UDC 519.85 https://doi.org/10.15407/pp2022.03-04.117

ALGEBRAIC APPROACH TO THE ANALYSIS OF LEGAL DOCUMENTS

Oleksandr Letychevskyi, Volodymyr Peschanenko, Maksym Poltorackiy

У даному дослідженні розглядаються проблеми аналізу статей законодавства та їх відповідності правовим документам. Ми пропонуємо використання алгебраїчного підходу для формальної верифікації юридичних текстів, які представляються у вигляді специфікації алгебри поведінки. Наявність формального представлення юридичних текстів дозволяє застосовувати алгебраїчні методи, такі як символічне моделювання, автоматичне доведення тверджень і алгебраїчне зіставлення. Підхід реалізовано для україномовних правових документів для виявлення невідповідності, неповноти та підтвердження відповідності. Було здійснено аналіз законодавчих актів для Податкового кодексу і наведено приклади невідповідності деяких тверджень. Проведено ще один експеримент для перевірки відповідності судових актів, договорів, ліцензій, нових законів, актів про оподаткування статтям законодавства з використанням методів алгебраїчного зіставлення. Формалізація юридичних документів, таких як судові рішення, має здійснюватися повністю автоматично, а існуюча база даних із такими документами має забезпечувати можливість використання технологій big data та машинного навчання. У цьому дослідженні ми розглядаємо використання алгебраїчного підходу в аналізі законодавчих вимог і правових документів у межах системи інсерційного моделювання. Предметом дослідження є створені законодавчою владою закони та юридичні документи, такі як судові рішення, угоди, ліцензії та судові справи. Проблемою, яка розглядається, є аналіз нормативно-правових документів на відповідність закону та аналіз статей закону, перевірка на невідповідності, неповноту та інші властивості. У цій роботі ми детально розглядаємо кожен елемент технології, описуємо методику та результати експериментів.

Ключові слова: інтелектуальний аналіз юридичного тексту, алгебра поведінки, символьне моделювання, формалізація, алгебраїчне зіставлення, неповнота

In this study, problems regarding the analysis of law articles and their conformance to legal documents are considered. The algebraic approach is used for the formal verification of legal texts that is presented as specification of behavior algebra. Having a formal presentation of legal texts allows for the application of algebraic methods, such as symbolic modeling, automatic proving of statements and algebraic matching. The approach was implemented for Ukrainian-language legal documents to detect inconsistency, incompleteness, and prove conformance. The analysis of legal texts has been implemented for Tax code and examples of inconsistency of some statements were demonstrated in the paper. Another experiment has been performed for checking of conformance of court statements, agreements, licenses, new laws, taxation acts to the articles of law with usage of methods of algebraic matching. The formalization of legal documents, like court verdicts, shall be implemented fully automatically, and the existing database with such documents shall provide the possibility to use big data technologies and machine learning. In this study, we consider the use of the algebraic approach in the analysis of legal requirements and law artifacts within the scope of the Insertion Modeling System (IMS). The subject of this research is the laws created by the legislature and the artifacts of legal activity, such as lawsuit decisions, agreements, licenses, and juridical cases. The problem to be considered is the analysis of legal documents for conformance with the law and the analysis of the law's articles, checking for inconsistencies, incompleteness, and other interested properties. In this paper, we consider every element of the technology in detail, and we describe the methods and results of the experiments.

Keywords: legal text mining, behavior algebra, symbolic modeling, formalization, algebraic matching, incompleteness

Introduction

There are many problems regarding legal knowledge processing, such as the analysis of legal documents for their conformance to the law and the analysis of the law itself. These problems face many difficulties, such as the mining of knowledge from natural text, the formal presentation of legal requirements, and the formal methods used for processing.

All such analyses can be implemented using algebraic methods. However, with these methods, the following problems arise: 1) How do we extract the formal presentation from the natural text that presents juridical artifacts, 2) How do we formalize the articles of law, and 3) How do we match the juridical cases with articles of law?

We consider the technology, which includes the IMS and front-end subsystems, for the processing of external data, as it consists of two parts. The first part involves the automatic extraction of the structure and the necessary data from the artifacts. The second part is the use of formal methods for analysis. Our contribution to this technology is the creation of the formal methods of legal information analysis and the presentation of knowledge in algebraic specifications that are input into formal methods.

The scheme below shows the total scheme for legal document processing.

The above figure considers two different activities: the formalization and analysis of the law and the check for conformity of legal artifacts with rules of the law. The articles of law present a set of textual statements that shall be translated into suitable specifications, especially algebraic specifications, for further processing.

The formalization of the law is a manual or semi-manual process and is more time-consuming. It demands the cooperation of legal experts and mathematicians, and the results are the specifications of algebraic behavior notations. The analysis of the specifications involves the use of a deductive system and different solvers. It ends with the verdict, which includes the issues and findings regarding inconsistencies in the law or non-deterministic situations and incompleteness that define the gap in the texts of the law.

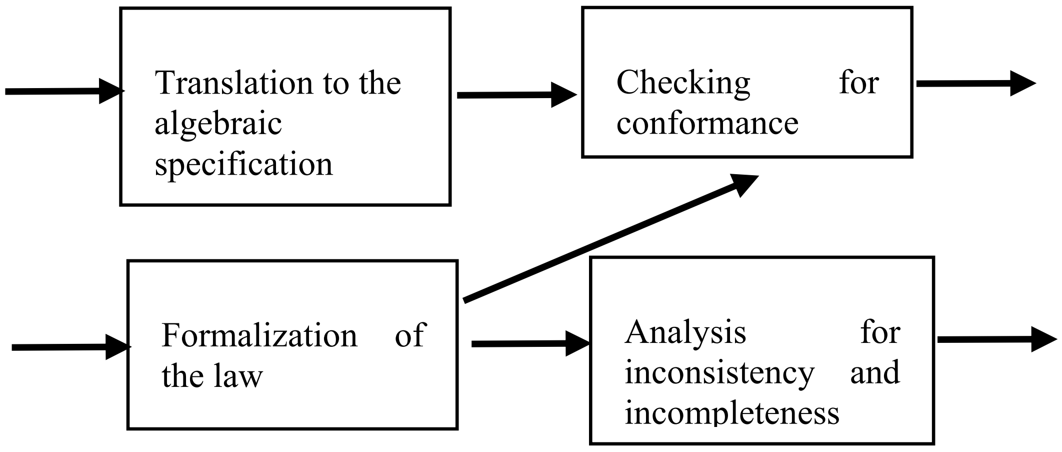

Translation to the algebraic specification
Checking for conformance
Formalization of the law
Analysis for inconsistency and incompleteness

Figure 1. Processing and analysis of legal documents (NT—natural test, AS—algebraic specifications)

The other activity, checking for correspondence of lawsuit artifacts pertaining to the law, anticipates the automatic transformation of artifacts into algebraic specifications and matches the specifications with formalized law. The result is a verdict that defines the issues. One example of this is the matching of trial decisions with the law to obtain evidence to prove or refute said decisions.

Related Works

During the last decade, the concept of an open data search engine has been developed. One such system is Openlaws.eu [2], where developers created a platform for the acquisition of a great array of legal knowledge, which provides users with the opportunity to deal with different types of regulatory and legal documents. It is necessary to mention the attempts to present the legal knowledge as XML models, as this provides the usage of the XQuery request language [3] for resolving the problem of data processing. There are a number of special XML standards that are used, such as LexDania (Denmark) [4], CHLexML (Swiss) [5], eLaw (Austria) [6], and SDU BWB (Netherlands) [7]. MET-ALex [8] is the open standard for the markup of legal documents.

The ontologies are the most popular kind of knowledge presentation for the modeling of different subject domains. The following are the popular ontologies used in the modeling of legal documents: FOLaw [9], LRI-Core [10], DOLCE + CLO [11], LKIF-Core [12], and UFO-L [13].

However, there are a number of different formal methods used in the analysis of legal texts on the basis of ontology. One of the attempts at automated model creation in OWL format is presented in [14]. The OWL-file presents the ontology, and usage of the OWL format provides the possibility to apply reasoning methods like Pellet [15]. The presentation of legal knowledge in the formal language SWRL [16], which is given as a set of Horn clauses, provides the usage of logical reasoning.

The UML language and its extension, OCL, is one of the tools for modeling and verifying legal documents [17]. It provides the detection of incompleteness in legal documents, and for these purposes, a special OCL solver [18] is needed.

Deontic logic [19] defines the elements of formalization via the terms “permissions” and “obligations” and defines some calculus over it. The only difference from logical programming in the proposed implementation of deontic reasoning is the use of fuzzy categories. The problem of using information technology for legal regulation is not new for the European Union (EU), so there are legal policy modeling systems in the EU like EUROMOD [20], POLIMOD [21], and Shyster [22].

The systems described above, to some extent, solve the problem of modelling regulations, but it should be noted that these approaches cannot effectively solve the problem of verifying the logical consistency of the legal framework. The XML presentation and ontologies are good enough for the search activity and exchange of legal data. The formal methods are used for proving static statements and do not take into account the behavior of agents for proving dynamic properties. Therefore, it is necessary to use more powerful technology with advanced reasoning and algebraic approaches.

Presentation of Legal Knowledge as Algebraic Specifications

The theory of agents and environmental interactions forms the base for the formalization of legal knowledge. This theory was introduced by David Gilbert and Alexander Letichevsky in [23].

Agents are entities that can change their state. They consist of two components: an informational component that defines the state of the agent and is presented by a set of the agents’ attributes, and a behavioral component that presents a set of all possible scenarios of behavior when the changing of an agent’s states is possible.

The functioning and interaction of agents are performed through the environment. The agents that interact in some environments are the agents inserted into the environment. The theory that studies the behavior of agents and environments is the insertion modeling theory. The environment is an entity that is similar to the agents, and it also has the attributes and possible behaviors of the agents. The information about the agents is accessible to the environment; however, the agent has no information about the other agents, and they interact via messages or shared memory in the environment.

Considering the legal entities that interact within the scope of some juridical activities, we can define agents as state institutions, private persons, economic subjects, companies, and taxation offices. Their interaction, in some environments, is defined by the law in the scope of some agreement or lawsuit.

Every agent can be formalized as an entity with a set of attributes that define its state. For example, the subject of some agreements has an attribute that defines their financial activity and tax payment. Its behavior shall be in compliance with the set of rules that are defined by law.

On the other hand, the rules that define the behavior of the interacting agents are a formalized set of formal descriptions in the language of behavior algebra.

Within the scope of the insertion modeling method, behavior algebra was introduced by D. Gilbert and A. Letichevsky in 1997 [24]. Behavior algebra is a two-sorted universal algebra. The main sort is a set of behaviors, and the second sort is a set of actions. The algebra has two operations, three terminal constants, and approximation relations. The operations are the prefixing a.u (where a is an action and u is a behavior) and non-deterministic choice behaviors u + v (associative, commutative, and idempotent operations on the set of behaviors). The terminal constants are successful termination Δ, deadlock 0, and unknown behavior ⊥. The approximation relation ⊑ is a partial order on the set of behaviors with minimal element ⊥. Examples of behavior expressions include the following:

$$B0 = a1.a2.B1 + a3.B2,$$

$$B1 = a4.\Delta,$$

$$B2 = \cdots$$

These imply that behavior, B0, could be interpreted as a sequence of actions a1, a2, and behavior B1, or as action a3 followed by behavior B2. Behavior B1 will finish the possible scenario after action a4.

Behavior algebra is also enriched by two operations: the parallel (||) and sequential (;) compositions of behaviors.

Furthermore, behavior algebra expressions can be presented in graphical form. The formalization language is called UCM (Use Case Maps), and it is standardized as part of URN (User Requirements Notation) upon ITU-T recommendation (Z.151) [25], which provides a scenario-based approach to requirements specification. Examples of UCM specifications are given in Fig. 2.

An agent changes its state under some conditions formed by the values of attributes. Every agent's action defines a triple: $B = < P, A, S >$, where P is a precondition of the action presented as a formula in some basic logic language, S is a postcondition, and A is a process that visualizes an agent's transition. As a basic logical language, we consider a set of formulas of first-order logic over polynomial arithmetic. For different subject domains, other theories can be used in formulas, for example, the theory of enumerated types, bitwise algebra, the theory of lists and queues, and the theory of sets. As a whole, the semantics of an action means that an agent could change its state if the precondition is true, and the state will change correspondingly to the postcondition, which is also a formula of the basic language. The postcondition can also contain an assignment statement. The process of action depends on the subject domain and illustrates the sequence of the action application.

Insertion Modeling System

The insertion modeling system is intended for the creation of models in terms of behavior algebra and runs the application (fig..2) that realizes the formal methods over behavior algebra specifications.

To define an agent in an environment, we need to define its type—that is, how it is formed by a set of attributes. The input of the agent's attributes is implemented via the client program (Fig. 2).

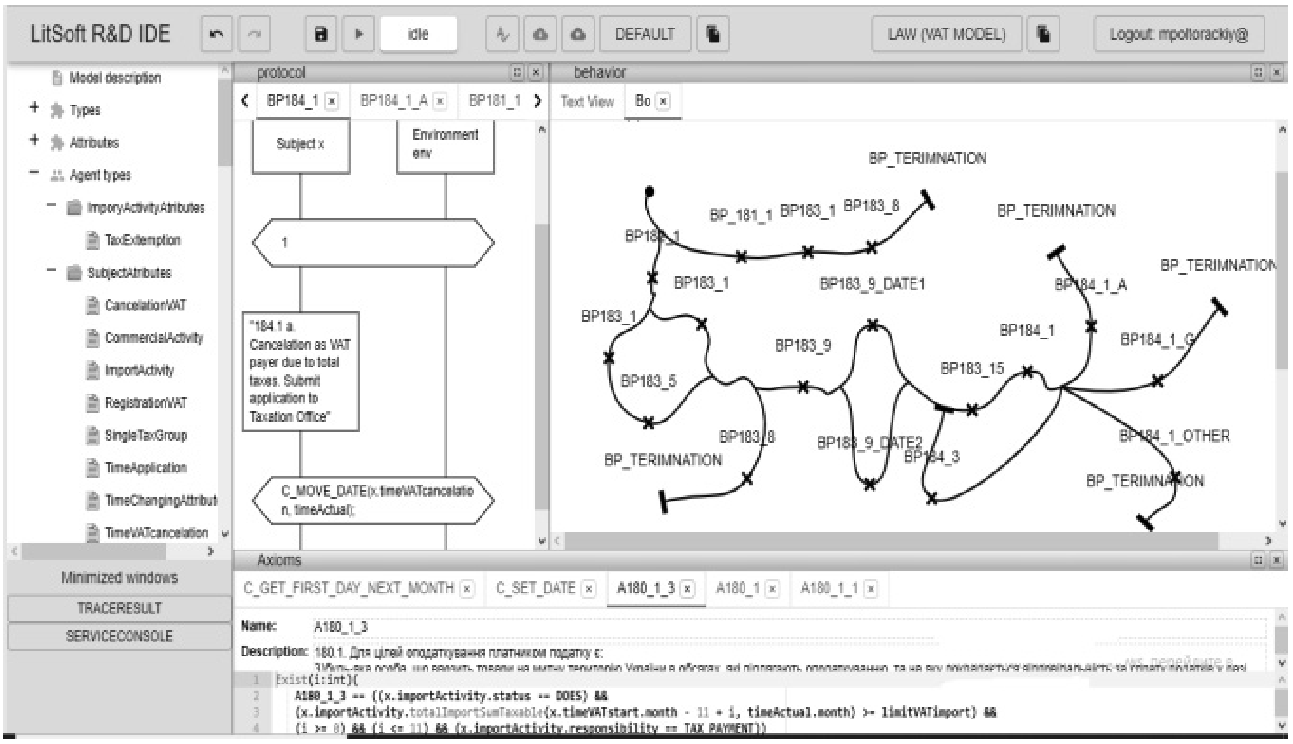
We can also input the initial environment formula or the initial values of attributes, such as axioms, over the attributes. The action that contains the precondition, process component, and postcondition can be input as the MSC (Message Sequence Chart) diagram.

The behavior of the agent can be presented as the formula in the behavior algebra specifications or as the UCM specifications.

Formal methods of verification in IMS anticipate the checking of the properties of the system. The properties could be classic, such as the absence of deadlocks and non-determinisms, or subject-domain-specific defined safety or liveness conditions. We can prove the reachability of demanded properties by using the symbolic modeling of behavior algebra expressions and actions application developed within the scope of IMS.

Forward symbolic modeling is a form of scenario generation from the initial state of a behavior algebra model to the sought-for property.

Backward symbolic modeling is a form of trace model scenario generation from a sought-for property given as a formula of an agent state to an initial state of algebraic behavior. Such a method is good for detecting a safety violation state. If we start from a safety violation state and traverse it, it will lead us to deadlocks; this property then becomes unreachable. Scenario generation can be performed with the given coverage for selected agents or with certain conditions.



Name: A180_1_3

Description: 180.1. Для цілей оподаткування платником податку є:
3) [illegible]

```
Exist(i:int){
  A180_1_3 == ((x.importActivity.status == DOES) &&
  (x.importActivity.totalImportSumTaxable(x.timeVATstart.month - 11 + i, timeActual.month) >= limitVATimport) &&
  (i >= 0) && (i <= 11) && (x.importActivity.responsibility == TAX_PAYMENT))
```

Figure 2. The window of IMS Client with the project of formalized articles from the tax code.

Algebraic matching is intended for checking for compliance of the given pattern to the model. It uses the formal methods of resolving the behavior equations, rewriting techniques, and proving that the local properties use the invariants generation.

Usually, when proving or analyzing the properties of the traversal of the possible states, the model can face combinatorial explosion. To overcome this issue, IMS uses methods of approximation, abstract interpretation, slicing, and some heuristic algorithms.

In IMS, some properties can be proved statically. It can be performed with the different solving/proving machines that were developed in the scope of IMS, or with the third-party tools like Microsoft Z3 solving machines, CVC4, and other modern systems.

Formalization of the Tax Code

We consider the version of the tax code of Ukraine and its formalization in the behavior algebra specifications [26]. First, we define the types of agents that interact with one another. One of these types is Individual, which has a set of attributes that can define individuals as taxpayers.

Presenting the attributes, we can define the statement when an individual will be a taxpayer, which corresponds to Article 180 of the tax code of Ukraine.

"Article 180. Taxpayers

180.1. For tax purposes, the taxpayer shall be:

1) any person carrying out or intending to carry out economic activity and registered as a taxpayer at own voluntary discretion according to the procedure stipulated by Article 183 hereof;

2) any person registered or subject to registration as a taxpayer;

3) any person importing goods into the customs territory of Ukraine in taxable amounts and charged with the payment of taxes in case of transfer of goods across the customs border of Ukraine according to the Customs Code of Ukraine" [27].

We created the statements, or axioms, that correspond to every item of Article 180.1 that can be defined in IMS, as shown below:

$$TAXPAYER(x) \Leftrightarrow Axiom\_1(x) \,||\, Axiom\_2(x) \,||\, Axiom\_3(x)$$

$TAXPAYER(x)$ is the predicate that has as parameter, the agent name, and defines agent *x* as a taxpayer if it is true. The axiom will be presented as follows:

$$Axiom\_1(x) \Leftrightarrow (x.activityStatus\ ==\ INTEND \,||\, x.activityStatus\ ==\ CARRY\_OUT)\ \&\&\ x.registration\ =\ TAX\_PAYER\_183$$

The first axiom is defined by the attribute of agent x, which has an enumerated type consisting of the constant *INTEND*, and *CARRY_OUT*, and others that define the activity status in the articles and type of registration, which also has an enumerated type.

In the process of formalization, we can find discrepancies in the law. For example, the first axiom can provide two different cases of understanding:

CASE 1: (x.activityStatus == INTEND || x.activityStatus == CARRY_OUT) &&(x.registration == TAX_PAYER_183)

CASE 2: x.activityStatus == INTEND || (x.activityStatus == CARRY_OUT && x.registration == TAX_PAYER_183)

Here, they differ by the location of the brackets and provide different content. This means that if someone intends to carry out economic activity, then they are already a taxpayer; however, this is nonsense. The axioms for the article 180.1.2 and 180.1.3 are presented as follows:

Axiom_2(x) ⬄ x.registration == TAX_PAYER || REGISTRATION_SUBJECT(x)

Axiom_3(x) ⬄ x.improting == true && x.importAmount >= TAXABLE_AMOUNT && x.taxRole == CHARGED(x)

Similarly, these axioms are defined by the predicates *REGISTRATION_SUBJECT(x)* and *CHARGED(x)*, which can be defined by the other axioms and by the Boolean attribute *importing*.

The consequence from the axiom is as if someone is not charged with a payment of taxes; however, if they are importing, then they are not a taxpayer, and this is nonsense. On the other hand, an individual will always be charged for tax payment when importing, so this phrase on charging is redundant.

Such discrepancies entail an ambiguous understanding, and if there are no explanations in the legal documents or precedents, then it is inconsistent or non-determined, which can be the source of trial error.

The articles in the tax code can be considered as static articles that define the relations between an agent's attributes and dynamics, which define the behavior of the agent when it changes its state or set of attributes. Correspondingly, the static articles are formalized as axioms, and the dynamic articles presenting behavior are formalized as actions.

Let's consider the articles that define the rules of an agent's behavior and present them as the actions in behavior algebra specifications.

"Article 181. The requirements for registration of taxpayers

181.1. Should the aggregate amount of transactions for supply of goods/services subject to taxation according to this section, including without limitations those using local or global computer network, accrued (paid) to the person within the past 12 calendar months exceed UAH 1000,000, the said person shall be registered as a taxpayer at the regulatory authority at their location (place of residence) in compliance with the requirements stipulated by Article 183 hereof, except for single tax payers"[27].

BP181_1 = (Forall(i:int) ((0 <= i <= 11 => x.sumService(timeActual - 11 + i, timeActual) >= limitVAT && x.singleTaxGroup == DOES_NOT) -> "181.1" x.timeVATstart = timeActual; x.registrationVAT.necessity = true)

In the action, we used the construction Forall for the presentation of the sum of i-elements. The attribute x.sumService is a parametrized (functional) attribute that presents the "amount of transactions for supply of goods/services subject to taxation according to this section, including without limitations those using local or global computer network". The other attributes can express other transactions. The attribute depends on the parameters that define the start month of the transaction and the end month. The attribute limitVAT is a constant that defines 1,000,000 UAH. This is a dynamic action that defines the change of the agent's state, especially if it changes the predicate registrationVAT.necessity to true; in this case, in the first month, the VAT shall be registered. The attribute x.singleTaxGroup shows that the individual is not a single taxpayer.

The action has the precondition that defines whether it can be true or not; then, action is possible and the postcondition can change the state of the agent. Furthermore, another example of action that presents the article in a formal way is shown as follows:

"**Article 184**. Taxpayer registration annulment

184.1. Taxpayer registration is valid until annulment thereof by way of removal from the Register of Taxpayers and shall take place in cases as follows:

a) any person registered as a taxpayer within the past 12 months has submitted an application for registration annulment if the total cost of goods/services subject to taxation supplied by the person within the past 12 calendar months is below the amount specified in Article 181 hereof, provided that the amount of tax liabilities is paid in cases specified in this section"[27].

BP184_1_a = (

VAT_CANCELATION_CONDITION_a -> ("VAT registrarion cancelation") (x.registration.status := DOES_NOT)),

The axiom *VAT_CANCELATION_CONDITION_a* is defined in the corresponding section of the description of the environment via the client program.

(VAT_CANCELATION_CONDITION_a ⬄

timaActual > x.timeVATperiod + 12 &&

SumService(timeActual,timeActual - 12) < limitVAT)

The behavior of the agents defines the rules of engagement and can be expressed in an algebraic form, especially by the system of behavioral equations. Let us consider the behavior that belongs to the registration of the agents as a payer of VAT:

B0 = BP_TIME || BP_REGISTRATION || BP_SALES,

BP_TIME = BP_NEXT_TIME.BP_TIME + BP_TERMINATION,

BP_REGISTRATION = BP_181_1.B1_NEC + BP182_1.B1_VOL,

B1_NEC = BP183_1.B2_NEC,

B2_NEC = BP183_8.BP_TERIMNATION,

B3_NEC = BP183_9_DATE3.B2,
B1_VOL = BP183_1.BP183_5.B2_VOL + BP183_1.B2_VOL,
B2_VOL = BP183_8.B_TERMINATION + BP183_9.B3_VOL,
B3_VOL = BP183_9_DATE1.B2 + BP183_9_DATE2.B2,
B2 = BP183_15.B3,
B3 = BP184_1.B4,
B4=BP184_1_A.BP_TERMINATION + BP184_1_G.BP_TERMINATION +
BP184_1_OTHER.BP_TERMINATION + 184_3.B2,
BP_SALES = BP_SALES_1.BP_SALES + BP_TERMINATION

The total behavior defines three parallel behaviors. The BP_TIME is the behavior of the environment that defines general things like time or external actions. The BP_SALES is the activity of the agent, a taxpayer, as he buys and sells goods or services. Finally, the BP_REGISTRATION defines the rules of the agent's registration as a VAT payer. The agent should or might change their VAT payment status if some conditions are reachable. The alternative representation of the behavior can be visualized as a UCM graph in the window of the client program (Figs. 3, 4).

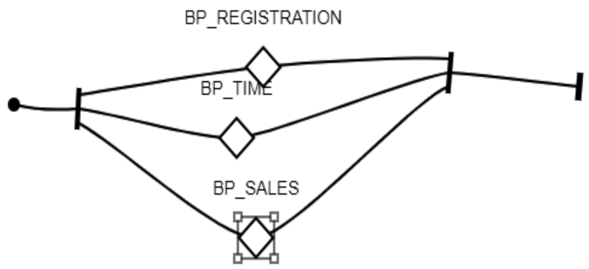


Figure 3. The UCM specifications for the parallel composition of the three behaviors of agents during the registration procedure

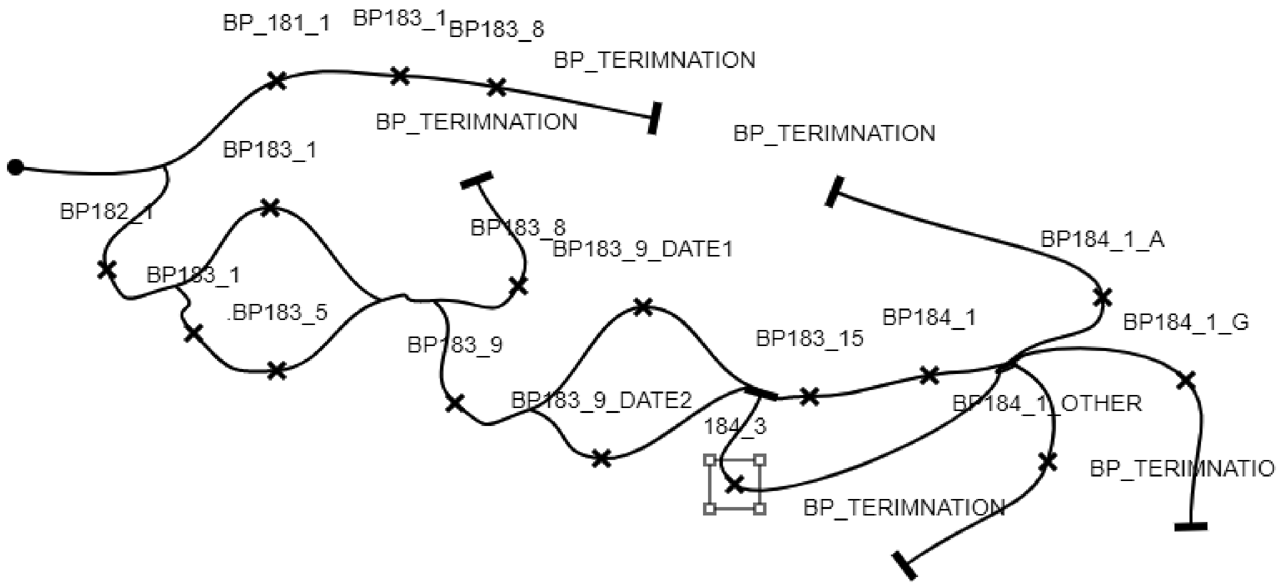


Figure 4. The UCM graph that defines the registration process

With the algebraic presentation of the tax code, we can analyze it for some properties via the use of the formal methods developed in the scope of behavior algebra theory.

Algebraic Analysis of the Tax Code

The analysis of the tax code works to detect any ambiguity or inconsistency within the articles and searches for cases that are not described in the law.

We can use the formula of inconsistency to precondition actions in the system of behavioral equations, as described in the tax code.

Let $a_1, a_2, \ldots$ be the list of preconditions that define the alternatives in the behavioral equation (1):

$$B = a1.B1 + a2.B2 + \ldots \tag{1}$$

The alternatives are *consistent* in the legal document if:

$$\cap\, a_i = 0 \tag{2}$$

If this formula (2) is true, then we have no ambiguous alternative in the tax code or other laws.

The alternatives are *complete* if:

$$\cup a_i = 1 \quad (3)$$

If this formula (3) is true, then we do not have any undefined cases in the legal document.

If the formula is not true, then we can create a counterexample that leads to non-determinism or incompleteness. For this purpose, we should start backward symbolic modeling from the state that is defined by the formula of violation and move to the initial state to get the values that allow it. If we find such a path, then the issue is reachable; otherwise, it is not reachable.

The precondition in the action shall be consistent with axioms, and axioms shall be consistent between themselves. For this purpose, we should check the satisfiability of the conjunctions of all axioms using the corresponding solvers. To reduce complexity, we can check the axioms that have dependent attributes.

These axioms can be inconsistent with the action of the legal document. If we provide exhaustive symbolic modelling—that means the coverage of all states and checking the satisfiability of axioms with preconditions of the actions—then we can be sure that formalized law and the set of axioms are consistent.

We have checked the tax code of Ukraine for inconsistency and incompleteness and have detected nearly 80 findings.

The static checking of axioms detected ambiguities in the articles. For example, Article 183.3 of the tax code stated that in the event of voluntary registration of a person as a taxpayer conforming with the requirements of sub-clause 6 of Article 180.1, the registration application shall be submitted to Article 183.7. However, the same person can be registered to Articles 183.1 and 183.2, as there is no statement clarifying that it is not for voluntary registration of a person as a taxpayer conforming with the requirements of sub-clause 6 of Article 180.1. This is further illustrated by the following propositional statements:

TaxPayerRegistration && Condition_180_1_6 ⇔
Registration_183,
TaxPayerRegistration && Voluntary && Condition_180_1_6 ⇔ *Registration_187_1*

The ambiguity is defined by checking the satisfiability of the conjunction of axioms.

Incompleteness can be defined by static checking of the preconditions of the actions. For example, in the event of voluntary registration to Article 183.9 (2–3), the date of the start of VAT payment shall be the same day on the next month, 20 days after the submission of the application. However, the tax obligation may occur before this date; therefore, it is unclear when to start such a date and how to change the registration procedure for mandatory VAT payers.

Analysis of Legal Documents

Algebraic matching methods check for consistency in juridical cases, such as court statements, agreements, licenses, new laws, taxation acts, and other legal documents. This is the most widespread and demanding problem in the legal world.

If we have a concrete case that corresponds to some sequence of agent actions, then we can consider matching the scenario with behavior. It is applicable for court statements to check or consider some cases to estimate the possibility of proceeding.

Another application of analysis considers behaviors that correspond to some legal documents and formalized law. In this case, we can check the consistency of these two behaviors and their applicability when we create the new law or some commercial agreement.

Methods of algebraic matching in behavior algebra anticipate the modeling of the parallel composition of two behaviors, or modeling of behavior, and the execution of the scenario. During the modeling, we detected an inconsistency in the environments of two legal entities, as the matched legal document did not correspond to the formalized law. Considering the two controversial interpretations of the cases, we can detect exactly which of the cases corresponds to the truth. However, sometimes, two controversial cases correspond to the truth, and this usually means there is ambiguity in the law.

Let us consider a case that illustrates the possible article inconsistency detection in the law by defining the non-determinism of axioms.

An enterprise bought the cars, and this amount was not included as tax credit; rather, it was included in the gross expenses that corresponded to the paragraph 7.4.2 of the law on VAT states: "Not included in the tax credit and refers to the gross costs of the amount of tax paid by the taxpayer when purchasing a car." Additionally, the Tax Administration refers to paragraph 5.2 of the law on income tax as follows: "Expenses are not included in gross expenses for payment of value added tax included in the price of goods (works, services) purchased by the taxpayer for production or non-production use."

By modeling the financial activity of the enterprise, we can monitor the time that the transaction occurred when the car was purchased. The state of the environment, E, is shown as follows:

TransactionSubject == PURCHAISING &&
PurchaisingSubject == CAR &&
TaxCredit == 0 &&
TransactionRefer == GROSS_EXPENSES

Accountant of the enterprise refers to the transaction as gross expenses that correspond to Article 7.4.2 of the law of VAT, which is presented by the following axiom:

A_7_4_2 ⇔ TransactionSubject == CAR => TaxCredit == 0 && TransactionReffer == GROSS_EXPENSES

The Tax Administration appointed a penalty corresponding to Article 5.2, which is presented by the following axiom:

A_5_2 ⇔ TransactionSubject == PURCHAISING => TransactionReffer != GROSS_EXPENSES

The state of the environment, which is defined in this case, is consistent with Article 7.4.2 and violates Article 5.2 simultaneously. Such an inconsistency occurs due to the ambiguity in the law, and this can be detected automatically by checking the satisfiability of the conjunction of axioms.

Court decisions usually contain two different points of view for the actions of the tax payer. The analysis of the documents anticipates the automatic recognition of these scenarios. A scenario is a sequence of events that correspond to the dates. Every event has a set of attributes of the interacting agent that are changed during the time fixed in this sequence.

The following scenario represents transactions in the supply chain of a logistics contract. We have transactions between the three stakeholders of the contract. Let x refer to the consumers of the service, and let y and z refer to the suppliers.

x.Transaction(1).Value = 1000 && x.Transaction(1).TAX_RATE == 0 -> (*x* sends payment to *y* for the logistic service and defines the tax obligations as 0)

y.Transaction(1).Value = 1000 (*y* receives the payment)

y.Transaction(2).Value = 700 (*y* sends the payment to sub supplier *z*)

The zero-tax obligation for the transaction follows from the corresponding article:

Article_195: *x.Transaction.TAX_RATE == 0 ⇔*
x.Transaction.Subject == LOGISTIC_INTERNATIONAL

This means that the transaction was implemented within the scope of the international logistics contract.

The court's decision was that the tax rate for the transaction should not be equal to 0 due to Article 196, which states:

*x.Transaction.TAX_RATE == VAT_Tax_RATE **
x.Transaction.Value ⇔ x.Transaction.Subject ==
LOGISTIC_INTERNATIONAL && y.Transaction.License ==
EXPEDITOR

This means that the supplier (y) receiving the payment shall have the license of international logistics; however, it provides an expedition service involving sub-suppliers that have the license.

After cassation appeal, the court of higher instance set that this transaction shall be considered as corresponded to the law on the expedition activity and this transaction consists of the sum of the costs of transport company z with license and costs of expedition agent y. The environment should be as follows:

x.Transaction.Value == y.Transaction.Value +
z.Transaction.Value &&
x.Transaction.Subject == LOGISTIC_INTERNATIONAL &&
z.Transaction.License != EXPEDITOR &&
y.Transaction.License == EXPEDITOR

Therefore, the final verdict that the high instance court confirmed was:

*x.TAX_RATE == VAT_Tax_RATE * (y.Transaction.Value - z.Transaction.Value)*

This formula follows the usage of axioms, the law on VAT, and the law on transporting and expediting activity, but the statements of the court and the taxpayer (x) are false. This example demonstrates the possibility of the agent's actions to conform to the law or a combination of the law. The information on the actions' parameters shall come from the accountant systems, or if it is the court's verdict, it shall be recognized from the text of the verdict.

Text Mining in Legal Documents

In the task of analyzing legal documents, there is a problem regarding the automatic presentation of documents in an algebraic manner. This problem, otherwise known as text mining, is prevalent in the legal domain, and there are many related works dedicated to this problem.

When we performed the translation of the natural text into the algebraic presentation, we tried to consider known modern techniques. The problem is that there is a lack of translators working with the Ukrainian language in Ukraine.

The task was to extract the behavior of agents, such as the taxpayer, and define the values of their attributes.

Recognizing agents is difficult due to the different usages of agents' names in legal documents (name of a company, abbreviation, short or long name of a company). Although, deep linguistic analysis combined with machine learning can provide the exact identification and corresponding formalization of agents and other entities from a legal text.

We had some connection of legal text with the law via the numbers and titles of articles. Formalized articles in law have a set of attributes that can correspond to possible words from the text in the documents. Using such correspondence, we can extract the equations as shown below:

ATTRIBUTE NAME = VALUE OF THE ATTRIBUTE

Another problem regarding the creation of scenarios is defining the sequence of events. This can correspond to concrete dates or appear in the text in some order. Usually, in the scope of the tax code, it is the sequence of transactions and corresponding taxes with reference to the number of articles.

The problem of legal text mining is widely discussed in the literature and has specific algorithms for different languages, but it is still a challenge, especially for Ukrainian and other Slavic languages.

One solution is to create a legal text that corresponds to specified linguistic rules that are suitable for recognizing. Some experiments with legal texts were implemented, and a set of such rules have been defined.

Next Experiments and Conclusions

There are many other laws in use in Ukraine that have yet to be formalized in an algebraic manner. For resolving this, we plan to use the existing tools to translate legal documents to the system of relation between notions. However, the formalization of the text as a set of axioms still demands manual work.

The practice of demonstrating that 80% of discrepancies are detected during the process of formalization means that such a process might be standardized, even during the development of new laws.

The formalization of legal documents, like court verdicts, shall be implemented fully automatically, and the existing database with such documents shall provide the possibility to use big data technologies and machine learning.

The next attempts to use the algebraic approach will focus on the formalization of business agreements and automatic checking for conformance to the tax code and other laws. The technology of smart contracts can also be integrated with the algebraic approach for checking the correctness of transactions and agents' actions between stakeholders.

The algebraic approach presents a solution to analyzing legal documents and maintaining their conformance to the law in order to avoid ambiguities and incompleteness. For this purpose, the following technologies shall be established:

Formalization of law articles via the algebraic model. This procedure can be implemented semi-automatically, where all subjects can be presented as the elements of ontology. The creation of the axioms and agents' behavior shall be realized manually.

Legal documents, such as licenses, court decisions, and agreements can be formalized automatically for further analysis and conformance to the law. Smart contracts can be translated automatically into the algebraic form. Our team has already performed experiments regarding the translation of the solidity language to algebraic specifications.

The methods of algebraic analysis pertaining to legal documents will be implemented and will include algebraic matching, symbolic modeling, and other methods of automatic statement proving. We plan to provide access for the integration with laws, as well as other legal documents of other countries, via corresponding translation programs.

It is anticipated that by providing web access for the use of algebraic servers that contain the methods of analysis, available translations of legal documents to algebraic specifications, and access to the database of formalized laws with the possibility of extending it.

References

1. LETICHEVSKY, A., LETYCHEVSKYI, O., & PESCHANENKO, V. (2016). Insertion modeling and its applications. Computer Science Journal of Moldova, 72(3), 357-370.
2. WASS, C., DINI, P., EISER, T., HEISTRACHER, T., LAMPOLTSHAMMER, T. J., MARCON, G., ... & WINKELS, R. (2013). OpenLaws. eu. In Proceedings of the 16th international legal informatics symposium IRIS (Vol. 292, pp. 21-23).
3. PEYCHEV, V. (2005). Legal document–a formal model. Problems of Engineering Cybernetics and Robotics, 55, 64-70.
4. PETERSEN, K. E. (2011). Experiences with "Lex Dania Live". In From Information to Knowledge (pp. 69-76). IOS Press. DOI: https://doi.org/10.3233/978-1-60750-988-2-69
5. DRIZA MAURER ARDITA, HOLENSTEIN URS PAUL, 2006. The electronic publication of Swiss legislative acts. Development of a structured model for federal, cantonal and communal legal texts: the XML–CHLexML schema, In: Bundesamt für Justiz, Mitarbeiterinnen und Mitarbeiter; Schindler, Benjamin. Helbing Lichtenhahn, 391–401.
6. KAMAL, A., QURESHI, N., TARIQ, M., & CANTT, R. Optimal and Economic Solution for eLaw Data Storage and Retrieval.
7. FRANCESCONI, E. (2011). A review of systems and projects: Management of legislative resources. Legislative XML for the semantic Web, 173-188. https://doi.org/10.1007/978-94-007-1887-6_10
8. BOER, A., HOEKSTRA, R., WINKELS, R., VAN ENGERS, T., & WILLAERT, F. (2002). Metalex: Legislation in xml. Legal Knowledge and Information Systems (Jurix 2002), 1-10.
9. ENGERS, T. V., BOER, A., BREUKER, J., VALENTE, A., & WINKELS, R. (2008). Ontologies in the legal domain. In Digital Government (pp. 233-261). Springer, Boston, MA. DOI: https://doi.org/10.1007/978-0-387-71611-4_13
10. BREUKERS, J. A. P. J., & HOEKSTRA, R. J. (2004). Epistemology and ontology in core ontologies: FOLaw and LRI-Core, two. In Proceedings of EKAW Workshop on Core ontologies [Internet]. Northamptonshire, UK: Sun SITE Central Europe.
11. GANGEMI, A., PRISCO, A., SAGRI, M. T., STEVE, G., & TISCORNIA, D. (2003, November). Some ontological tools to support legal regulatory compliance, with a case study. In OTM Confederated International Conferences" On the Move to Meaningful Internet Systems" (pp. 607-620). Springer, Berlin, Heidelberg.DOI: https://doi.org/10.1007/978-3-540-39962-9_64
12. HOEKSTRA, R., BREUKER, J., DI BELLO, M., & BOER, A. (2007). The LKIF Core Ontology of Basic Legal Concepts. LOAIT, 321, 43-63.
13. GRIFFO, C. (2015). Ufo-l: A core ontology of legal concepts built from a legal relations perspective. Doctoral Consortium Contributions, IC3K-KEOD.
14. CASANOVAS, P., PALMIRANI, M., PERONI, S., VAN ENGERS, T., & VITALI, F. (2016). Semantic web for the legal domain: the next step. Semantic Web, 7(3), 213-227. DOI: https://doi.org/10.3233/SW-160224
15. SIRIN, E., PARSIA, B., GRAU, B. C., KALYANPUR, A., & KATZ, Y. (2007). Pellet: A practical owl-dl reasoner. Journal of Web Semantics, 5(2), 51-53. DOI:https://doi.org/10.1016/j.websem.2007.03.004
16. HORROCKS, I., PATEL-SCHNEIDER, P. F., BOLEY, H., TABET, S., GROSOF, B., & DEAN, M. (2004). SWRL: A semantic web rule language combining OWL and RuleML. W3C Member submission, 21(79), 1-31. https://www.w3.org/Submission/SWRL/

17. CABOT, J., CLARIS, R., & RIERA, D. (2008, April). Verification of UML/OCL class diagrams using constraint programming. In 2008 IEEE International Conference on Software Testing Verification and Validation Workshop (pp. 73-80). IEEE. DOI:https://doi.org/10.1109/ICSTW.2008.54
18. ALI, S., BRIAND, L. C., ARCURI, A., & WALAWEGE, S. (2011, October). An industrial application of robustness testing using aspect-oriented modeling, UML/MARTE, and search algorithms. In International Conference on Model Driven Engineering Languages and Systems (pp. 108-122). Springer, Berlin, Heidelberg.. DOI:https://doi.org/10.1007/978-3-642-24485-8_9
19. MCCARTY, L. T. (1989, May). A language for legal discourse i. basic features. In Proceedings of the 2nd international conference on Artificial intelligence and law (pp. 180-189).
20. SUTHERLAND, H., & FIGARI, F. (2013). EUROMOD: the European Union tax-benefit microsimulation model. International journal of microsimulation, 6(1), 4-26.
21. Taylor, R., Sutherland, H., & Gomulka, J. (2001). Using POLIMOD to evaluate alternative methods of expenditure imputation. Microsimulation unit research note MU/RN38.
22. POPPLE, J. (1996). A pragmatic legal expert system. Dartmouth (Ashgate).
23. LETICHEVSKY, A. A., & GILBERT, D. (1998). A general theory of action languages.
24. LETICHEVSKY, A., & GILBERT, D. (1999). Interaction of agents and environments. Resent trends in Algebraic Development technique, LNCS 1827, 272.
25. ITU-T, Z. 151 User requirements notation (URN)–Language definition. ITU-T, Nov, 2008, https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Z.151-201810-I!!PDF-E&type=items
26. LETICHEVSKY, A., LETYCHEVSKYI, O., PESCHANENKO, V., & POLTORACKIJ, M. (2017, September). An Algebraic Approach for Analyzing of Legal Requirements. In 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW) (pp. 209-212). IEEE. DOI: https://doi.org/10.1109/REW.2017.51
27. Tax Code of Ukraine, https://zakon.rada.gov.ua/laws/show/2755-17?lang=en#Text

Received 02.08.2022

About the authors:

Oleksandr Letychevskyi,
Doctor of physical and math sciences
Head of digital Automata Theory Department
Glushkov Institute of Cybernetics, Kyiv, Ukraine.
Number of scientific publications in Ukrainian publications – 31.
Number of scientific publications in foreign publications – 59.
h-index – 5.
https://orcid.org/0000-0003-0856-9771

Volodymyr Peschanenko,
Head of Department of Informatics, Software Engineering and Economic Cybernetics.
Kherson State University, Kherson, Ukraine.
Number of scientific publications in Ukrainian publications – 17.
Number of scientific publications in foreign publications – 31.
h-index – 5.
https://orcid.org/0000-0003-1013-9877

Maksym Poltorackiy,
PhD in Information Technology, lecturer
Department of Computer Science and Software Engineering
Kherson State University, Kherson, Ukraine.
Number of scientific publications in Ukrainian publications – 5.
Number of scientific publications in foreign publications – 15.
h-index – 3.
https://orcid.org/0000-0001-9861-4438

Place of work:

Letychevskyi Oleksandr Oleksandrovich
Digital Automata Theory Department Glushkov Institute of Cybernetics
Kyiv, 03187, Ukraine, Akademika Glushkova Avenue, 40
Ph.: (+38) (044) 526-20-08
Fax: (+38) (044) 526-41-78
e-mail: oleksandr.letychevskyi@litsoft.com.ua

Peschanenko Volodymyr Serhiyovych
Department of Computer Science and Software Engineering
of Kherson State University
Kherson, 73003, Ukraine, University Street, 27
Ph.: +(38) (0552) 326731
Fax: +(38) (0552) 492114
e-mail: volodymyr.peschanenko@litsoft.com.ua

Poltorackiy Maksym Yuriyovych
Department of Computer Science and Software Engineering
of Kherson State University
Kherson, 73003, Ukraine, University Street, 27
Ph.: +(38) (0552) 326731
Fax: +(38) (0552) 492114
e-mail: maxim.poltoratskiy@litsoft.com.ua

Прізвища та ім’я авторів і назва доповіді англійською мовою:
Letychevskyi O. O., Peschanenko V. S., Poltorackiy M. Yu.
Algebraic approach to the analysis of legal documents

Прізвища та ім’я авторів і назва доповіді українською мовою:
Летичевський О. О., Песчаненко В.С., Полторацький М. Ю.
Алгебраїчний підхід у аналізі юридичних документів

Contacts for the editor: Maksym Poltorackiy, lecturer at the Kherson State University,
e-mail: mpoltorackiy@gmail.com, ph.: +(38)0506686238