose to play a Nash Equilibrium, the cheating is not convenient. For such games, Nash Equilibrium is the solution principle to apply. If the honest player does not play Nash Equilibrium Strategy, he may lose out comparably with the Nash Equilibrium. So, he plays Nash Equilibrium. In such case, for the cheater it is convenient to play a Nash Equilibrium strategy.

As a conclusion, this type of game may be thought as a Nash Game if the game has Nash equilibrium. If the game doesn't have Nash Equilibrium or it has many Nash Equilibria, the principle of the Optimum-Nash profile is applied. One of them chooses his strategy as in Nash game (leader). He can apply the maximin or the Stackelberg strategy of the leader. The opponent chooses his strategy as the last player in Stackelberg game (follower).

3.6.2 Set of Optimum-Nash Profiles

Evidently, if the honest player chooses the Nash Equilibrium Strategy, the set of solutions is identical to NES.

If the honest player chooses maximin strategy, e.g. the first player chooses one of the elements of $\mathrm{MS_A} = \mathrm{Arg}\max\limits_{i\in I}\min\limits_{j\in J} a_{ij}$, the opponent chooses every element from $J^* = \mathrm{Arg}\max\limits_{j\in J} b_{ij}$.

If the honest player chooses Stackelberg leader strategy, the opponent chooses the follower strategy. In such case, the ON Profile is a Stackelberg equilibrium.

3.6.3 Optimum-Nash Profile Existence

Based on the above, ON Profile exists for every ON game. It may be NE, SE, or a simple Maximin-Optimum Profile.

3.7 Optimum-Stackelberg Taxon

Optimum-Stackelberg Taxon contains two symmetric elements:

$$\mathrm{OST} = \{\Gamma_{10\,11}, \Gamma_{11\,10}\}\,.$$

Let us suppose that each player knows the opponent's chosen strategy, and only one of them knows additionally that the opponent knows his chosen strategy. So, the one which doesn't know that the opponent knows his chosen strategy, will simply select his strategy as optimal response to the opponent's strategy (he will play as an unconscious leader in a Stackelberg game), but the other (which knows additionally that the opponent knows his chosen strategy; player with the value of knowledge vector equal to '11') will know the opponent's reaction and will play as a follower in a Stackelberg game.

Proposition 3.6. *If every player knows a priory what information leaks he will use (he knows the values of his respective knowledge vector), then the player with the value of knowledge vector equal to '11' will play as a leader, and his opponent will play as a follower.*

It is not the case we consider.

3.7.1 Optimum-Stackelberg Solution Principle

If the first player doesn't suspect of information leaks to the second player ($\Gamma_{10\,11}$), but he knows the strategy j selected by the second player, then he chooses his strategy as an optimal response to j, i.e. $i^* \in I^* = \underset{i\in I}{\mathrm{Argmax}}\ a_{ij}$. Let us suppose that $\#I^* = 1$. The second

player knows that for his selected strategy j the first player will select i^*. He must select his strategy as an optimal response to the i^*, i.e. $j^* \in J^* = \underset{j \in J}{\text{Argmax}}\, b_{i^* j}$. So, the solution of $\Gamma_{10\,11}$ is (i^*, j^*).

By analogy, we can define the solution concept for $\Gamma_{11\,10}$. If the second player doesn't suspect of information leaks to the first player, but he knows the strategy i selected by the first player, then he chooses his strategy as an optimal response to i, i.e. $j^* \in J^* = \underset{j \in J}{\text{Argmax}}\, b_{ij}$. Let us suppose that $\#J^* = 1$. The first player knows that for his selected strategy i the second player will select j^*. He must select his strategy as an optimal response to the j^*, i.e. $i^* \in I^* = \underset{i \in I}{\text{Argmax}}\, a_{ij^*}$. So, the solution of $\Gamma_{11\,10}$ is (i^*, j^*).

Let us denote such a game by $\text{OS}\Gamma = \text{OS}$. The symmetric one is denoted as $\text{SO}\Gamma = \text{SO}$.

3.7.2 Set of Optimum-Stackelberg Solutions

Let us remember that to define solution concept we impose the cardinality of sets I^* and J^* to be 1. To define the set of solutions we must exclude this supposition. So, for $\Gamma_{10\,11}$ the set $I^* = \underset{i \in I}{\text{Argmax}}\, a_{ij}$ represents all optimal responses to strategy j of the second player. The second player knows/calculates this optimal response set. On its basis, by applying Maximin Principle he defines his set of Maximin Response $J^* = \underset{j \in J}{\text{Argmax}} \underset{i \in I^*}{\min}\, b_{ij}$. So the set of solutions of $\Gamma_{10\,11}$ is $I^* \times J^*$.

Analogically, for $\Gamma_{11\,10}$ the set $J^* = \underset{j \in J}{\text{Argmax}}\, a_{ij}$ represents all optimal responses to strategy i of the first player. The first player knows this optimal response set. On its base, by applying Maximin Principle he defines his set of Maximin Response $I^* = \underset{i \in I}{\text{Argmax}} \underset{j \in J^*}{\min}\, a_{ij}$. So the set of solutions of $\Gamma_{11\,10}$ is $I^* \times J^*$.

3.7.3 Optimum-Stackelberg Solution Existence

Proposition 3.7. *Every finite Optimum-Stackelberg Game* OS *has an Optimum-Stackelberg solution.*

Proof. The proof follows from the definition of the Optimum-Stackelberg Solution and the finiteness of the strategy sets. □

4 Taxonomy of two-matrix games with information leaks and three or more levels of knowledge

According to the above result, there are $4^3 = 64$ possible kinds of games with information leaks $\Gamma_{\gamma^A\gamma^B}$ in the case when the vectors of knowledge have three components $\gamma^A = \left(\gamma_1^A, \gamma_2^A, \gamma_3^A\right)$ and $\gamma^B = \left(\gamma_1^B, \gamma_2^B, \gamma_3^B\right)$ (information leaks may occur on 3 levels).

In this case and in the general case, is it enough to examine only seven taxa of games as for games with two level of knowledge or the number of taxa increases?

Theorem 4.1. *The number of taxa for two-matrix games with information leaks with the number of knowledge levels $l \geq 2$ does not depend on l.*

Proof. Firstly, let us observe that the abstract maximal number of possible taxa depends on number of solution principle applied by two players. In our case, we apply only four solution principle: Nash equilibrium, Stackelberg equilibrium, Maximin principle, and Optimum principle. So, the maximal number of taxa may be equal to 16. But, the rules of the games and knowledge possessed by players in the case of two levels of knowledge make up possible only seven taxa.

By induction, it is provable that this number of taxa remains unchanged for $l \geq 3$. □

5 Repeated two-matrix games with information leaks

If the games described above are considered as molecular games, then we can examine a series of molecular games on every stage of

which a molecular game is played. Evidently, such games are a simple consequence of the games of the seven types, corresponding to seven taxa highlighted above.

6 Taxonomy of multi-matrix games with information leaks and three or more levels of knowledge

In the case of three and more players, we can adjust the molecular approach and we can denote the games by their atoms (atom players). Evidently, the number of taxa for such games can increase. Can we present a taxonomy of such games? Can we present a scheme or a table of elementary or molecular games? We are going to answer soon to these questions.

7 Conclusions

Normal form games pretend to be a mathematical model of situations often met in reality. Actually, they formalize an essential part of real decision making situations and processes, but not ultimate. Real decision making situations are influenced by different factors, which may change the essence of the games and the solution principle applicable for their solving. It follows that the initial mathematical models must be modified, at least.

This **work in progress** presents a taxonomy of normal form games with information leaks. Every taxon contains the games solvable on the base of the same solution principle, highlighted in the name.

The games with arbitrary pseudo-levels and levels of knowledge, and the games with bribe are the subject of the work in progress.

References

[1] R. Bellman, *Dynamic Programming*, New Jersey, USA: Princeton University Press, 1957, XVIII+340 p.

[2] S.J. Brams, *Theory of Moves*, Cambridge: Cambridge University Press, 1994, XII+248 p.

[3] N. S. Kukushkin and V. V. Morozov, *Theory of Non-Antagonistic Games*, 2nd ed., Moscow: Moscow State University, 1984, 104 p. (in Russian)
[4] G. Leitmann, "On Generalized Stackelberg Strategies," *Journal of Optimization Theory and Applications*, vol. 26, 1978, pp. 637–648.
[5] J.F. Nash, "Noncooperative game," *Annals of Mathematics*, vol. 54, pp. 280–295, 1951.
[6] J. Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*, 2nd ed., Princeton, NJ, USA: Annals Princeton University Press, 1947, XVIII+625 p.
[7] H. Stackelberg, *Marktform und Gleichgewicht (Market Structure and Equilibrium)*, Vienna: Springer Verlag, 1934, XIV+134 p. (In German)
[8] V. Ungureanu, "Nash equilibrium set computing in finite extended games," *Computer Science Journal of Moldova*, vol. 14, no. 3 (42), pp. 345–365, 2006.
[9] V. Ungureanu, "Solution principles for simultaneous and sequential games mixture," *ROMAI Journal*, vol. 4, no. 1, pp. 225–242, 2008.
[10] V. Ungureanu, "Linear discrete-time Pareto-Nash-Stackelberg control problem and principles for its solving," *Computer Science Journal of Moldova*, vol. 21, no. 1 (61), pp. 65–85, 2013.
[11] V. Ungureanu, "Mathematical Theory of Pareto-Nash-Stackelberg Game-Control Models," in *Workshop on Applied Optimization Models and Computation*, Indian Statistical Institute, Delhi Centre, Program and Abstracts, pp. 55–66, Jan. 28-30, 2015.
[12] V. Ungureanu, "Linear Discrete Optimal and Stackelberg Control Processes with Echoes and Retroactive Future," in *2015 20th Int. Conf. Control Systems and Computer Science*, Bucharest, Romania, Proceedings, vol. 1, CSCS20 Main Track, pp. 5–9, May 27-29, 2015.
[13] V. Ungureanu and V. Lozan, "Linear discrete-time set-valued Pareto-Nash-Stackelberg control processes and their principles," *ROMAI Journal*, vol. 9, no. 1, pp. 185–198, 2013.

Valeriu Ungureanu, Received December 22, 2015

Valeriu Ungureanu
Moldova State University
Alexe Mateevici str., 60
Chişinău, MD-2009
Republic of Moldova
Phone: +(373)22 57 76 27
E–mail: v.ungureanu@ymail.com

Computer Science Journal of Moldova, vol.**24**, no.1(70), 2016

Optical Character Recognition Applied to Romanian Printed Texts of the 18th–20th Century*

Svetlana Cojocaru Alexandru Colesnicov
Ludmila Malahov Tudor Bumbu

Abstract

The paper discusses Optical Character Recognition (OCR) of historical texts of the 18th–20th century in the Romanian language using the Cyrillic script.

We differ three epochs (approximately, the 18th, 19th, and 20th centuries), with different usage of the Cyrillic alphabet in Romanian and, correspondingly, different approach to OCR.

We developed historical alphabets and sets of glyphs recognition templates specific for each epoch. The dictionaries in proper alphabets and orthographies were also created. In addition, virtual keyboards, fonts, transliteration utilities, etc. were developed.

The resulting technology and toolset permit successful recognition of historical Romanian texts in the Cyrillic script. After transliteration to the modern Latin script we obtain no-barrier access to historical documents.

1 Introduction

At present Internet is the most valuable deposit of information as it can be accessed and researched from any point. New information is prepared electronically and can be exposed effortlessly. If we want to expose historical documents, we are to digitize them.

*The results published in this article were presented on November 13, 2015 at the seminar dedicated to the memory of Prof. Iu. Rogojin

Sometimes we even can access graphical images of text pages but this form effectively restricts their availability. In particular, graphical presentation makes impossible full text search.

Full text search needs textual transcription of the historical source that can be got by OCR. It was statistically showed that full-text search and quick access to contents are very important for the users, because access to the original document becomes smoother [1].

Using OCR for historical documents started in early 1990-s and progressed in parallel with the advance of OCR tools. Since 2008 big OCR projects have started, like large-scale OCR of newspaper collections in the United Kingdom and Austria [1]. Modern projects referred to in [1] are IMPACT (**Imp**roving **Ac**cess to **T**ext) under FP7, and EOD under EU Culture 2007-2013 programme.

The conversion of historical documents from the paper to accessible and searchable electronic form meets two obstacles that are not fully cleared till now.

Nowadays state-of-the-art in OCR guarantees relatively good results only on modern texts. For historical typography, results are worse. There are several causes of it. Historical fonts vary even in one book, and are less readable. Old paper introduces speckles and distortions. Linguistic components and resources of modern systems don't often know the peculiarities of historical language variations. Each text yields its own specific mix of features and problems, which implies that the quality of OCR for historical documents may vary from perfect to almost unacceptable.

The second general problem is produced by the historical orthography and language changes. Most users of digital libraries don't have a good command of old language and desire to use the modern orthography at their search. Any word can have numerous variants in the historical documents because of language evolution and lack of orthography standardization. To get satisfactory replies at search, it is necessary to skip over the gap between modern and new orthography.

In different languages, availability of texts in original historical orthography differs. For example, Romanian Cyrillic script of the 18th century has glyphs that are not supported by most OCR programs.

There are such variants in accessibility of lexical resources at the search in the historical documents. Very subtle details should be taken into account because of the alphabet evolution; for example, the Romanian language in the middle of the 19th centure used more than 17 alphabet modifications.

This situation is usual for many languages and for many cases when scientists, students, publicists, writers, statesmen, etc. want to learn from original historical documents without intermediate interpretations. Therefore, national systems for no-barrier access to historical documents are necessary, being supported by historical lexical resources, proper OCR tools and tools for quick interpretation of new unknown texts. Such systems should become available for interested users of these cultural data.

The OCR of manuscripts is a specific challenge, and we will not discuss it here.

In the paper, we would discuss the factors defining the reliability of the OCR result, and the techniques permitting to enhance it by the example of printed historical Romanian texts of the 18th–20th century in the Cyrillic script. The following epochs were preliminary distinguished in the Cyrillic scripts for Romanian, using the principle "since the present and back centuries" (see details in [2]):

Epoch 1: the 2nd half of the 20th century, Moldavian SSR, Russian-based Cyrillic script.

Epoch 2: 1830–1860, the so-called transitional alphabets, mix of Romanian Cyrillic and Latin script.

Epoch 3: the early 19th century and back, the Romanian Cyrillic script.

For epoch 1, the problem seems to be almost solved, and we shortly discuss our achievements in Sec. 4. We concentrate our discussion mainly on the 2nd epoch (Sec. 5). We research also epoch 3; our results are presented in Sec. 6.

2 Production process

The following four stages form the process of producing the textual transcription of a printed historical document.

1. Digitization (scan) and image preprocessing.
2. OCR.
3. Text post-processing.
4. Quality evaluation.

For **scanning,** we recommend specialized book scanners, for example, Plustek OpticBook [3], and scan with at least 600 DPI resolution. The worst case is when we get already scanned source from some collection and cannot regulate its properties.

There are several freely available programs for **image preprocessing** like line straightening, image cleaning, converting to black-and-white. One of such programs is ScanTailor. A big collection of such tools is presented at [4]. In particular, Agora is an interesting tool that analyses blocks of text and images on pages.

OCR is a most complicated and error-prone stage. We tested several OCR systems and selected ABBYY FineReader (AFR) [2],[5]. The latest AFR versions include some image preprocessing but we recommend separate tools as more powerful and versatile. The OCR program performs segmentation of image to characters, and produces text comparing characters with patterns. Then the dictionaries for supposed languages can be used to check the spelling of resulted text and correct it. Training mode can be proposed when the user manually corrects text segmentation to glyphs and pattern-to-glyph mapping.

Post-processing of the text mainly includes manual correction of the OCR outcome, and extracting words to replenish the dictionary used at OCR. AFR permits some manual corrections in its output window before storing the resulting text. Allocation of textual blocks may also need correction, depending on purpose. For example, it is not necessary for full text search. The post-processing may continue up to full restoration of physical text appearance.

Quality assurance also depends on purpose of text processing. It can be done at several levels: the scan and dataset level, institutional

and the project consortium level. It is recommended to perform thorough post-evaluation and error spotting over the first produced text samples to ensure consistency in further production.

3 Factors affecting scan and OCR quality

The recognition quality depends on: the scan quality; the alphabet selection; the OCR engine training over specific texts; the availability of dictionary corresponding to the proper historical period. In its turn, scan quality is influenced by factors like: black-letter typefaces; irregular spacing between letters and words; changing font sizes; poor paper; inconsistent inking; speckles; distortion and other geometric deformations of text, non-straight lines; text strike-through. In the worst case, these may imply the manual correction of each page image, e.g., despeckling.

The case of color and negative (white letters on black or dark background) printing is also very difficult. AFR splits the image of each page to blocks that can be attributed as text, table, or image. This splitting is not always perfect; the manual correction may be necessary.

OCR quality may be affected by: alphabet diversity; mix of scripts; use of special characters, digraphs, ligatures; use of accents; use of historical vocabulary; poor vocabulary recognition.

The task of dictionary creation seems to be a true vicious circle as it supposes studying a lot of potential hardly accessible sources, and extracting data through language and script barriers.

4 Recognition of Moldavian Cyrillic script

Moldavian Cyrillic script was used in Moldavian ASSR and Moldavian SSR. It was based on the Russian alphabet with one additional letter "ӂ". The typography of that period permits to obtain good scans. The dictionary was produced from recognized books themselves using manual correction of words; it can be expanded from new books. Details are discussed in [2], [5]. The purpose of recognition was mainly

re-editing valuable books in modern Latin script; for this purpose, a transliteration utility was developed [5].

5 Recognition of transitional alphabets

Transitional alphabets were used in the Romanian typography since 1830 and until 1860–1870 [6]. They can be characterized by regular many-to-one mapping of old Romanian Cyrillic letters to the mix of Latin and Cyrillic letters. This mapping could be expanded further to modern Latin Romanian script; slightly different orthography makes an obstacle. The existence of such mapping distinguishes the old Romanian Cyrillic and transitional scripts from Moldavian Cyrillic script that cannot be ([5]) regularly mapped to the modern Latin script.

There were many different transitional scripts. Our impression is that different typographers used them depending on the existed stock of letterpunches, progressively replacing the Cyrillic letterpunches with the Latin ones whenever the former were worn. We can see different alphabets at the same year. Book [6, p. 115] shows a "record" example of 1840 where the title page was printed in four different scripts simultaneously (old Cyrillic, simplified Cyrillic, transitional, and Latin). Sources count up to 17 variants of the transitional scripts. This diversity makes a main problem at OCR of these documents.

We used two approaches to OCR of Romanian transitional scripts. The first approach is to reproduce the scanned text after OCR in its original glyphs. It is possible with the corresponding AFR configuring and training, and by providing the proper dictionary (Fig. 1, p. 112). It produces 7% of erroneous words.

The second approach was tested to solve the problem of alphabet variation. We rejected the principle of the exact text reconstruction after OCR. AFR permits to output the result in original glyphs or substitute them by any sequence of letters from the selected alphabet of recognition. This is called "ligatures" in AFR documentation. For AFR output, we invented a Latinized version of the alphabet that can be set in one-to-one mapping with any transitional alphabet. For example, both "т" (Cyrillic) and "t" (Latin) will be recognized as "t".

(a)

ЛȢІ ШЕКСПІР. 5

алцїї 'л аȣ̆ апроват кȣ ꙟmфокаре, фъръ ꙟndoiалъ,
пентрȣ къ bedea ꙟнтр'ꙟncȣл ȣn прілеж d'а аръта
cimцimѫнтȣл поетік ал ероȣлȣї лор, поате ꙟнкъ
de кѫnd се афла ꙟn леагън. Кьчї Аȣвреі історі-
сеще къ тѫпърȣл Віліаm, nenȣтѫnd съ се сȣбпȣе
кȣ плъчере daторіілор крȣнтȣлȣї съȣ̆ стат, кънта
а'л ꙟnпълца прin експресіїле ȣnȣї новіл cimцimѫnт,
пронȣnцѫnd кѫте ȣn помпос dicкȣрс de кѫте орї
тъіа вре о вітъ.

(b)

Л8Ĭ ШЕКСПІР 5

алції 'л а8 апробат к8 ꙟmфокаре, фъръ ꙟndoiалъ,
nentp8 къ bedea ꙟnтр-ꙟnc8л 8n прілеж d'а аръта
cimцimѫnт8л поетік ал еро8л8ĭ лор, поате ꙟnкъ
de кѫnd се афла ꙟn леагън. Кьчі А8брei історі-
сеще къ тѫпър8л Віліаm, nen8тѫnd съ се с8бп8с
к8 плъчере daторіілор кр8nт8л8ĭ съі стат, къ8та
а'л ꙟnпълца прin експресііле 8n8ĭ новіл cimцimѫnт,
прon8nцѫnd кѫте 8n помпос dicк8pc de кѫте орĭ
тъіа вре о вітъ.

Figure 1. Romanian transitional script (1848) after OCR: (a) source; (b) text

Because of one-to-one letter mapping, the exact reconstruction of the text from a book is achieved applying a simple letter substitution selecting the desired variant of the transitional alphabet. We are developing the corresponding conversion utility.

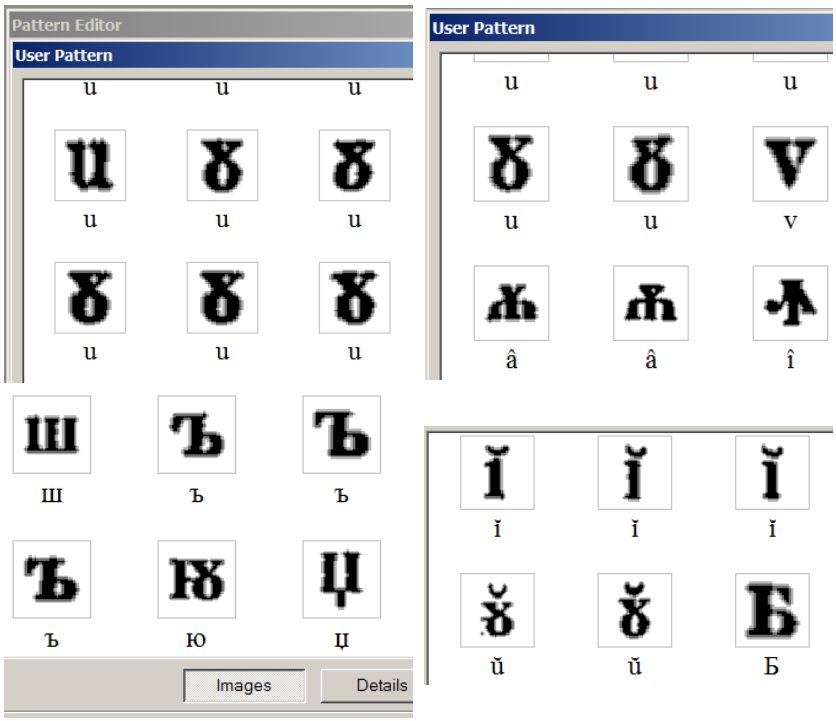


Figure 2. Part of AFR pattern collection for Romanian transitional alphabets with substitutions ("ligatures")

This approach also reduces drastically the volume of the dictionary. For example, "trekut" ("past", modern Latin script "trecut") in the recognition dictionary may check up to 16 variants obtaining by independently replacing t→т, r→p, k→к, u→ŭ).

This restriction of the recognition alphabet solves one small problem of interaction with AFR. AFR does not support arbitrary Unicode

glyphs in its dialogs and forms. Old Romanian letter "↑" was introduced in Unicode only after 2009. Standard system fonts do not contain some Romanian Cyrillic (and transitional) letters. As a result, we see in AFR empty boxes "□" instead of letters during training, alphabet formation, etc.

Work with ligatures also reduces errors to 4.8% (word level; see Sec. 6).

After training, we collected a set of glyph patterns for Romanian transitional alphabets. Part of this collection is shown in Fig. 2, p. 113.

Resuming, the OCR of Romanian transitional script should be performed as follows. Configure AFR with the corresponding "user language". Set the alphabet for this language from the corresponding string. Fill the recognition dictionary from the corresponding file. In the pattern editor of AFR, download recognition patterns from the corresponding file. After recognition, apply the utility and remap the result to the necessary variant of the transitional alphabet to restore the original glyphs.

You can also use the AFR output (before its remapping) to replenish the recognition dictionary. The recognition quality grows as the dictionary grows. We repeated recognition several times using the recognized text as new words source, with manual checking of the included words because of the absence of the historical lexicons.

6 Recognition of old Romanian Cyrillic script

AFR recognizes old Romanian Cyrillic Script. Small problems arose due to absence of necessary glyphs in system fonts, as it was already noted above. In fact, only three fonts in the whole world have old Romanian Cyrillic letters: Kliment, Unifont (bitmap font), and Everson Mono [7]–[9].

For example, the juridical text from 1786 was recognized with engine training and user supplied dictionary (Fig. 3, p. 117). This results in 4.5% errors (word level) with original glyphs and 3% errors with ligatures. We observed this effect with transitional scripts also.

This unexpected result is to be explained. The most likely reason

is that AFR skips some glyphs that are supposed to be recognized properly in the training mode. With original glyphs, AFR skips more glyphs, while, at the glyph substitution, AFR should train substituted glyphs and performs better training.

7 Conclusions

Digitization of historical texts includes their scanning and recognition; the latter was performed by ABBYY FineReader 12.

To use OCR for the Romanian Cyrillic script, we developed a set of historical alphabets and sets of glyphs templates, which are specific for each epoch. The spelling dictionaries in proper alphabets and orthographies were also created. Some auxiliary supporting tools like virtual keyboards, fonts, transliteration utilities, etc. were also developed.

Images were preprocessed with specific pre-OCR tools.

We have analyzed two approaches to recognition: using authentic glyphs, and using glyph substitution. The second approach solves the problem of diversity for transitional alphabet, and, due to some peculiarities of the AFR training mode, produces fewer errors.

OCR can dramatically increase the usability of digital libraries. The proposed solutions of the problems discussed in the paper can significantly impair the quality of the OCR outcomes. With it, full-text search and no-barrier access to digitized historical documents become possible.

References

[1] Historical Lexicon of Slovene:
Available: `http://www.digitization.eu/tools-resources/language-resources/1322-2/`

[2] E.Boian, C.Ciubotaru, S.Cojocaru, A.Colesnicov, L.Malahov. Digitization, recognition and conservation of the cultural and historical heritage. Academos, Nr. 1(32), 2014, pp. 61–68. ISSN 1857–0461. (In Romanian)

[3] Available: http://plustek.com/uk/products/opticbook-series/

[4] Available: http://www.digitisation.eu/tools-resources/tools-for-text-digitisation/

[5] C.Ciubotaru, S.Cojocaru, A.Colesnicov, V.Demidov, L.Malahov. Regeneration of Cultural Heritage: Problems Related to Moldavian Cyrillic Alphabet. Presented at The 11th International Conference "Linguistic Resources and Tools for Processing the Romanian Language". 26–27 November 2015. Eds: D.Gîfu, D.Trandabăţ, D.Cristea, D.Tufiş. pp. 177–184. ISSN 1843–911X. Available: http://consilr.info.uaic.ro/2015/Consilr_2015.pdf.

[6] S.Cazimir. The transitional alphabet. – Bucharest: Humanitas, 2006. ISSN 973–50–1401–7. (In Romanian)

[7] Available: http://kodeks.uni-bamberg.de/aksl/schrift/KlimentStd.htm

[8] Available: http://www.unifoundry.com/unifont.html

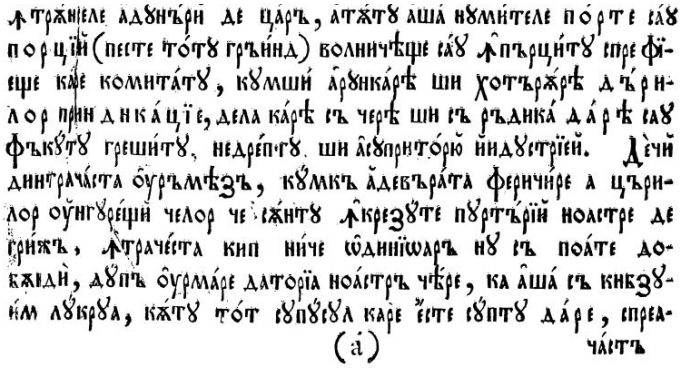
[9] Available: http://www.evertype.com/emono/

Svetlana Cojocaru, Alexandru Colesnicov,
Ludmila Malahov, Tudor Bumbu

Received March 14, 2016

Institute of Mathematics and Computer Science
Str. Academiei 5,
Chişinău, MD-2028,
Moldova
Phone: +373 22 72 59 82
E–mail: {svetlana.cojocaru,kae,mal}@math.md
bumbutudor10@gmail.com

(a)



(b)

Дечи диитрачаста оурѣмѣзъ, кѹмкъ адевърата феричире а църилор оугѹреции челор че сѫнтѹ ѧкрезѹте пѹртърїи ноастре де грижъ, ѧтрачеста кип ниче ѡдинїѡаръ нѹ съ поате добѫнди, дѹпъ оурмаре даторїа ноастръ чѣре , ка аша съ кнбзѹим лѹкрѹл, кѫтѹ тот сѹпѹсѹл каре есте сѹптѹ даре, спреа

(a) частъ

(c)

Дечи диитрачаста оурѣмѣзъ, кѹмкъ адевърата феричире а църилор оунгѹре^^^ челор че сѫнтѹ їкрезѹте пѹртърїи ноастре де грижъ, їтрачеста кип ниче ѡдинїѡаръ нѹ съ поате добѫнди , дѹпъ оурмаре даторїа ноастръ чѣре , ка аша съ кибзѹим лѹкрѹл, кѫтѹ тот сѹпѹсѹл каре есте сѹптѹ даре, спреа-

(a) частъ

Figure 3. Recognition of the juridical text of the 18th century in the Romanian Cyrillic script: (a) source; (b) original glyphs; (c) "ligatures"

Computer Science Journal of Moldova, vol.**24**, no.1(70), 2016

Large-scale executable biology using rapid integration of computational models*

Vladimir Rogojin, Ion Petre

Abstract

We plan to develop a systematic framework for assembling large-scale computational biological models by reusing and combining already existing modelling efforts. Our goal is to build a software platform that will compile large-scale biomodels through successive integrations of smaller modules. The modules can be arbitrary executable programs accompanied by a set of (I/O) interface variables; they may also have an internal structure (such as a metabolic network, interaction network, etc.) that yields its executable part in a well defined way. Firstly, wherever possible, modules with the compatible internal structure will be joined by combining their structure and by producing new larger executable modules (like, combining two metabolic networks, etc.). Then, irrespective of the underlying internal structure and modelling formalisms, all the modules will be integrated through connecting their overlapping interface variables. The resulting composed model will be regarded as an executable program itself and it will be simulated by running its submodules in parallel and synchronizing them via their I/O variables. This composed model in its turn can also act as a sub-module for some other even large composite model. The major goal of this project is to deliver a powerful large-scale modeling methodology for the primary use in the fields of Computational Systems Biology and Bioinformatics.

Keywords: computational biomodelling, multiscale whole-cell modeling, model integration, model refinement, executable biology.

*The results published in this article were presented on November 13, 2015 at the seminar dedicated to the memory of Prof. Iu. Rogojin

1 Introduction

Predictive and comprehensive models of biological cells are highly significant for the understanding and engineering biological systems. Such large-scale whole-cell models have the potential to direct experiments in molecular biology, facilitate computer-aided design and simulation in synthetic biology, and enhance personalized therapeutic methods.

The ultimate goal of our research is to develop and implement a generic technique that will allow for automatic building of custom large-scale multi-level comprehensive models that are able to describe accurately a biological phenomenon of interest with a desired level of details. The main idea here stays in reusing and integrating of already existing disparate modeling efforts into more representative, and therefore, more accurate, simulations of the targeted phenomenon. The models will be organized into a hierarchical structure by their levels of abstraction, so that a user can easily navigate through the hierarchy and choose the appropriate levels of details for the resulting integrated model.

In Computational Systems Biology a large number of individual teams of researchers target isolated subsystems and processes from living cells for modeling and simulation. In many cases, different researchers model independently overlapping, highly related or even the same cellular structures and processes. Moreover, those computational models focus on some particular cellular sub-systems while ignoring cross-talks with the other sub-systems and the other factors that also influence the sub-system under studies. A step forward towards building representative cellular models will be to join the disparate efforts of different modelers and build complex large-scale models via integration of already existing ones.

The greatest challenge here comes from the fact that different research teams come with their own concepts, abstraction levels, formalisms and methodology preferences as well as with their own technological limitations for their models. They contribute to the literature a mass of knowledge in the form of models and simulations in different formats, level of details and data types. One has to overcome those challenges in order to bring numerous modeling and simulation efforts into a significantly more representative larger comprehensive model that will capture all the features of initial models

and will agree still with all the respective initial experimental data.

Our method will involve searching for a common ground of integration between models, regardless of their formalisms and implementation. That common ground can be used to join models automatically into a single comprehensive simulation system. This will allow models to be integrated on several levels of details, depending on the required analysis. We will base our developments on a formal framework for integration of models of different formalisms, implementations and level of details that was presented in [1].

A software implementation of an integrated whole-cell model was presented in [2]. A model Mycoplasma genitalium was obtained through joining of 28 different independent computational submodels, where each submodel represents some specific cellular process [2]. The model described the dynamics of every molecule over the entire life cycle and accounts for the specific function of every annotated gene product. Each submodel was using formalism most appropriate to its functioning and existing knowledge. The submodels were assumed to run approximately independently on a short time scales. Simulations were performed by running through a loop in which the submodels were run independently at each time step but depended on the values of variables determined by the other submodels at the previous time step. The simulation software worked as a set of communicating running in parallel Matlab programs, where each program implements a submodel. The whole-cell software managed to predict the observed experimental data very accurately as well as it helped also to discover new previously unknown cellular behaviors. In the similar spirit, we will let for coordinated synchronized execution of different simulation instances within our simulation framework.

In our research, we propose a follow-up for the effort of developing large-scale biological model building techniques and model integration [1], [2]. We will represent a model as an executable simulating program that can be integrated with the other executable simulating programs standing for other models via a well defined API. We will allow for hierarchical type of model integration, that is, a number of complex models obtained due to previous integrations of simpler models can also be integrated together and form even more complex models.The practical outcome of our research will be a software modeling platform that will provide a systematic framework for integra-

tion and coordinated execution of different simulation instances, each simulating different parts of the biological phenomenon of interest. We are going to validate our methodology and computational platform through building a prototype for a whole-cell model of the life cycle of either yeast or E.coli, two of the most studied model organisms [3], [4]. Existing data in well-curated model database such as Biomodels [5] should be enough at least for a rough prototype of such a whole-cell model.

2 Background

A formal framework for integration of models of different formalisms, implementations and level of details was presented in [1]. The framework involves searching for a common computational ground between the models that will allow to integrate them, regardless of their formalism and implementation, across different methodologies into a single agent-based simulation system. Our abstract model descriptor will be based in particular on the concept of behavioral inclusion trees from [1].

Parameter fitting of a model to the experimental data is formulated as a global optimization problem. The goal is to tweak parameters of the model in such way that the predicted behavior of the model is as close as possible to the experimental data. The objective function here shows how far are the simulated data points from the experimental ones and takes as arguments the set of parameters to fit. The optimization goal is to find values of the parameters where the objective function reaches the global optimum (minimum). In particular, COPASI provides functionality for quantitative ODE-based model parameter estimation. It allows to fit both reaction kinetic rate parameters as well as initial concentrations of the metabolites.

Quantitative model refinement is an essential step in the model development cycle. Starting with a high level, abstract representation of a biological system, one often needs to add details to this representation to reflect changes in its constituent elements. Any such refinement step has two aspects: one structural and one quantitative. The structural aspect of the refinement defines an increase in the resolution of its representation, while the quantitative one specifies a numerical setup for the model that ensures its fit preservation at every refinement step. The refinement should be done so as to ensure the

preservation of the numerical properties of the model, such as its numerical fit and validation. In [6]–[10] there were presented methods for quantitative model refinement in a number of modeling frameworks, such as ODE-based models, rule-based models, Petri net models, guarded command language. We plan to use quantitative refinement when bringing different models to the same level of details.

A number of efforts towards integrating a plethora of publicly available molecular-scale experimental measurements that render intracellular molecule functions and interactions has facilitated data-driven large-scale model development [11]. For instance, in [11] there was developed a toolkit called Moksiskaan that can integrate information about the connections between genes, proteins, pathways, drugs and other biological entities from a large number of databases. As the result, one can obtain a comprehensive network model encompassing signaling, metabolic, gene regulatory, etc. intracellular relations. We will employ Moksiskaan for collecting a vast range of biological data needed for model construction and fitting.

Anduril [12] is an open source component-based workflow framework for scientific data analysis developed at the Computational Systems Biology Laboratory, University of Helsinki. Anduril also provides API that allows integrating rapidly various existing software tools and algorithms into a single data analysis pipeline. An Anduril pipeline comprises a set of interconnected executable programs (called components) with well-defined I/O ports, where an output port of a component may be connected to the input ports of some number of other components. (for example, see Figure 1). During execution of a pipeline, a component will be executed as soon as all the input data are provided by the up-stream components. After execution of the component, its results become available for the downstream components that can be executed in their turn as soon as all the necessary input data are provided for them. Anduril is highly scalable computational platform, it runs well both on desktop personal computers as well as on powerful supercomputers and clusters. Anduril can run multiple processes in parallel and its pipelines can be scaled for a distributed execution across the network. The size of Anduril pipelines can range from several component instances that could run in few seconds up to thousands of component instances requiring several days of execution. We will base our model integration platform on Anduril.

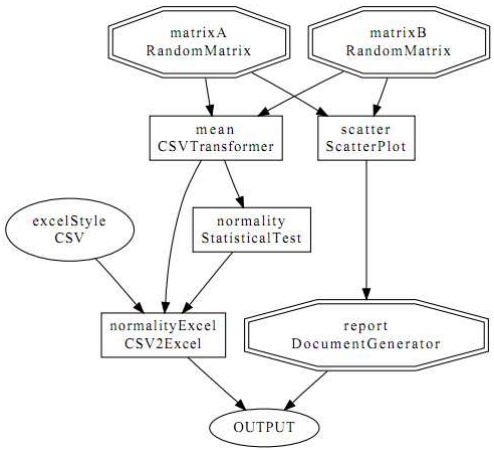


Figure 1. Example of an Anduril workflow (pipeline). Workflow is a set of tasks (component instances) organized in a directed network. Each component instance has some defined set of inputs and outputs, where outputs of an instance are connected to inputs of some other instances. Instances without any inputs import data into the workflow. Instances without any outputs normally output the workflow's computation results. For example, here *matrixA* and *matrixB* are instances of a component *RandomMatrix* that is a program generating a random matrix; instance *mean* of component *CSVTransformer* calculates mean for each row of matrices coming from *matrixA* and *matrixB* instances; instance *scatterPlot* generates a specific plot for the matrices from *matrixA* and *matrixB*, etc.

3 Integration of computational models

We are planning to obtain a software platform, that will allow a user to specify the input data (like set of initial models with their software implementations, model and environment parameters, abstraction levels, experimental data to fit to, etc.) for some target biological phenomenon and to query for prediction on some target features of the behavior of the modeled biological system (for instance, a time course for concentrations of some biochemical species, the time of cell growth in particular environmental conditions, etc.).

Here, we focus on developing model refinement and integration techniques for biological computational models. This project is a first step to-

wards building a system that, based on a user query, will allow collecting and reuse existing biological computational models in order to generate automatically custom case-specific comprehensive models for particular biological phenomenon at the desired abstraction level. We will develop a methodology that will allow minimizing or avoiding refitting the initial models to their corresponding experimental data while performing model refinement and model integration activities.

Our research is a follow-up for the effort of developing large-scale biological model building techniques and model integration [1],[2]. Particularly, we will allow for an arbitrary nature of a modeling formalism and its implementation. A model will be represented by an executable simulating program and can be integrated with the other executable simulating programs standing for other models via a well defined API. We will allow for hierarchical type of model integration, that is, a number of complex models obtained due to previous integrations of simpler models can also be integrated together and form even more complex models. We will allow for a high level of scalability and flexibility in the sense that a model can be tuned to a particular use case to fit the expected/desired behavior and to incorporate the desired level of details. In particular, we will employ here the previous practices related to numerical model parameter fitting [13] and model refinement [6]–[10].

We will address the following challenges:

1. Develop and implement intra-formalism integration methods for a number of different formalisms across different abstraction levels. That is, we are aiming to develop a systematic approach for automatic abstraction level adjustment via refinement and building a single model through joining a number of initial models, while working within the same formalism;

2. Develop and implement inter-formalism integration methods;

3. Deploy a generic (plug&play) platform that allows to “plug” computational models into the composite model, refine them and obtain the respective large-scale models and simulations.

4 Methods

We discuss here the methods that we are planning to apply in our research.

4.1 Abstract model descriptor

We will develop a methodology to describe in an abstract way models so that it will be possible to decide automatically how to join the models and how to instantiate the resulting integrated model. This *abstract model descriptor* will abstract from the model type (discrete or continuous, stochastic or deterministic), the modeling frameworks and models implementations.

The model descriptor will include such concepts as an *entity* (formalizes a real world physical object), a *process* (represents an activity involving one or more entities), a *variable* (a measurable/observable and/or affectable property of an entity or process). An entity may consist of a number of other entities, a process can be split in subprocesses, a variable may characterize/affect the current state of an entity or a process. Also, an entity may be an abstraction of some other entity and a process may be an abstraction of some other process.

We regard the abstract model descriptor as a digraph with annotated nodes and edges, Figure 2. In order to enable automatic integration of models basing on their descriptors, the user has to map nodes and edges of the descriptors to the corresponding components of the models or simulating instances. The exact way, how this mapping should be performed will depend on the particular type, formalism and implementation of each of the models.

4.2 Model integration and simulation

Here, the user should provide a set of initial models to integrate along with their abstract descriptors and mappings between the models and their descriptors. We remind, that we regard a model as an executable program or as a formal construct (a set of chemical reactions, signaling pathways, gene regulatory nets, etc.) together with its well-defined mapping onto an executable simulation instance.

We consider the following steps for integrating multiple user-provided models into one:

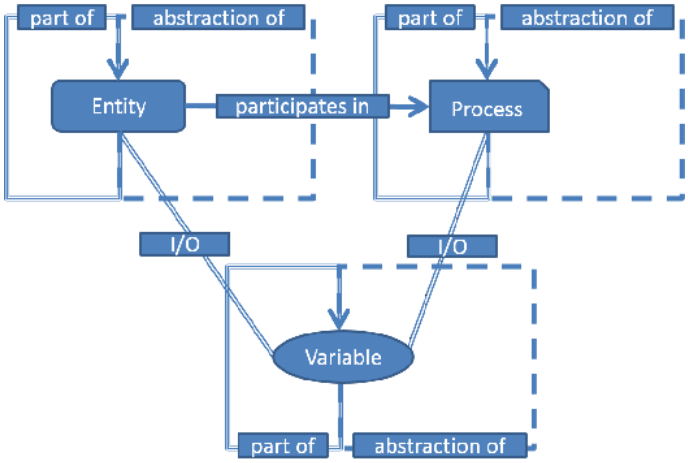


Figure 2. Schematic representation of abstract model descriptor. An entity can be a part of/abstraction of some other entity, a process can be a part of/abstraction of some other process and a variable can be a part of/abstraction of some other variable. The *part-of* relation is represented by a triple-line edge on the plot, and *abstraction-of* is represented as a dotted line. Entity can participate in a process. The relation *participates-in* is represented by solid line. A variable can be regarded as an interface to the state of an entity or process. The relation *input/output (I/O)* is represented by a double-line edge.

1. For all the models for which we have formally defined constructs, decide what constructs can be combined and how (for instance, two metabolic networks can be joined into one). We need to develop methods for joining models with the "compatible" internal structures into one model with its internal structure being a union of the original structures. See for example Figure 3. The internal structures will be joined through their overlapping components. Those overlapping components from two different structures that are at different levels of details will be brought to the same abstraction level via model refinement techniques [6]–[10]. The internal structure integration methodology and model refinement process strictly depend on the formalism being used, and we will develop the integration and refinement techniques for a

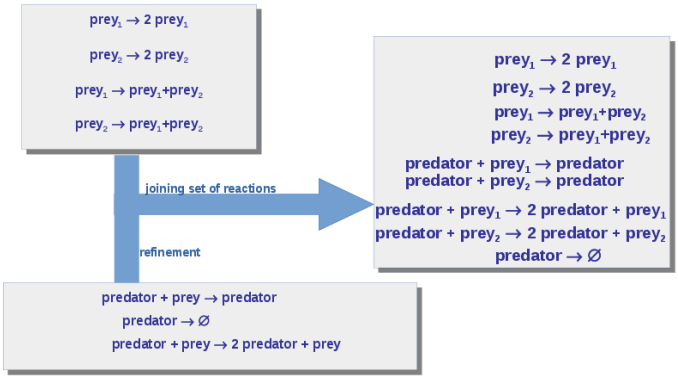


Figure 3. Example of integration of two models within the same formalism into one. The result: modified classic Lotka-Volterra Pray-Predator model [14] with two types of prey. The model is regarded in terms of chemical reactions (reaction rates and initial species concentrations are omitted here). One of the initial models describes dynamics (basically proliferation) of two types of prey. The other initial model describes predator's dynamics and its relation to prey specie (note that this model does not consider different types of prey species). During the integration process, the predators' model gets refined in order to include relations to the two types of prey species, then sets of reactions from the prey and the refined predator models get united.

number of classical formalisms (such as mass-action based ODEs, Petri nets, Boolean logic networks, etc.) separately. As the result of such integration of models with the compatible internal structures we will get one simulation instance that captures the behavior of the original models;

2. The user may provide a set of experimental data to fit the initial models. Fit the models to the user-provided time series data where necessary. Translate the resulting formal models into executable simulation instances;

3. Basing on the abstract model descriptors, decide connections between the executable simulation instances. The connections can be uni- or bi-directional. Here, a connection will also mean translation of an output

from a source simulation instance into an appropriate input format for the destination simulation instance. The type of translation will be decided automatically basing on the classes of the source and destination instances.

As the result, the integrated model will be the set of executable simulation instances with the scheme of their interconnections. The integrated model can be considered as a simulation instance itself and can be further integrated with the other models.

During the simulation the simulation instances will send to each other synchronizing messages that include the current states of some of their components as well as their contexts. The connection scheme defines which instance sends messages to which instances and what components states are included in each of the messages. In particular, the states of entities, processes and variables will be communicated. The abstract model descriptor of each of the initial models will be used to identify the overlapping or related components between different models. This information will be used to decide what components of the respective models should be refined, and what connections to form between instances. If in one of the models one identifies a set of components that is a refinement of a component from some other model, then one has to perform the respective refinement in the other model in order to obtain a set of the components that coincides exactly with the components from the former model. The refinement procedure depends on what type, formalism and model implementation is used in the respective model. When having same component (at the same level of abstraction) in two different models, one can form a connection between those two models that specifies in particular what components in these models are connected (also, probably, in which direction). One also associates to the connection a data transformer/adapter in order to transform the representation of states of the model component between the respective formalisms and data formats. See Figures 4, 5, 6 and 7 for an example of combining two simulation instances basing on their abstract descriptors.

The set of connections between a pair of instances defines what components communicate their states to the partner model within a message. Also, the context of one model that is being sent to the other model along with the states includes the simulated time point in order to synchronize two models.

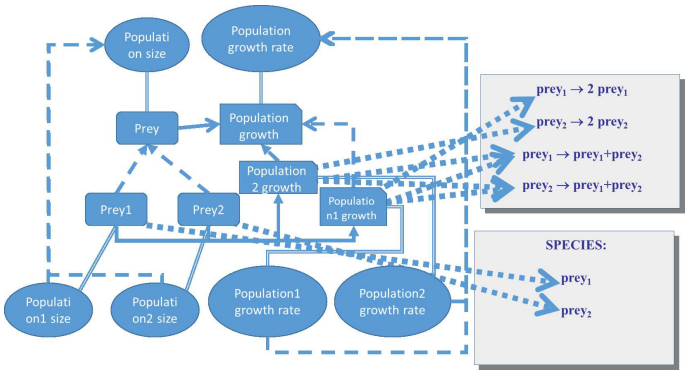


Figure 4. An abstract model descriptor for the prey population dynamics and its mapping to the internal model structure (set of reactions and species)

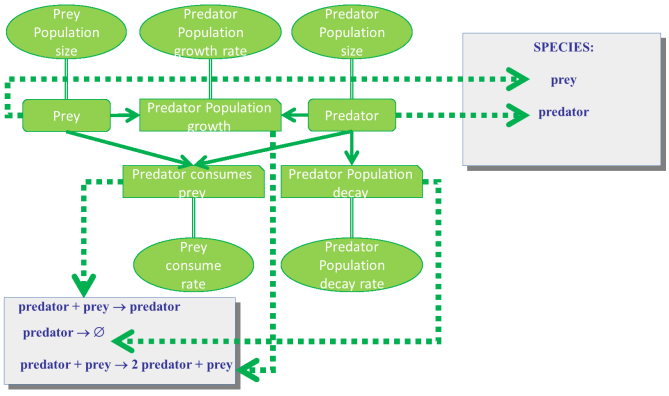


Figure 5. An abstract model descriptor for the predator population dynamics (including population growth/decay and prey consumption) and its mapping to the internal model structure

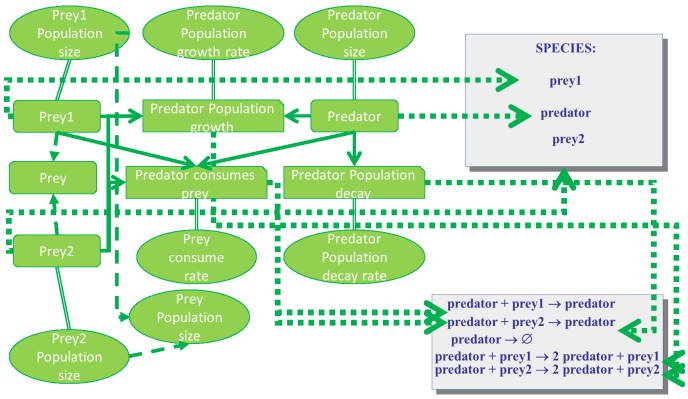


Figure 6. An abstract model descriptor for the predator population dynamics that was refined with two types of prey in order to have overlapping components between submodels from (a) and (b) at the same level of details

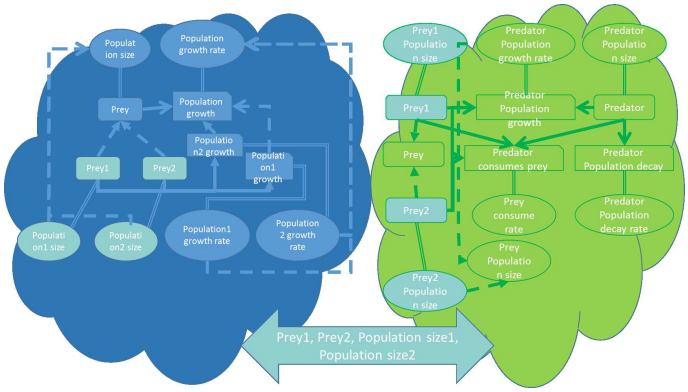


Figure 7. Basing on the abstract model descriptors for submodels from (a) and (b) it was decided to refine submodel for (b) into (c), and the overlapping components were detected between (a) and (c), that are “Prey1” and “Prey2” entities, “Population1 size” and “Population2 size” variables. Hereby, we establish connection between simulation instances corresponding to (a) and (c) that update each other with the states of “Prey1”, “Prey2”, “Population1 size” and “Population2 size” components.

We will develop an API which the simulation instances should handle in order to exchange messages with each other during the simulation. When a simulation instance receives a message, it should decide how to update its internal states of the respective components, especially in the case when the received message brings set of states contradicting the instance's own internal states.

4.3 Software implementation

We will produce a software platform that will construct a large-scale model through integration and refinement of the existing models.

Our platform will be built on top of the Anduril workflow framework due to the fact that Anduril provides systematic computational environment for rapid integration of existing computational tools and algorithms into an organized pipeline. An integrated model will be implemented as an Anduril pipeline, where components will represent the initial simulation instances, and connections between the components will represent connections between the respective models/instances. We also note, that an Anduril pipeline can serve as a component to be included in the other pipeline. In other words, Anduril will allow integrating a complex model within the other more complex model.

A simulation instance will be a software program wrapping the respective existing simulation software for the respective model, while handling the API to communicate to other instances and updating the local states of the instance according to the states received from the other instances. Also, the wrapper has to handle translation of states between different abstraction levels. In particular, we can incorporate COPASI as a dynamically linked library and access all of its simulation and parameter estimation functionality from our simulation instance program, that can "talk" to other instances and input/output data into/from the COPASI-simulated model from/to the other simulation instances. Petri nets can be simulated with S4 Snoopy modeling software that we are going to access from the respective simulation instance via well-define API of S4. Simulation instances for the Rule-Based modeling will use BioNetGen software as a command-line tool, or incorporate directly BioNetGen program code. We will use CellNetAnalyzer Matlab li-

brary in the simulation instance for Boolean networks. Generally speaking, any third-party simulation software can be incorporated into our integrative simulation framework via a wrapping program that handles our API for the communication with the other simulation instances. Schematically, this idea is represented in Figure 8.

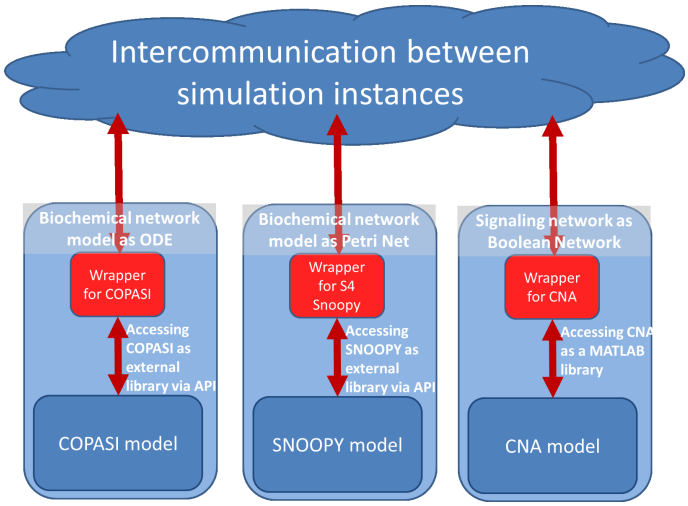


Figure 8. Scheme of the general organization of simulation instances. Each simulation instance simulates a model of some biological phenomena (for instance, biochemical networks, signaling networks, DNA transcription, RNA translation, etc.) probably in some well-defined formalism (for instance, mass-action based ODE model, Petri nets, Boolean networks, etc.) that is being implemented/simulated by some software (for instance, COPASI for ODE, Snoopy for Petri nets, CellNetAnalyzer for Boolean networks, etc.) A simulation instance consists of the original simulating software and the wrapper that translates between the simulation software and the messages for the other simulation instances.

5 Discussion

As the result, this project should deliver a powerful methodology for construction of large-scale models via integration of existing models and data.

The methodology will combine model refinement and integration techniques and assumes both data- and hypotheses-driven model construction approaches.

We believe that our research will contribute to the scientific modeling community through the development of a methodology for connecting different computational biological models irrespective of their formalisms, concepts and implementations. The platform that we are planning to implement will allow to build large-scale models and run comprehensive simulations by joining independent modeling and simulation efforts from a great number of research biomodeling projects. As the result, one will be able to build rapidly and easily large-scale custom models from a set of already existing models that will satisfy user-specific particular needs. In particular, the experimental biologist researchers should be the first to benefit from our platform due to the fact that comprehensive large-scale modeling can help in discovering new features of cellular processes without the need to conduct multiple expensive laboratory experiments. We may list among the other stakeholders bioinformaticians, medical-related researchers, synthetic biologists, bio-engineers, pharmacologists and any one else who needs to run expensive bio-experiments in-vivo or in-vitro.

References

[1] M. Patel and S. Nagl, "The role of model integration in complex systems modelling," *Understanding Complex Systems*, 2010. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-15603-8

[2] J. R. Karr, J. C. Sanghvi, D. N. Macklin, M. V. Gutschow, J. M. Jacobs, B. Bolival Jr., N. Assad-Garcia, J. I. Glass, and M. W. Covert, "A Whole-Cell Computational Model Predicts Phenotype from Genotype," *Cell*, vol. 150, no. 2, pp. 389–401, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0092867412007763

[3] L. Pray, "L. h. hartwell's yeast: A model organism for studying somatic mutations and cancer," *Nature Education*, vol. 1, no. 1, p. 183, 2008.

[4] P. S. Lee and K. H. Lee, "Escherichia colia model system that benefits from and contributes to the evolution of proteomics," *Biotechnology and Bioengineering*, vol. 84, no. 7, pp. 801–814, 2003. [Online]. Available: http://dx.doi.org/10.1002/bit.10848

[5] C. Li, M. Donizelli, N. Rodriguez, H. Dharuri, L. Endler, V. Chelliah, L. Li, E. He, A. Henry, M. I. Stefan, J. L. Snoep, M. Hucka, N. Le Novère, and C. Laibe, "Biomodels database: An enhanced, curated and annotated resource for published quantitative kinetic models," *BMC Systems Biology*, vol. 4, no. 1, p. 92, 2010. [Online]. Available: http://www.biomedcentral.com/1752-0509/4/92

[6] D.-E. Gratie, B. Iancu, S. Azimi, and I. Petre, "Quantitative model refinement in four different frameworks," in *From Action Systems to Distributed Systems*, L. Petre and E. Sekerinski, Eds. Taylor & Francis Group, 2016.

[7] D.-E. Gratie and I. Petre, "Hiding the combinatorial state space explosion of biomodels through colored petri nets," *Annals of University of Bucharest*, vol. LXI, pp. 23–41, 2014.

[8] C. Gratie and I. Petre, "Fit-preserving data refinement of mass-action reaction networks," in *Language, Life, Limits*, ser. Lecture Notes in Computer Science, A. Beckmann, E. Csuhaj-Varju, and K. Meer, Eds., vol. 8493. Springer, 2014, pp. 204 – 213.

[9] B. Iancu, D.-E. Gratie, S. Azimi, and I. Petre, "On the implementation of quantitative model refinement," in *Algorithms for Computational Biology*, ser. Lecture Notes in Computer Science, A.-H. Dediu, C. Martin-Vide, and B. Truthe, Eds., vol. 8542. Springer International Publishing, 2014, pp. 95–106.

[10] B. Iancu, E. Czeizler, E. Czeizler, and I. Petre, "Quantitative refinement of reaction models," *International Journal of Unconventional Computing*, vol. 8, no. 5-6, pp. 529–550, 2012.

[11] M. Laakso and S. Hautaniemi, "Integrative platform to translate gene sets to networks," *Bioinformatics*, vol. 26, pp. 1802–1803, 7 2010. [Online]. Available: http://dx.doi.org/10.1093/bioinformatics/btq277

[12] K. Ovaska, M. Laakso, S. Haapa-Paananen, R. Louhimo, P. Chen, V. Aittomki, E. Valo, J. Nunez-Fontarnau, V. Rantanen, S. Karinen, K. Nousiainen, A.-M. Lahesmaa-Korpinen, M. Miettinen, L. Saarinen, P. Kohonen, J. Wu, J. Westermarck, and S. Hautaniemi, "Large-scale data integration framework provides a comprehensive view on glioblastoma multiforme." *Genome medicine*, vol. 2, no. 9, pp. 65+, Sep. 2010. [Online]. Available: http://dx.doi.org/10.1186/gm186

[13] E. Czeizler, V. Rogojin, and I. Petre, "The phosphorylation of the heat shock factor as a modulator for the heat shock response," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 5, pp. 1326–1337, 2012.

[14] F. Hoppensteadt, "Predator-prey model," *Scholarpedia*, vol. 1, no. 10, p. 1563, 2006, revision 91666.

Vladimir Rogojin, Ion Petre

Received February 9, 2016

Computational Biomodelling Laboratory
Turku Centre for Computer Science, and
Department of Computer Science
Åbo Akademi University
Agora, Domkyrkotorget 3
20500 Åbo
Phone: +358 2 215 3361
E–mail: vrogojin@abo.fi, ipetre@abo.fi

Computer Science Journal of Moldova, vol.**24**, no.1(70), 2016

Advantages of application of unconventional computing to image processing and whence these advances come*

Lyudmila Burtseva

Abstract

This paper presents the advantages of P system computing based approach to solving image processing problems. Being significantly reduced relative to classic algorithms, P system computing based algorithms nevertheless produce better results and even resolve the problems just unsolvable for classic algorithms. Despite these evident advances, unconventional computing approaches remain less popular in practical image processing because their hardware is mostly virtual that supposes computing on simulator. This obstacle can be in perspective eliminated by direct implementation of P system computing based solutions on today's HPC hardware.

Keywords: Medical imaging, unconventional computing, P system, HPC, image processing.

1 Introduction

This paper intends to demonstrate the advantages of image processing problems solving by application of P system computing [1] that is the branch of unconventional computing.

Unconventional computing, also known as alternative computing, uses new or unusual devices instead of classical silicon ones. The most widely distributed branches of unconventional computing are: quantum

*The results published in this article were presented on November 13, 2015 at the seminar dedicated to the memory of Prof. Iu. Rogojin

computing, wetware computing, DNA computing, molecular computing, nanocomputing. The algorithms and programming methods of unconventional computing are conceptually different from the traditional ones. Essentially this difference springs from nature of unconventional computing that supposes computing in parallel on large amount of elementary devices. So, the algorithms and programming methods of unconventional computing suppose to be intrinsically parallel. Practically this aspect means that developers do not need tips-and-tricks to avoid exhaustive search (brute force).

Image processing – in particular medical imaging – is the most perspective user of unconventional computing algorithms. Unconventional computing based solutions are significantly reduced relative to classic based one; nevertheless they not only produce better results, but even resolve just classically unsolvable problems. Despite these evident advances, practical domains like medical imaging still rare use unconventional computing, because "hardware" is mostly virtual that supposes computing on simulator. This obstacle can be surmounted by recently proposed implementation of unconventional computing solutions on today's hardware – HPC (high performance computing) [2].

Advances of applications of unconventional computing to image processing mostly spring from easy obtaining of pixel neighborhood and possibility to use brute force algorithms without computational overdraft.

In the presented work P system computing – a particular branch of unconventional computing – is applied. This is bio-inspired paradigm, and its models, known as membrane systems or P systems, reproduce the membrane structure of the biological cell. Computing is performed by execution of rules of objects movement or transformation. The rules are applied in parallel. P systems demonstrate challenging applications in images processing and analysis. A variant of P system, called tissue-like P system, shows itself as suitable representation of image: pixel is mapped to cell in tissue [3].

In this work the above formalism and representation are applied to the solution of noisy medical image processing problems. The presented solution is used to illustrate the advantages of application of P system

computing.

2 Key features from which advantages come

As it was stated above, programming methods of unconventional computing are conceptually different from the traditional ones. Because of this principal difference the methods of unconventional computing often look tricky and advantages of their application are veiled.

To explane whence the advances come, the application of P system computing to solving practical problems of image processing is presented by illustrative example.

Firstly, P system computing workflow of this example has to be explained. Let us consider the cell as small computer. The substances which are transferred into the cell, could be considered as input. The substances which go out of the cell, could be considered as output. Some substance conversion done inside the cell is data processing. Another name of P system computing is membrane computing because the main aspect of this computing is the cell membrane that divides work area to inner cell and environment. The computation consists in execution of rules which act toward membrane providing:

- objects transition through membrane;
- objects conversion inside membrane or during transition.

The tissue P system is the abstract map of living tissue. The rectangle tissue of cells ordered in rows and columns is the natural representation of image where pixels are the cells and cell content is color value.

Having the brief scheme of P system computing we can show the springs of advantages of this computing application to image processing tasks.

From the very beginning the important aspect has to be stated. The today's application of unconventional computing is effective only for the cases which are hard solvable or even unresolvable by classical

methods. Of course this aspect is generated by absence of real hardware and restricted simulators capability.

According to the mentioned aspect, the test use case was selected. This set was selected from medical ultrasound images of gallbladder in SonaRes system database [4]. To prepare the test set of images, regions of interest (ROIs) are subtracted from full image. The resulted images present suitable test because of:

- hard noise make useless application of shape recognition methods;
- small size strikes application of gradient methods.

Moreover, SonaRes is the functioning system with its own retrieval module. For test purposes only those images were selected, the retrieval of which fails.

Formal definition of the problem is: to develop effective algorithms for extracting image-based features from ultrasonic images to be used for image matching and classification in diagnosis supporting process.

The problem solving by P system computing has two steps:

step (1) obtaining of features from test image and

step (2) retrieval the images with similar features from database of already processed images.

For presented use case the contours were chosen as images signatures because they can define the majority of gallbladder pathologies. At the **step (1)**, contours of image artifacts are traced applying P system based grayscale image region-based segmentation algorithm adopted from one proposed by [3]. The details of adoption are presented in our work [5]. The adoption was not a problem because of unconcern of P system based methods to image dimension or grayscale/color content of pixel representation. The proposed algorithm produces the acceptable segmentation result independently on image noisiness. Graphical-related basis of algorithm is the edge-based segmentation using the cross-like 4 points adjacency.

While P system based algorithm of segmentation is efficient and effective, the one currently used for retrieval at **step (2)** is reduced to comparison of contour obtained from pattern image with contours from the database. The problem of contours comparison is mostly finding of shape descriptor suitable to particular retrieval algorithm, in our case – shape descriptor by height function (length of perpendicular to the chosen axis) [6]. The coefficient of similarity is defined as ratio of "common" points obtained by retrieval to the total number of points in contour received from the pattern image.

These outlines of solving steps allow one to demonstrate the specific for P system computing features which make solving noisy image processing problems faster and simpler.

For the both steps, productive P system computing key features are the same:

1. Charge-less using of brute force algorithms makes available the per/pixel processing. Per/pixel processing in this case allows avoiding algorithmic tricks needed for retrieving features indistinguishable by another methods.

2. Tissue structure formalism gives easy access to neighborhood of every shape and complexity. For given problem such easy access provides the separation of grayscale levels that give as result the contour.

The key features application can be demonstrated in detail by presented example. In general, algorithms for the both steps consist in marking of target pixels by specific objects transferred from environment. Marking process is managed by rules. Algorithm executes identical rules for each pixel simultaneously applying massive parallelism feature of P system computing. Rules executing process applies another key feature – accessibility of any neighborhood of every cell of tissue structure.

Focusing on **step (1)** algorithm that is implemented efficiently, the springs of P system based solution advantages can be presented. When implementing segmentation, the rules are executed for each pixel in parallel but computation takes *only 2 steps*.

[I] First set of rules searches for border pixels until all half-border objects will be checked.

[II] Second set of rules repeats the same process converting temporary borders to final ones.

Specifically the native massive parallelism of P system computing makes the proposed algorithm so reduced.

3 Simulation results and perspective of direct HPC implementation

The demonstrated efficiency of P system based algorithm of course provokes the question how appropriate are the obtained results.

As stated above, today the only implementation of P system based solution can be done by simulator.

In this section the sufficiency of P system based image processing can be confirmed by extracts of simulation results. Sample group includes the results of presented use case and results of several tests on use cases taken from works where advanced classical algorithms were applied.

The used simulator is based on P lingua [7] framework. P lingua has full set of patterns for implementation of elements of membrane computing. While segmentation algorithm is the adopted one, its implementation was done in frame of presented work.

The use case for simulation is a set of real medical ultrasound images of gallbladder. Fig.1 shows the test results of cases of gallbladder polyps. In the case of this pathology, contours recognition plays the main role showing the difference between stones and polyps, which have straps with outer wall of gallbladder. P system based algorithm shows the acceptable results, while our attempts to get solutions by classical methods did not give the results at all.

Fig.2 shows the comparative results of test on medical images considered as hard processed in works [8] (left pair) and [9] (right pair).

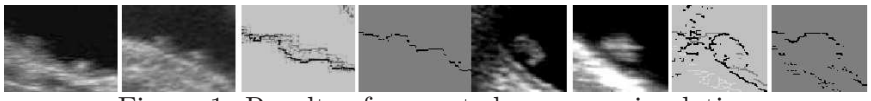


Figure 1. Results of presented use case simulation

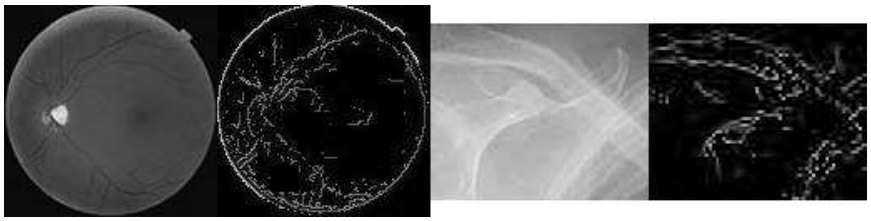


Figure 2. Results of comparative use cases simulation

There is a sufficient number of medical imaging problems solutions by unconventional computing algorithms, which are successfully implemented by simulators. But even working on parallel hardware, simulators cannot implement massive parallel processing of unconventional computing. Moreover, researchers does not trust the results obtained by simulator or just do not intend to spend the time for "ghost" computing. Therefore, despite the described advantages, medical imaging problems solutions based on unconventional computing still meet "non-existent hardware" obstacle. The mentioned "misunderstanding" challenge can be successfully answered by modern research of direct implementation of unconventional computing on today's hardware of high performance computing (HPC). Although this research is very fresh, the problems of such both important and suitable domain as image processing are certainly in list of considered tasks.

We recently have joined this research basing our decision on long experience of Spanish colleagues in CUDA implementation of P systems simulator [10]. Medical imaging use case that is currently the subject of our research shows itself as suitable and perspective application to develop the HPC implementation of presented P system based solution.

4 Conclusions and further work

P system based approach to image processing demonstrates effectiveness and productivity even being implemented on simulator. The solutions of test real-life problems of medical imaging domain promise successful further development of implementation of this approach on today's HPC hardware. The development supposed representing P system computing blocks as universal pattern from which the solution of particular task can be built. To extract suitable blocks more test problems of different domains have to be solved.

References

[1] Gh. Paun, *Membrane computing. An introduction* (Natural Computing Series). Springer, 2002, 412 p. Available: DOI: 10.1007/978-3-642-56196-2.

[2] M. Snir, "The future of supercomputing," in *Proceedings of the 28th ACM international conference on Supercomputing (ICS '14)*. ACM, New York, NY, USA, 2014, pp. 261–262.

[3] H. A. Christinal, D. Diaz-Pernil and P. Real, "Region-based segmentation of 2D and 3D images with tissue-like P systems," *Pattern Recognition Letters*, vol. 32, no. 16, pp. 2206–2212, Dec. 2011.

[4] L. Burtseva, S. Cojocaru, C. Gaindric, I. Secrieru, O. Popcova and D. Sologub, "SONARES a decision support system in ultrasound investigations," *Computer Science Journal of Moldova*, vol.15, no.2(44), pp. 153–178, 2007.

[5] Yu. Rogozhin, A. Alhazov, L. Burtseva, S. Cojocaru, A. Colesnicov and L. Malahov, "Solving Problems in Various Domains by Hybrid Models of High Performance Computations," *Computer Science Journal of Moldova*, vol.22, no.1(64), pp. 3–20, 2014.

[6] Junwei Wang, Xiang Bai, Xinge You, Wenyu Liu, and Longin Jan Latecki, "Shape matching and classification using height functions," *Pattern Recogn. Lett.*, vol. 33, no. 2, pp.134–143, Jan. 2012.

[7] M. Garcia-Quismondo, R. Gutierrez-Escudero, I. Perez-Hurtado, M. J. Perez-Jimenez and A. Riscos-Nunez, "An Overview of P-Lingua 2.0," in *Membrane Computing* (Lecture Notes in Computer Science, vol. 5957), Springer Berlin Heidelberg, 2009, pp. 264–288. Available: DOI: 10.1007/978-3-642-11467-0_20.

[8] S. Mudigonda, F. Oloumi, K. M. Katta and R. M. Rangayyan, "Fractal analysis of neovascularization due to diabetic retinopathy in retinal fundus images," in *E-Health and Bioengineering Conference (EHB)*, Iasi, 2015, pp. 1–4.

[9] V. Kovalev, A. Prus and P. Vankevich, "Mining lung shape from X-ray images," in *Machine Learning and Data Mining in Pattern Recognition* (Lecture Notes in Computer Science, vol. 5632), Springer Berlin Heidelberg, 2009, pp. 554–568. Available: DOI: 10.1007/978-3-642-03070-3_42.

[10] M.A. Martínez-del-Amor, L.F. Macías-Ramos, L. Valencia-Cabrera, A. Riscos-Núñez and M.J. Pérez-Jiménez, "Accelerated Simulation of P Systems on the GPU: A Survey," *Communications in Computer and Information Science*, vol. 472, pp. 308–312, 2014.

Lyudmila Burtseva

Received April 4, 2016

Institute of Mathematics and Computer Science
Academy of Sciences of Moldova
5 Academiei str., Chişinău, MD-2028, Moldova
E–mail: luburtseva@gmail.com