UDC 519.85 https://doi.org/10.15407/pp2022.03-04.128

FORMAL SEMANTICS AND ANALYSIS OF TOKENOMICS PROPERTIES

Oleksandr Letychevskyi, Volodymyr Peschanenko, Maksym Poltoratskyi, Yuliia Tarasich , Maksym Vinnyk

Сьогодні ми спостерігаємо швидкий розвиток продуктів і послуг, в основі яких лежить технологія блокчейн. Криптовалюти та токени стають невід'ємною частиною повсякденного життя людини. Одним із головних і, водночас, найскладніших задач для кожного проєкту є створення самокеруючої економіки токенів. Порушення таких властивостей, як рівновага та децентралізація, може призвести до провалу проєкту та фінансових втрат. Використання математичних і формальних методів є простим і ефективним способом створення самокеруючої економіки токенів на етапі розробки MVP. Незважаючи на швидкий розвиток токеноміки, її популярність і швидкі темпи впровадження технології блокчейн в різних сферах бізнесу, побудові формальних моделей токеноміки присвячено мало наукових та технічних робіт.
У цьому дослідженні ми представляємо алгебраїчний підхід до аналізу властивостей токеноміки. Підхід до алгебраїчного моделювання реалізовано в рамках системи інсерційного моделювання (IMS), розробленої в Інституті кібернетики ім. Глушкова НАН України під керівництвом академіка НАН України, проф. О.А. Летичевського. Методи алгебраїчного моделювання доводять властивості безпеки та живучості. Інсерційне моделювання — це підхід до моделювання складних розподілених систем, який базується на теорії взаємодії агентів та середовищ. Алгоритм моделювання базується на історичних даних біржової торгівлі та ліквідності токенів, що дозволяє нам робити точні прогнози та показувати можливі результати.
Розглянуто взаємодію агентів токеноміки та їх поведінку, представлену рівняннями поведінкової алгебри. Розглянуто використання алгебраїчного підходу до розробки токеноміки для Інтернету речей, представлено формальні представлення та методи аналізу властивостей. Отримані результати дозволяють обговорювати можливість використання формальних методів у дослідженні токеноміки.

Ключові слова: алгебраїчне моделювання, токеноміка, децентралізація, токеномічна рівновага, формальні методи, інсерційне моделювання.

Today we are witnessing the rapid development of products and services based on blockchain technology. Cryptocurrencies and tokens are becoming an integral part of a person's daily life. One of the main and, at the same time, the most difficult task for each project is the creation of a self-governing token economy. The violation of properties, such as equilibrium and decentralization, can result in the failure of a project and financial losses.
Using the math and formal methods is a simple and efficient way to create self-sustainable token economies right at the stage of MVP development. Despite the rapid development of tokenomics, its popularity, and the rapid pace of implementing blockchain technology in various business areas, only a small number of works have examined formal models of tokenomics.
In this study, we present an algebraic approach to analyzing the properties of tokenomics. The algebraic modeling approach is implemented within the framework of the Insertion Modeling System (IMS) that was developed at the Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine under the guidance of an Academician of the National Academy of Sciences of Ukraine, Professor A.A. Letichevsky. The algebraic modeling methods prove the properties of safety and liveness. Insertion modeling is an approach for modeling complex distributed systems, which is based on the theory of interaction between agents and environments. The modeling algorithm is based on the historical data of exchange trading and the liquidity of tokens - which allows us to make accurate predictions and show possible outcomes.
The interaction of tokenomics agents and their behavior, represented by the equations of behavioral algebra, is considered. The use of the algebraic approach to tokenomics development for the Internet of Things is considered, and the formal representations and methods of property analysis are presented. The obtained results allow us to discuss the possibility of using formal methods in the study of tokenomics.

Keywords: Algebraic Modeling, Tokenomics, Decentralization, Tokenomic Equilibrium, Formal Methods, Insertion Modeling..

Introduction

In tokenomics project creation, the most difficult task is to determine a model of interaction between project participants. The model must provide economic equilibrium in its functioning for the project authors to obtain the predictable profit.

The project takes into account the motivation of players and investors and the long-term use of a conditional financial unit — a token that ensures the fulfillment of the goal. On the one hand, the implementation of the project is complicated by the fact that different scenarios might arise, which does not depend on the implementation of smart contracts, and on the other hand, it is almost impossible to comprehend the whole set of tokenomics scenarios in different behaviors. Therefore, formal methods, probabilistic methods, simulation, and other mathematical approaches are used for project tokenomics development. In this paper, we consider the algebraic semantics of project tokenomics, which indicates tokenomics components and the formal processes description that occur in it.

This representation of tokenomics allows the use of formal methods that verify tokenomics properties, analyze its states, and predict probable scenarios. For tokenomics formalization, we used the algebra of behaviors [1] and the theory of agents and environments [2], which helped determine the main components and actions of tokenomics in the form of behavioral equations.

We have presented a general approach that may be used to build a formal tokenomics model in terms of behavior algebra. This makes it possible to apply a method such as algebraic modeling and provide proof of execution of the liveness and safety properties.

This approach was earlier implemented in systems that the authors designed, such as the Tokenomics constructor [3], model maker [4], and the so-called tool “Algebraic Virtual Machine” (AVM). A formal model in the specification of behavior algebra provides an opportunity to use appropriate formal methods within these systems.

Related Works

Despite the rapid development of tokenomics, its popularity, and the rapid pace of implementing blockchain technology in various business areas, only a small number of works have examined formal models of tokenomics.

The use of differential game theory and stochastic modeling techniques for analyzing tokenomics models is considered in [5]. The article simulates supply and demand as stochastic dynamic systems. Services and demand for services are considered Poisson processes. The cadCAD tool [6], designed for the automated design of complex systems, was used for modeling. The basis of modeling is system dynamics. The use of this tool was also described in the article, and the authors presented their experience with using the cadCAD system to analyze the Insolar tokenomic model.

An interesting example of an agent-oriented approach is the use of the Tokesim tool [7]. Tokesim is an agent-oriented token economy simulator developed in Python programming language and created using the Mesa ABM and OpenRPC infrastructure. The tool allows developers to model interactions with smart contracts and test smart contracts based on Ethereum.

A theoretical analysis method for estimating the profit of different roles in tokenomics and a root method for calculating the weights of indicators is proposed in [8]. The proposed algorithms were implemented in the Python programming language. The authors created a value creation network to analyze the main factors and proposed two economic models: a model of the hierarchical structure of the alliance and a new model of profit.

In [9], a dynamic cryptocurrency/token pricing model that allows users to conduct peer-to-peer transactions on digital platforms was proposed. The equilibrium value of tokens is determined by aggregating the transactional demand of heterogeneous users rather than discounting cash flows, as in standard valuation models.

In [10], the authors performed modeling of markets with a minimum number of variables and compared the speed of transactions, stability, and design properties of tokens at different levels of tokenization based on a chemical approach. The article presents three methods of analysis that, according to the authors, provide important information about economic systems: kinetic analysis, stability analysis, and an economic approach to open markets and, in particular, the recalculation of prices according to the law of supply and demand.

Regarding tokenomics modeling, authors often write about the use of tools such as BlockSci, Blocksim, and Simblock, but these applications do not solve the problem.

We consider an algebraic approach that allows building a model of the self-governing economy with the study and proof of the satisfiability of properties.

Theory of Agents and Environments and Algebra of Behavior

In the theory of agents and environment interaction, we consider agents as entities that change their state in the process of evolution. The state of the agent is determined by its attributes, which are typed, i.e., determined by a certain theory with operations and predicates in it. For example, linear arithmetic with arithmetic operations and such predicates as equalities and inequalities, or byte theory, where operations are the copying of bytes and predicates are comparing of their contents. In tokenomics, agents are participants in a tokenomic game, and the attributes are the number of tokens and the amount of fiat money, which are determined by arithmetic over floating-point numbers. Boolean functions, such as disjunction, conjunction, and negation, are also used to express attribute statements.

The environment also contains attributes that are available to all agents. In turn, only their attributes are available to agents. The states of all agents and the values of the attributes of the environment constitute the general state of the environment, which formulas can express over the attributes that may relate to different theories. Each agent can perform certain actions that change its state, such as changing the values of the attributes of the agent. Each action consists of a precondition and a postcondition, and they are also formulas from the theories defined in the model. The precondition determines that it is compatible with the state of the environment, and that its conjunction is satisfiable. Also, the postcondition determines how the state of the environment will change after the action.

Example of action in tokenomics (agent x purchases N tokens).

$$\mathrm{buy}(\mathrm{x}, \mathrm{N}): (\mathrm{x.fiat} >= tokenPrice * N) -> (x.Fiat = x.Fiat - tokenPrice * N;\ x.token\ =\ x.token\ +\ N)$$

An assignment operator is used in action formulas that show how the environment is changing. The general formula of the environment will be used and changed according to such operators when modeling tokenomics.

Expressions of behavioral algebra are the behavioral operations of the actions of agents. Operation “.” (prefixing) $\mathrm{a}.S$ determines the execution of action a under the behavior S. Another operation “+” (alternative choice) defines the branching of two behaviors. These behaviors can be performed non-deterministically or take into account the precondition of the action.

The behavioral equation can contain parallel or sequential compositions of behaviors. The expression of behavioral algebra consists of the actions, behaviors, and operations of behavioral algebra, and of corresponding compositions. The behavioral equation contains a unique identifier of behavior on the left part and a behavioral algebra expression on the right part. Let us consider how to use behavior algebra in the formalization of tokenomics.

Behavioral Equations in Tokenomics

Equations determine the behavior of agents who are interacting in the project. We consider the following types of agents:

Agent that presents a team of authors of a project. This agent builds the service and its tokenomics to make a profit in tokens and increase the token's liquidity. Let us call it a *Team*.

Investor agents buy cryptocurrencies and provide fiat support to the project team. Typically, such agents are those who buy tokens in early rounds in private sales at a fairly low price. Also, investors have the privilege of making a profit from the operation of the service. These will be indexed agents named *Inv*(i).

Agent who presents users of the service and buys tokens to pay for services. Let us denote him as *Users*.

Agent that presents traders, i.e., those who buy and sell tokens for the purpose of speculation for profit. They are guided by token liquidity criteria and can hold tokens. Let us call the agent *Traders*.

Other types of agents (who will be rewarded by the team to improve service efficiency) can also be involved in the project. Usually, these are advisors, the marketing team, the project's technical support team, and other groups that help increase the token's liquidity. Let us denote the set of such participants as *Support*. In addition, all agents interact with the *Exchange* agent.

The tokenomics project examines the token's lifecycle in some environments, in which agents exchange cryptocurrencies with goals to invest, buy and sell, provide service, and receive rewards. This anticipates the involvement of several parallel processes that must be written in behavior algebra using a parallel composition.

PROJECT = INVESTMENT_ROUNDS || TRADING || SERVICE

This equation shows that the project is the parallel execution of several processes, including those related to investment, trading, and service. The components of the parallel composition are synchronized by the time in relation to some *timeUnit*, but each of the processes may begin earlier or later. This main equation represents the upper level of all tokenomics. Next, each of the processes must be described by other behaviors and atomic agents' actions that operate in tokenomics. If we start with the investment part, the behavior of the involved agents is described as the sale of tokens in a certain round. Regardless of whether it is a private or public round, according to a certain sales schedule. The tokens that are issued are distributed both for sale and for distribution among those agents who are also involved in building the service.

INVESTMENT_ROUND = ((sellTokens. distributeTokens); (nextRound. INVESTMENT_ROUNDS + !nextRound))

This equation contains the *sellTokens* and *distributeTokens* actions that are performed sequentially for each round. The round is determined by the *nextRound* action. When the number of rounds is exhausted, the action !*nextRound* is triggered and the behavior ends.

The semantics of atomic actions are presented as follows:

sellTokens: (Forall (i: int) (1 ≤ i ≤ INVESTOR_NUMBER) →
(Inv(s). token = Inv(i). token + tokenInvest(i, timeUnit);
totalToken = totalToken − tokenInvest(i, timeUnit))

To study the different properties of tokenomics, we can determine the appropriate number of agents for each type. Differentiation of investors, as well as traders, may be required to study such property as centralization. We used a universal quantifier to implement a loop in the postcondition. The loop contains the variable *tokenInvest(i, timeUnit)*, which determines the sales schedule according to the defined *timeUnit* interval. The second action is defined by similar semantics, but now for *Support* agents.

distributeTokens: (Forall (i: int) (1 ≤ i ≤ SUPPORT_NUMBER) →
(Support(i). token = Support(i). token + tokenSupport(i, timeUnit);
totalToken = totalToken − tokenInvets (i, timeUnit)))

Trading modeling is a non-trivial task, as traders' intentions to buy, sell, or hold tokens may depend on factors that cannot be modeled. These can be military actions, statements of politicians or businessmen about their intentions, or the sale of company shares. The liquidity and fluctuations of the bitcoin exchange rate are also decisive for traders' intentions.

Let us write the equation for traders' behavior:

TRADING = exchangeListing. $\mathrm{TRADING}_{\mathrm{CYCLE}}$,
TRADING_CYCLE = (buyToken(X) || sellToken(Y)); (nextTimeUnit. TRADING_CYCLE)),

Trading begins with the registration of the token on the exchange, where its starting price, the number of tokens, and the amount of fiat money that supports the value of the token are set.

During the trading cycle for the selected period, the agent who determines the set of traders can buy a certain number of tokens X and sell tokens Y.

The number of tokens X and Y is calculated by a formula that is determined by dependencies on some attributes, such as, for example, the liquidity of the tokens.

In addition, it is possible to consider historical data on changes in buying and selling trends relative to other factors, such as changes in the bitcoin exchange rate, and in modeling, to plan trading activity for different scenarios of bitcoin exchange rate changes. Using historical data of different cryptocurrencies, we built their regression and connected it with the change in the rate of bitcoin.

Another way to determine the semantics of traders' behavior is to consider the basic properties of trading activity and to consider generalized behavior for further algebraic modeling. To do this, we determined the formula for

X and for Y according to the historical data of the change in properties.

If we consider the chart of changes in the value of liquidity in trading activities on the exchange, we have curves that depend on time (Fig. 1).

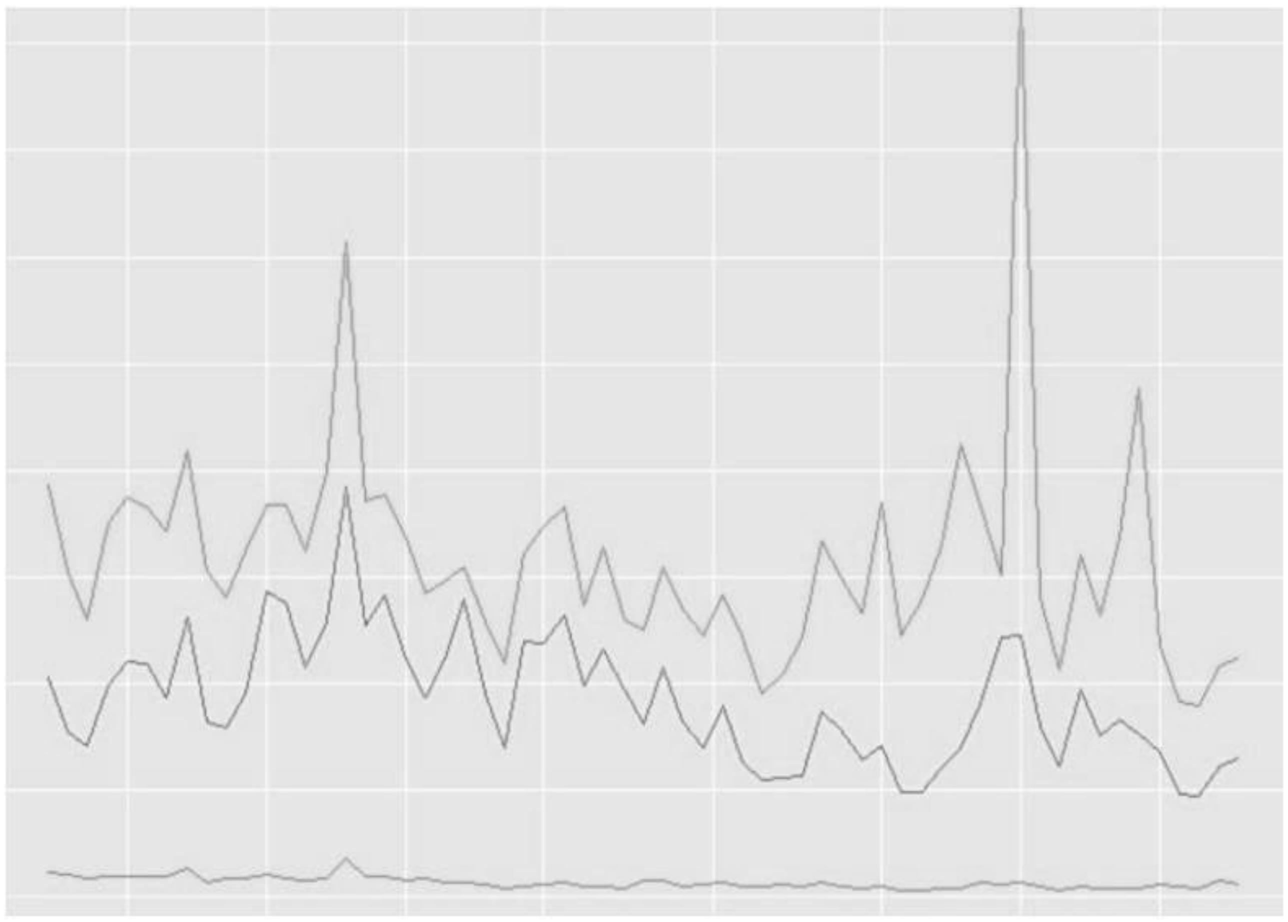

Fig. 1. Chart of changes in the value of liquidity in trading activities on the exchange.

Considering the various properties of the graph, we can highlight some sets of them. Let us present three of them for a better understanding of the semantics of behavior (Fig. 2).

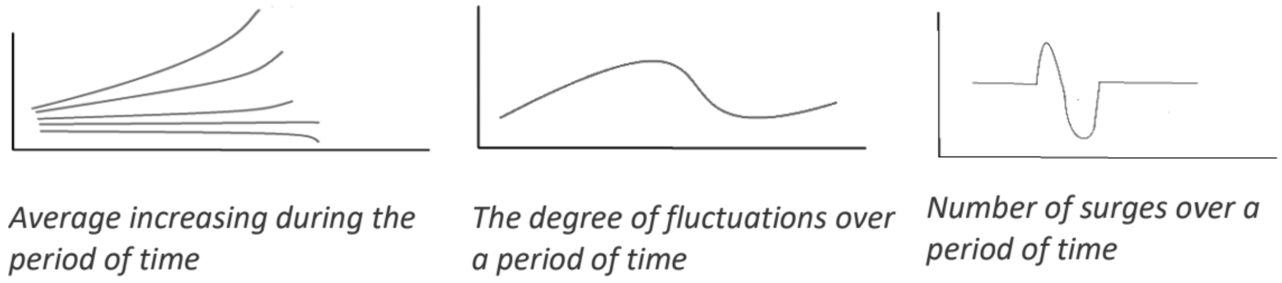
Average increasing during the period of time
The degree of fluctuations over a period of time
Number of surges over a period of time

Fig. 2. Elements of the graph of trading activities.

The average increase or decrease in liquidity was determined by the difference between the initial and final values during the selected unit of time. The degree of fluctuation is determined by the number of changes in the direction of an increase or decrease in liquidity value. We also determined the number of surges that exceeded a certain threshold Z.

As a result of collecting such properties and studying their trends for some crypto-currencies, we can get many scenarios of liquidity change, which is determined by the following formula:

Forall (t: int) (T(1) $\leq$ t $\leq$ T(2)) && ($L1 \leq Liquidity(t) \leq L2$) &&
Exist (k: int, i: int) (Liquidity(Tk(i)) Liquidity(Tk(i) + 2)) && (0 $\leq i \leq N$) &&
Exist (m: int, j: int) (Liquidity(Tm(j)) Liquidity(Tm(j) + 2)) &&(0 $\leq j \leq N$) &&
(Liquidity(Tm(j) + 1) > = Z)

These formulas will be useful in algebraic modeling. They determine the semantics of the trading activity of a set of scenarios, which is limited to these formulas of curve properties.

The number of tokens X and Y can be calculated by some function of the liquidity value:

$X = F(Liquidity(i), sale)$

$Y = F(Liquidity(i), buy)$

Function F depends on the type of exchange on which the token is sold. On the one hand, this may be determined by historical data on changes in liquidity for centralized exchanges and, on the other hand, by the game of traders on increasing or decreasing liquidity on a decentralized exchange.

The token's liquidity is determined by the service that provides the tokenomics project. During service, there is an intensive exchange of tokens between the agents involved in the service. Necessary tokens are bought on the exchange by agents who are interested in using them in the service. Agents who earn tokens can behave like traders or sell surpluses on exchanges. The behavior of such agents significantly affects liquidity.

In the general case, the service corresponds to the action represented by the mining *function W*: $M = W(R, P)$,

where R is the amount of payment for the service, M is the number of mined tokens, and P is the service parameters.

The composition of this function depends on the specifics of the service, i.e., its parameters. In various projects, the service provides regulators, penalties, and motivations that meet the rules of changing its parameters

Thus, the equation of behavior algebra for the service part is presented in the following form:

$SERVICE = (buyToken(Users, A).miningFunction.SellToken(Miners, B).nextTimeUnit.SERVICE)$

buyToken (Users, A) – the purchase of tokens by service users. A is the number of tokens that is determined by the need to pay for the service during the selected period of time.

sellToken (Miners, B) – sale of mined tokens. B is the number of tokens determined by the desire of miners to sell tokens in accordance with the trends of traders or a fixed part of the token surplus. All these actions are defined in the syntax of actions with behavioral equations.

Tokenomics' Properties

Analyzing project tokenomics requires that different scenarios of token behavior lead to the desired result. On the one hand, to the equilibrium of parameters, on the other hand, to the planned profit from the service. Such desirable parameters can be expressed using arithmetic inequalities, and properties of this type are called liveness properties.

We are talking about the reachability of liveness properties. Reachability may consist of the existence of such a state of the general environment that is compatible with the corresponding property. Since in algebraic modeling, both property and state are formulas, then the reachability of a property is determined by the satisfiability of the conjunction of property and state formula, i.e., by proving that a formula can be true if there are such values of agent attributes. Thus, the properties of liveness can be proved by algebraic modeling or other formal methods, such as the generation of invariants.

Another problem that can be solved by using algebraic modeling is finding the initial parameters of tokenomics that correspond to a set of possible scenarios that, in turn, correspond to the properties of liveness. To solve this problem, we use backward modeling, which determines the symbolic scenario from the desired property to the initial state. The initial state represents the set of scenarios that lead to this property.

The tokenomics equilibrium can also be expressed by the properties of liveness, i.e., the stay of attributes within the desired values. Another representation of the equilibrium property is the convergence of data to some intervals. The formula of convergence in algebraic language can be presented in different forms, but it is necessary to prove the properties of the set of values of attributes in different periods of time. This can be different periods of time in which the satisfiability of the formula shall be proved, or certain sets of points, which are defined to represent the property of convergence of the desired values of attributes to some intervals.

Other properties define events that are not desirable in scenarios. These can also be interval restrictions, such as for liveness properties, or another formula over attributes. One example of such properties is the property of centralization, i.e., of the situation when a certain group of agents collected a significant number of tokens and can significantly affect the tokenomics for unjustified profit. Algebraic modeling methods were also used for this formula. Sometimes it is useful for tokenomics to determine its resistance to attackers (phenomena such as Sybil attacks with the creation of many controlled nodes), fake trading (trading between own wallets), and other means that can negatively affect the liveness of tokenomics. One way to define attacks is to reach an attack behavior pattern. Such a pattern can be represented by formulas between attributes and by using a symbolic behavioral expression. For example, let us consider the following behavioral equation:

$B1 = (a1.X; B0; a2.a3.Y; a4)$

In this expression, X and Y are unknown behaviors that are not present in the left parts of the system of behavioral equations. That is, this equation is a pattern of some behavior, which says that after the action $a1$ is X that is arbitrary behavior of agents. Afterward a known behavior $B0$ and the sequence of actions $a2$ and $a3$ is following, After this sequence an arbitrary behavior Y follows until the action $a4$.

To find such pattern, special algorithms of algebraic matching are used, which the authors used to detect resistance to attacks in software systems [11].

Algebraic Modeling as an Analysis of Tokenomics' Properties

Unlike well-known types of modeling with specific values, such as simulation or probabilistic modeling, algebraic modeling is performed under a set of formulas that cover a set of states. If the simulation determines the specific states of the agent and the reachability of the properties is checked by selecting different specific values of the agents' attributes, then the state determines the set of possible values of agents' attributes, which are determined by formulas. Algebraic modeling is the unfolding of a behavioral equation according to the agents' actions being executed. Unfolding $R(S,a)$ of the behavior equations system S with respect to action a is possible if the precondition of action a is satisfiable, i.e., there are such values of attributes that appear in the precondition that the precondition is true. Unfolding $R(S,a)$ of the equation $S = a.S^1$ is performing by substitution of right side of S^1 into equation S. Operations of alternative choice in equations entail the different scenario of unfolding. Let S is a system of behavioral equations, then applying the unfolding with respect to the actions, we obtain a sequence of unfoldings: $R\ (S,a1), R\ (S^1,a2), R\ (S^2,a3)$,..., which corresponds to the sequences of actions $a1, a2, a3, \dots$ Let $R0$ is the initial state of the system of agents, then the symbolic execution of action $a(R0)$ is a new formula of state $R1$, which corresponds to the postcondition of action a. Formula $R1$ is obtained using the predicate transformer function [12] pt, which depends on the conjunction of the initial formula with the precondition and postcondition:
$\mathrm{R1} = \mathrm{pt}(\mathrm{R0}\ \&\ pre(a), post(a))$

By unfolding the behavioral equation, we perform algebraic modeling and obtain a sequence of formulas of the system's states of agents $R0, R1, \dots$

Let P is a tokenomic property that must be checked in the R state. The property is reachable in the *R* state if $P\&R$ is satisfiable, i.e., there are attribute values where $P\&R$ are true. We are proving that in some state R, property P will be reachable. For example, the conditions of equilibrium tokenomics are carried out after some time, or under no circumstances is unreachable centralization (i.e., the concentration of tokens in certain agents).

The satisfiability or reachability of certain properties may not be sufficient to determine tokenomics. Since modeling for arbitrary attributes has been considered, it is necessary to determine the specific initial values of those attributes that have not been defined. To do this, using the formula that determines the reachability of the $P\&R$ property, we perform backward algebraic modeling using a backward predicate transformer:
$pt^{-1}\ (R, Post\ (a)) = R1$

Carrying out the modeling will reach the initial state from which the property R is reached. This will be a subset of the initial state from which forward algebraic modeling was performed. Thus, all other possible scenarios that lead to the property of R can be analyzed. The initial state may be narrowed, depending on the degree of tokenomics nondeterminism. Finally, when all possible scenarios have been covered, the initial values of the tokenomics can be selected from the initial state, which was narrowed by backward simulation.

The number of undefined attributes determines the complexity of this process, so the degree of certainty depends on the values that can be known from the beginning — for example, the number of tokens that were planned to load on the exchange, the initial number of fiat money or tokens, and the distribution of tokens for agents involved in mining.

Case Study

Let us consider the project of deploying an antenna system service in a certain space using the Internet of Things (IoT), which can be connected to these antennas. The project provides coverage fees for agents who purchase antenna equipment. The system was implemented on a blockchain platform where a token was defined for payments and rewards. IoT users have to buy a token on the exchange and pay for network use. To balance the tokenomics, so-called emissions have been identified, which are calculated depending on the amount of traffic. In addition, some part of the tokens that were paid for the use of the network was subject to the so-called "burning", i.e., was withdrawn from circulation.

The total issue of tokens consists of two parts. First is the tokens in circulation, which are on the exchange and at the participants' own, and the second is the part that replenishes the tokens that are in circulation, due to the increase in the number of IoT users. From the network usage fee, part of the tokens goes to pay rewards to antenna owners, and part goes to investors' as profit. To make a profit, the investor must block some amount for which the reward is accrued. The balance between burned tokens and emissions maintains the tokenomic balance.

We consider listing a company on a decentralized exchange, so a new token price will be formed each time the number of tokens bought or sold changes.

The following equations correspond to IoT tokenomics:

```
B0 = (INVESTMENT_STAGE; PRODUCT_STAGE),
INVESTMENT_STAGE = (sellTokens. distributeBudget. (NextRound. INVESTMENT_STAGE +! NextRound)),
PRODUCT_STAGE = exchangeListing. newPrice. SERVICE,
SERVICE = ({ANTHENA_OWNERS_ACTIONS || USERS_ACTIONS || TRADING_ACTIONS};
(nextMonth. SERVICE +! NextMonth. Delta)),
USERS_ACTIONS

        = (buyToken (Users, monthlySerivcePrice * Traffic

        * deviceUsers / tokenPrice) . newPrice. usersPayService (monthlySerivcePrice * Traffic

        * deviceUsers / tokenPrice); emission. newPrice. marketingUsers . compLiquidity),

TRADING_ACTIONS

        = {(sellToken (Traders, Exchange . token * tradingPart

        * Liquidity) . newPrice) || (buyToken (Traders, Exchange. token * tradingPart

        * Liquidity) . newPrice) || (sellToken (Inv, Inv. token

        * saleInvestorCoef) . newPrice) || (sellToken (Miners, Miners. token * sale) . newPrice)},

ANTHENA_OWNERS_ACTIONS = IncreaseAgentsOwners
```

We use the previously discussed formal agent's actions in the equation concerning investments, such as *sellTokens* and *distributeBudget*. They are executed for each round in a loop that is determined by the nextRound action.

The product part (the stage of starting the service) begins with the *exchangeListing* action, which determines the registration of the token on the exchange with the introduction of a liquid cryptocurrency pair. The *newPrice* action is triggered every time the number of tokens on the exchange changes.

newPrice: ((1) −> "$NewTokensPrice$" ($tokenPrice\ =\ Exchange.fiat\ /\ Exchange.token$))

In the *SERVICE* equation, we have parallel synchronized behaviors that contain actions that determine the purchase of new antennas, the actions of IoT users, and the actions of traders. The synchronization of parallel composition is determined by curly braces. All of these behaviors occur in a loop lasting a certain number of months of the studied part of tokenomics.

Users in the behavior *USERS_ACTIONS* — buy the number of tokens needed to pay for traffic according to their new amount and traffic consumed. The parameters of the action determine the number of tokens. Users pay these tokens for the service that, in turn, is distributed among investors, miners, and other players according to the share of their participation—by money or activities. The share is determined by certain constants*: INVESTOR_ PROFIT, MINER_PROFIT*, and *SUPPORT_PROFIT*. The action that determines the payment of traffic and the distribution of rewards is a function of mining. After allocating the share of rewards, some parts of the tokens are burned, and the emission action replenishes the number of tokens on the exchange according to the amount of traffic.

For *TRADING_ACTIONS* actions, we use the trading ratio *tradingPart* in the total turnover of tokens, determining its possible values by inequality. The traders' activities are in accordance with the value of liquidity, taking into account the limitations of the set of scenarios, which are considered in accordance with the historical data of trading.

The actions of miners *ANTHENA_OWNERS_ACTIONS* are determined by the purchase of new antenna equipment in accordance with the increase in their number.

When having a fully formalized tokenomics project, it is necessary to determine its initial parameters, such as distribution of the share of investors and miners— *INVESTOR_PROFIT, MINER_PROFIT*, as the largest part of profits that affect tokenomics and determine the size of emissions and the number of "burned" tokens.

Let us define these attributes as symbolic or arbitrary, and start forward algebraic modeling. In addition to these attributes, as symbolic attributes, we have the attribute *Liquidity, deviceUsers*, and the attribute that determines the number of miners. The author of the project can determine the desired limits of the values of these attributes. We obtained a set of states through modeling. The states of the general environment at each modeling step will be the formulas *Si*, which depend on the symbolic attributes $x1, x2, \ldots$: $S_1(x1, x2, \ldots), S_2(x1, x2, \ldots) \ldots S_n(x1, x2, \ldots)$

In step n we obtained a formula that must be investigated for the desired properties. Let this be the desired token price within some limits:

$P1 <= tokenPrice <= P2$

This price range is reachable in step n if the conjunction $S_n(x1, x2, \ldots)\ \&\&\ (P1\ <=\ tokenPrice\ <=\ P2)$ is satisfiable.

Using the appropriate solvers, we can prove this fact. If the formula is satisfiable, backward algebraic modeling should be performed to determine the initial formula, if possible, by simplifying it. In the process of backward modeling, we obtained the sequence:

$S_n(x1, x2, \ldots), S_{n-1}^{-1}(x1, x2, \ldots), S_{n-2}^{-1}(x1, x2, \ldots), \ldots\ S_1^{-1}\ (x1, x2, \ldots)$

Next, we perform the operation of a concretization in formula $S_1^{-1}\ (x1, x2, \ldots)$, i.e., we look for those values of symbolic attributes at which the formula is true. These values will be valid for the reachability of the property that we tested. We do not provide formulas for this project to save space, as the formulas can reach several pages. By having the specific values of the attributes, it is possible to build a chart of one of the possible tokenomics scenarios for the token price (Fig. 3).

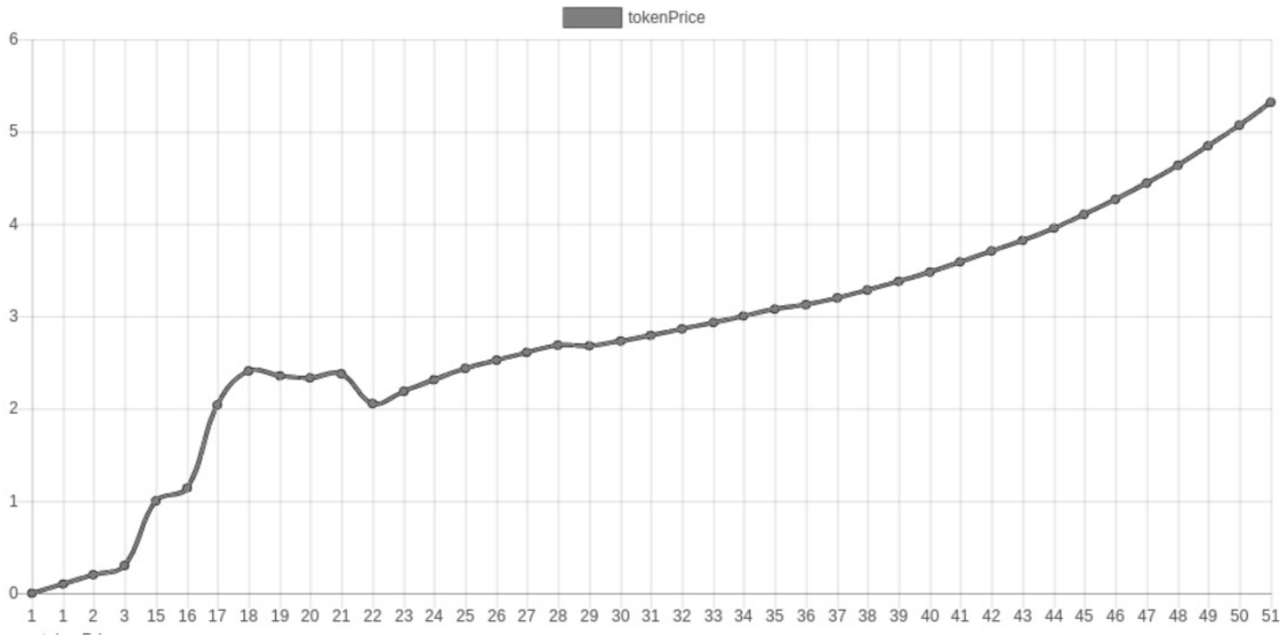

Fig. 3. Chart of token price change during the researched part of the project.

The equilibrium property is determined by the desired attribute constraints or by their time convergence, $T(x)$. The formula that determines convergence can be represented in different forms. One is the gradual reduction of the maximum fluctuation interval, such as the token price, on successive time intervals. This property is expressed by the formula presented below:

Forall (j: int) (1 <= j <N / INT) && (Amp (j)> = Amp (j + 1)) && Forall (i: int) (2 <= i <= INT && Apm (j) == max (tokenPrice (T (j + i - 1)) - tokenPrice (T (j + i))),

where N is the total number of time intervals, and INT is the length of the selected interval in time units. We use the max macro in algebraic language, which determines the maximum expression's value.

To test, it is necessary to consider the satisfiability of the conjunction of the equilibrium formula and the final state formula, taking into account that all the values of the token price are stored in the corresponding functional symbol $tokenPrice(k)$.

The reachability of the undesirable property of centralization is determined by the uneven distribution of tokens among investors, traders, or other groups of agents. Denoting by M the critical number of agents that have tokens, and by d the critical proportion of tokens that violates decentralization, we have the following formula:

Exist (j1, j2,… jM: int) (Inv (j1) .token + Inv (j2) . token +… + Inv (jM) .token >= d * totalToken).

The critical amount M must be small enough, for example, from 1 to 3.

If this formula is satisfiable in algebraic modeling, it means that there is a scenario that leads to centralization. You can find the specific initial attributes of this scenario and show the corresponding specific case.

Conclusion

We used behavioral algebra to consider the formal semantics of agent interactions in a decentralized system project. This formalization shapes the behavior of the token and its life cycle and allows the use of formal methods to test the properties of tokenomics.

The authors used the technology in the analysis of tokenomics for Internet of Things projects, education projects [13, 14], and other projects relating to services, including finance. Creating tokenomic models gives us the possibility of considering charts of token behavior and token distribution between agents. However, as the practice has shown, it is complex to select initial parameters manually, and unforeseen scenarios can occur even when considering a limited set of attribute values.

Therefore, the use of backward algebraic modeling and other formal methods is extremely useful for creating tokenomics projects. It should be noted that computations in tokenomics can be quite complex for automatic solvers, and in this case, the support of mathematicians is needed to simplify the formalization or approximation of calculations. The authors have developed a set of actions and behaviors from which the tokenomics model can be compiled. This can be created by the tokenomics constructor [3] or by the special model creator program [4]. However, these actions are based on the studied projects, and therefore there may be cases when they are not enough and need to create new ones. Therefore, the study of possible behaviors continues and generalizes, and future versions of the tokenomics constructor will have more opportunities for design.

The study of properties by formal methods is also important for projects, as previous methods, such as simulation, do not give an accurate conclusion about the performance of properties or the prevention of undesirable behaviors. The use of modern solvers has made it possible to provide evidential proof of the properties of tokenomics (both safety and liveness properties).

The formalization of trading elements makes it possible to predict a set of scenarios based on historical data, which are also presented in terms of behavior algebra and used at the stage of algebraic modeling. The next step is to generate a smart contract code from the model. In this case, as we have a studied model, smart contracts will be generated in such a way as to ensure reliability and resistance to malicious actions.

References

1. Letichevsky, A.: Algebra of behavior transformations and its applications. In: Kudryavtsev, V. B., Rosenberg, I. G., (eds.) Structural Theory of Automata, Semigroups, and Universal Algebra, NATO Science Series II. Mathematics, Physics and Chemistry, vol. 207, pp. 241-272. Springer (2005). https://doi.org/10.1007/1-4020-3817-8_10.
2. Letichevsky, A., Gilbert, D.: A Model for Interaction of Agents and Environments. In: Bert, D., Choppy, C., Mosses, P.D. (eds) Recent Trends in Algebraic Development Techniques., WADT 1999, LNCS, vol. 1827, pp.311-328 (1999). https://doi.org/10.1007/978- 3-540-44616-3_18.
3. Tokenomics formal methods and Tokenomics constructor, https://garuda.ai/ [Accessed: 2022/02/16]
4. Insertion model creator site, https://rd.litsoft.com.ua/, last accessed 2022/02/16.
5. Modeling site, https://arxiv.org/ftp/arxiv/papers/1907/1907.00899.pdf [Accessed: 2022/02/16]
6. cadCad site: https://github.com/cadCAD-org/cadCAD [Accessed: 2022/02/16]
7. Tokesim site: https://github.com/tokesim/tokesim#about-the-project [Accessed: 2022/02/16]
8. Guo, C., Zhang, P., Lin, B., Song, J. A: Dual Incentive Value-Based Paradigm for Improving the Business Market Profitability. Blockchain Token Economy. Mathematics 2022, 10, 439. https:// doi.org/10.3390/math10030439.
9. Cong, Lin William, Ye, Li, Neng, Wang: Tokenomics: Dynamic adoption and valuation. The Review of Financial Studies 34(3), 1105-1155 (2021).
10. Pardi, A-L, Paolucci, M. A.: Chemical Analysis of Hybrid Economic Systems—Tokens and Money. Mathematics 9(20):2607 (2021). https:// doi.org/10.3390/math9202607.
11. Letychevskyi, O.: Two-Level Algebraic Method for Detection of Vulnerabilities in Binary Code. In: 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), pp. 1074-1077. Metz, France (2019). https://doi.org/10.1109/IDAACS.2019.8924255.
12. Letichevsky, A., Letychevskyi, O., Peschanenko, V., and Weigert, T.: Insertion modeling and symbolic verification of large systems. In: Fischer J., Scheidgen M., Schieferdecker I., and Reed R., (eds.). SDL 2015: Model-Driven Engineering for Smart Cities, pp. 3-18, Springer, Cham, Switzerland (2015). https://doi.org/10.1007/978-3-319-24912-4_1.
13. Letychevskyi, O., Peschanenko, V., Poltoratskyi, M., Tarasich, Y.: Our Approach to Formal Verification of Token Economy Models, In: ICTERI: International Conference on Information and Communication Technologies in Education, Research, and Industrial Applications, pp. 348-363, Shpringer, Cham (2019). https://doi.org/10.1007/978-3-030-39459- 2_16.
14. Letychevskyi, O., Peschanenko, V., Poltoratskyi, M., Kovalenko, P., Mogylko, S. Radchenko, V.: Formal Verification of Token Economy Models. In: IEEE Explore Digital Library, IEEE International Conference on Blockchain and Cryptocurrency(ICB), pp. 201- 204, Seoul, Korea (South) (2019). https://doi.org/10.1109/BLOC.2019.8751318

Received 03.08.2022

About the authors:

Oleksandr Letychevskyi,
Doctor of Physical and Mathematical Sciences
(Mathematical and Software of Computers and Systems)
Head of Department of Theory digital machine № 100
V.M.Glushkov Institute of Cybernetics of the NAS of Ukraine
Kyiv, 03187, Ukraine, Akademika Glushkova Avenue, 40
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,
Doctor of Physical and Mathematical Sciences
(Mathematical and Software of Computers and Systems)
Head of Department of Computer Science and Software Engineering.
Kherson State University, Kherson, Ukraine.
Kherson, 73003, Ukraine, University Street, 27
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
Kherson, 73003, Ukraine, University Street, 27
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

Yuliia Tarasich,
PhD in Information Technology, lecturer
Department of Computer Science and Software Engineering
Kherson State University, Kherson, Ukraine
Kherson, 73003, Ukraine, University Street, 27
Number of scientific publications in Ukrainian publications – 12.
Number of scientific publications in foreign publications – 15.
h-index – 3.
https://orcid.org/0000-0002-6201-4569

Maksym Vinnyk
Candidate of Pedagogical Sciences
Vice-Rector for Financial and Economic and Scientific
and Pedagogical Work, Associate Professor of Chair
of Computer Science and Software Engineering
Kherson State University, Kherson, Ukraine
Kherson, 73003, Ukraine, University Street, 27
Number of scientific publications in Ukrainian publications – 15.
Number of scientific publications in foreign publications – 13.
h-index – 4.
https://orcid.org/0000-0002-2475-7169

Place of work:

Oleksandr Letychevskyi
Department of Theory digital machine № 100
V.M.Glushkov Institute of Cybernetics
of the NAS of Ukraine
Kyiv, 03187, Ukraine, Akademika Glushkova Avenue, 40
Ph.: (+38) (044) 526-20-08
Fax: (+38) (044) 526-41-78
e-mail: incyb@incyb.kiev.ua

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

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

Yuliia Tarasich,
Department of Computer Science and Software Engineering
Kherson State University, Kherson, Ukraine
Kherson, 73003, Ukraine, University Street, 27
Ph.: +(38) (0552) 326731
Fax: +(38) (0552) 492114
e-mail: office@ksu.ks.ua

Maksym Vinnyk
Department of Computer Science and Software Engineering
Kherson State University, Kherson, Ukraine
Kherson, 73003, Ukraine, University Street, 27
Ph.: +(38) (0552) 326731
Fax: +(38) (0552) 492114
e-mail: office@ksu.ks.ua

Прізвища та ініціали авторів і назва доповіді українською мовою:
Летичевський О.О., Песчаненко В. С., Полторацький М.Ю., Тарасіч Ю.Г., Вінник М.О.
Формальна семантика та аналіз властивостей токоекономіки

Прізвища та ініціали авторів і назва доповіді англійською мовою:
Letychevskyi O.O., Peschanenko V.S., Poltoratskyi M. Yu., Tarasich Yu. H., Vinnyk M.O.
Formal semantics and analysis of tokenomics properties

Відповідальний виконавець – Юлія Тарасіч,
+380980028534, yutarasich@gmail.com