https://doi.org/10.31891/2307-5732-2026-361-50
УДК 004.021

SHAPYRO OLEKSII
Kharkiv National University of Radio Electronics
https://orcid.org/0009-0005-3539-266X
e-mail: oleksii.shapyro@nure.ua
GOTSULYAK KATERYNA
Kharkiv National University of Radio Electronics
https://orcid.org/0009-0001-9032-4249
e-mail: kateryna.horishnia@nure.ua

METHODS FOR DECOMPOSITION OF CONJUNCTIVE FORMULAS IN FINITE PREDICATE ALGEBRA

In this paper the subject matter is the methods of decomposing conjunctive formulas in the algebra of finite predicates and the possibility of using dependency structures of relational data to build compact logical networks. The work examines the interrelation between operations of relational algebra and predicate algebra, which makes it possible to form a mathematical framework for the rational representation of complex predicate models. The relevance of the study is determined by the need to create high-performance solutions for linguistic information processing and to optimize parallel computational structures. The goal of the work is to develop an effective method of binary predicate decomposition that reduces the number of auxiliary variable values and ensures the transformation of multidimensional predicate formulas into a compact system of binary relations suitable for hardware implementation in the form of logical networks. This approach improves model performance and reduces hardware complexity. To achieve this goal, the following tasks were carried out: analysis of functional, multivalued, and connection dependencies; determination of the conditions for the existence of quantifier-conjunctive decomposition; and investigation of cases in which a predicate can be represented as a system of binary formulas. The limitations of Cartesian decomposition were identified, and it was shown how dependency structures eliminate these drawbacks, forming more compact and structurally justified models. The research methods include the apparatus of finite predicate algebra, the theory of relational dependencies, techniques for constructing auxiliary predicates, variable elimination using existential quantifiers, and modeling processes in the form of logical networks. Special attention is paid to analyzing the structural properties of formulas that influence decomposition efficiency. The results of the research show that the proposed method allows obtaining more compact binary models in comparison to the Cartesian approach. Predicate decomposition applied to a morphological inflection model has confirmed a significant increase in efficiency during hardware implementation of the logical network and improved performance indicators. In the conclusions, it is emphasized that the use of dependency structures is an effective tool for optimizing formal models and opens opportunities for further automation of predicate-structure construction in tasks of logical analysis and knowledge processing.

Keywords: *predicate algebra, logical networks, binary decomposition, Cartesian decomposition.*

ШАПИРО ОЛЕКСІЙ, ГОЦУЛЯК КАТЕРИНА
Харківський національний університет радіоелектроніки

МЕТОДИ ДЕКОМПОЗИЦІЇ КОН'ЮНКТИВНИХ ФОРМУЛ В АЛГЕБРІ СКІНЧЕННИХ ПРЕДИКАТІВ

У даній статті предметом дослідження є методи декомпозиції кон'юнктивних формул в алгебрі скінченних предикатів та можливість використання структур залежностей реляційних відношень для побудови компактних логічних мереж. У роботі розглядається взаємозв'язок операцій реляційної та предикатної алгебр, що дозволяє сформувати математичний апарат для раціонального подання складних предикатних моделей. Актуальність дослідження зумовлена потребою створення високопродуктивних рішень для обробки лінгвістичної інформації та оптимізації паралельних обчислювальних структур. Метою роботи є розроблення ефективного методу бінарної декомпозиції предикатів, що зменшує кількість значень допоміжних змінних та забезпечує перетворення багатовимірних предикатних формул у компактну систему бінарних відношень, придатних до апаратної реалізації у вигляді логічних мереж. Такий підхід дозволяє підвищити швидкодію моделей та зменшити їх апаратну складність. Для досягнення мети виконано такі завдання: аналіз функціональних, багатозначних і зв'язкових залежностей, визначено умови існування кванторно-кон'юнктивної декомпозиції та досліджено випадки, коли предикат може бути поданий як система бінарних формул. Виявлено обмеження картезіанської декомпозиції та показано, як структури залежностей дозволяють усунути ці недоліки, формуючи більш компактні та структурно обґрунтовані моделі. Методи дослідження включають апарат алгебри скінченних предикатів, теорію залежностей реляційних відношень, методи побудови допоміжних предикатів, елімінацію змінних та моделювання процесів у вигляді логічних мереж. Особлива увага приділяється аналізу структурних властивостей формул, що впливають на ефективність декомпозиції. Результати дослідження показують, що запропонований метод дозволяє отримати компактнішу структуру бінарних моделей порівняно з картезіанським підходом. Декомпозиція предикатів у моделі словозміни підтвердила значне підвищення ефективності при апаратній реалізації логічної мережі та покращення показників продуктивності. У висновках підкреслено, що використання структур залежностей є дієвим інструментом оптимізації формальних моделей та відкриває можливості для подальшої автоматизації побудови предикатних структур у задачах логічного аналізу та обробки знань.

Ключові слова: *алгебра предикатів; логічні мережі; бінарна декомпозиція; Декартова декомпозиція.*

Стаття надійшла до редакції / Received 18.11.2025
Прийнята до друку / Accepted 11.01.2026
Опубліковано / Published 29.01.2026

Introduction and problem statement

Among the many formalisms applicable to one degree or another to the tasks of processing informal information, the most appropriate is the use of predicate algebra, systems of equations which are technically

implemented in the form of a logical network. One of the advantages of this approach is its direct applicability to all the following types of problems, which is provided by the declaratives of the logical network as a method of solving systems of predicate equations. The notation of the predicate equations itself becomes possible due to the algebra of predicates [1].

Another advantage of logical networks is the wide parallelization of calculations, which, provided the correct construction of the task model guarantees high efficiency.

On the example of processing test information by artificial intelligence, we can consider the general classification of tasks that arise in the process of working with primary informal information [2]:

- analysis (for example, recognition - obtaining from informal information specific parameters necessary for the application of some formalism);
- normalization - bringing information to some reference form, which is relevant in the search for information;
- synthesis - the expression of the internal representation of information stored in accordance with the formal requirements in a form adapted for human perception;
- mixed tasks.

In addition, the brain-like computer, based on the technology of logical networks, is close in structure to the structure of the human brain, which gives reason to hope for the creation of information technology-based systems based on the capabilities of human.

Despite the availability of formal means for the description of relations within predicate algebra, there are no methods that guarantee a compact and lossless decomposition of predicates into binary components without inflating auxiliary variables. In practice, widely used Cartesian decompositions introduce auxiliary predicates with large domains, making hardware/low-level implementations of logical networks resource-intensive and slow. Nor is there an agreed-upon criterion for when a task's dependency structure ensures a direct binary decomposition without further loss of interpretability. Consequently, an approach is needed that, in declarative terms (predicate algebra), formulates the conditions for correct decomposition and yields an operational scheme (a logical network) that minimizes auxiliary resources.

Analysis of research and publications

The study of categories quickly became an independent abstract discipline and is now an important branch of pure mathematics. In addition, it influenced the conceptual foundations of mathematics and the language of mathematical practice. It offers elegant and powerful tools for expressing connections between major branches of mathematics and provides mathematicians with tools for mathematical research that occupy more and more space in the arsenal of mathematics [3].

However, to date, the theory of categories still does little to solve problems such as the description of the mechanisms of human intelligence, such as the development of artificial intelligence, the creation of parallel computers, intelligent radar systems, and so on.

Therefore, it is necessary to establish a connection between the theory of categories and modeling of systems based on artificial intelligence using an intermediate field of knowledge - the algebra of finite predicates, by which a predicate interpretation of the category can be built. Based on this interpretation, elements of the theory of the modified category for the processes of modeling the functions of artificial intelligence are developed.

A distinctive feature of artificial intelligence tasks is that they are difficult to formalize, necessary to enable the automation of their solutions. Thus, it is expedient to study the universal tool of informal information processing – the human brain and to model some of its functions related to the tasks based on modern technologies. The essence of the approach is that human intelligence is seen as logic in action, as some material embodiment of the mechanism of logic. Works on logic algebraization were performed [4].

Schematic implementation of formulas describing algebraic-logical structures leads to characteristic engineering networks that have not been used before, and which are called logical networks. When comparing these networks with the main types of neural structures, a deep similarity in the structure of technical and biological structures is revealed. Based on this similarity, we can determine the functions of different types of neural structures and describe in exact mathematical and technical terms the principles of brain function [5].

The main thing in this method is the movement from top to bottom: from general system considerations to algebraic-logical structures, and from them to logical networks, which are then identified with biological neural structures.

Unlike the traditional information technology algorithmic approach, the declarative task model is not an algorithm. The task model presented in a declarative form is usually a set of facts, knowledge of facts and rules, according to which actions are taken on facts that allow to obtain new knowledge based on old ones [6].

A logical network is a method for solving systems of predicate equations and has descriptive capabilities of predicate algebra. The property of a logical network is the parallel execution of all elementary logical operations – that is, the maximum possible at the logical level of parallel calculations. This approach guarantees high efficiency in solving problems that are reduced to a logical conclusion or to the solution of logical equations.

To formalize the tasks of processing informal information using logical networks, it is necessary to explore the possibilities of rational application of the algebra of finite predicates. To this end, it is necessary to model the most common tasks from different areas of the subject area. Examples include areas such as the task of modeling the intelligent recognition mechanisms of inconspicuous radar objects [7].

The mathematical tools developed for the decomposition of relations and predicates are designed primarily to create logical networks, which, in turn, are the basis for building a parallel-action computer. The mathematical results

of the work can be used in automatic language processing systems, computer-aided design systems for information systems. Logical networks can be effective in creating specialized parallel-action devices designed to simulate complex computing systems, for example, a system of complete adequate morphological analysis of natural language texts.

Mathematical model suitable for implementation in the form of a logical network should be represented by a system of binary equations, i.e., each equation of such a system should have two variables. This form of presentation is a logical network model. In this case, the system of binary equations should be as compact as possible in order to save hardware resources when implementing the model.

Converting a model into a compact binary form requires conjunctive decomposition of predicates. In the area of the decomposition apparatus, a shortage of justified methods, that does not require checking the equivalence of the obtained result to the original predicate, and at the same time rational methods that allow obtaining a compact binary representation for a given formal model. When developing models of logical networks for processing linguistic information, developers mostly used their own intuitive methods, rather than strict, justified methods [8]. The logical network models obtained in this way need experimental verification of their adequacy to the described object.

Almost all available formal means of decomposition of predicates in these works do not consider the relationship between variables. Taking these features into account in the structure of a predicate allows one to decompose it in a more rational way. Meanwhile, the theory of relational databases offers a means of decomposing relations in the form of statements about the dependencies between attributes (in the section on normalizing relations). Given the deep connection between relations and predicates, the means of decomposition of relations can be applied to solve the problem of decomposition of predicates. As an analysis of the literature has shown that such tools are currently not available in the apparatus of predicate algebra.

Formulation of the goals of the article

The goal of the work is to develop an effective method of binary predicate decomposition that will reduce the number of auxiliary variable values and ensure the transformation of multidimensional predicate formulas into a compact system of binary relations suitable for hardware implementation in the form of logical networks. This approach should improve model performance and reduces hardware complexity.

To achieve this goal, the following tasks must be completed: analysis of functional, multivalued, and connection dependencies; determination of the conditions for the existence of quantifier-conjunctive decomposition; and investigation of cases in which a predicate can be represented as a system of binary formulas.

Exposition of the main material

The only difference between category theory and predicate algebra is that the former moves from top to bottom, aims to learn higher logical mechanisms and therefore uses as starting points the concept of a record level of commonality. The second, starting from the needs of informatization, moves in the study of the same logic of thinking from the bottom up. If it were possible to give a convincing interpretation of the concepts formed by the theory of categories and the methods developed by it in terms of predicate algebra, i.e., concretizing, to bring them closer to informatization, it would significantly enrich the tools of predicate algebra. It is the algebra of predicates that we are going to use as such an intermediate field of knowledge.

A logical network copies a person's actions, but with the only difference that a person acts sequentially and a network acts in parallel. The network works in cycles. In the first half-bar of the i-th bar, the network for each of its equations of the form $K(x, y) = 1$ (K is the relation given by such equation) finds:

1. According to the known knowledge of $P_i(x)$ about the value of the variable x at the beginning of i- th bar knowledge $Q'_i(y)$ about the value of the variable at the end of the i-th bar;

2. According to the known knowledge of $Q_i(y)$ about the value of the variable b at the beginning of the i-th bar, knowledge of $P'_i(x)$ about the value of the variable x at the end of the i-th bar.

Mathematically, these two operations are expressed by formulas (1, 2):

$$x \in A\big(K(x,y)P_i(x)\big) = Q'_i(y), \quad (1)$$

$$x \in B\big(K(x,y)Q_i(y)\big) = P'_i(x), \quad (2)$$

where A and B are the variable areas of x, y.

In the second half of each bar, the network finds the common part $P_{i+1}(x)$ of all knowledge $P_{i1}(x), P_{i2}(x), \dots, P_{il}(x)$ about the value of each of its objective variables x parties, which come through the branches of the network from all to the pole x. This operation is expressed as follows (3):

$$xP_{i1}(x) \wedge P_{i2}(x) \wedge \dots \wedge P_{il}(x) = P_{i+1}(x), \quad (3)$$

The obtained knowledge $P_{i+1}(x)$ is then used as the state of the pole x at the initial moment of the $i + 1$st cycle. The symbol l denotes the number of branches approaching the pole x. Before the beginning of the $i + 1$st bar, knowledge of the set $P_{i+1}(x)$ is formed in each pole, which is always included in the knowledge of the set $Pi(x)$, which was kept in the same pole at the beginning of the i-th bar. Thus, the only result of the logical network is the refinement of the knowledge contained in all its poles in accordance with the original data.

With the introduction of the theory of categories, along with the usual concept of category, there was a more general concept of objectless category.

Let M be some set. Its elements, denoted by the symbols $f, g, h, \dots$, are called morphisms. Suppose, moreover, that some, in the general case, partial operation $fg = h$ is given, which acts with $M \times M$ in M. It is called the multiplication of the morphisms f and g. The morphism h is called the product of the morphisms f and g. Any morphism $e \in M$ is called singular or identical if it satisfies the following conditions:

- for any $f \in M$ for which there is a product $fe \in M$, the equality $fe = f$;
- for any $f \in M$ for which there is a product $ef \in M$, the equality $ef = f$ holds.

This definition tacitly allows the existence in the category of many units. It is the presence of many units (and only this) that distinguishes a category from other known algebraic structures. The singular morphisms ef and $e'f$ are named according to the right and left for the morphism $f \in M$, if $fef = f$ and $fe'f = f$. For any morphism $f \in M$ there are a single right and a single left morphisms. This statement is called the law of identity. Thus, for each $f \in M$ there is a single right unit ef and a single left unit $e'f$, such that $fef = fe'f = f$.

Multiplication of morphisms is associative: $(fg)h = f(gh)$ for any $f, g, h \in M$ for which there are products: $(fg)h$ and $f(gh)$. The set M, which contains at least single morphisms, and on which the operation of multiplication of morphisms with the above properties is called an object-free classical category K. Write $M = Mork, f \in M, f \in Mork$. $Mork$ is the set of all morphisms of category K. If $f \in Mork$, then we say that the morphism f is a K-Morphism.

Next, we will talk about the development of new predicate decomposition tools based on statements about dependencies from the normalization of relational relations. At the beginning, an exact definition of the decomposition procedure with respect to relational relations and predicates is given; statements about dependencies that allow decomposition of relational relations are considered; their generalizations are proposed. Based on the relations between relational algebra and predicate algebra, dependency statements are transformed into new predicate decomposition tools.

A predicate in the basis of a disjunctive conjunctive predicate algebra (DCPA) is usually regarded as an analytic notation of a relation. Therefore, a predicate decomposition is a decomposition of a relation at the analytical level (simply manipulating formulas), when not the relations themselves are processed, but the corresponding predicate formulas.

We clarify the concept of decomposition, as it is understood in relational databases. To do this, we give the concept of the dependence of the compound, which is used in the theory of dependencies.

Let $\Gamma = \{X_1, \dots, X_l\}$ be a subset family of sets H. If $H = \bigcup_{i=1}^{l} X_i$, then Γ is called overlapping of set H. An overlapping Γ is called a partition of a set H if the sets $X_1, \dots, X_l$ are pairwise disjoint.

Let R be some relation, and the family of sets $\{X_1, \dots, X_l\}$ forms the covering of the set $at(R)$. The relationship R satisfies the connection $[X_1, \dots, X_l]$ dependencies if

$$R = R[X_1] \circ R[X_2] \circ \dots \circ R[X_l], \tag{4}$$

Expression (4) is the so-called algebraic restriction of integrity.

Other types of dependencies that are used in the theory of dependencies – functional and multi-valued dependencies – are special cases of the connection dependency. Therefore, any dependence made $R \in Re(T)$ in allows us to represent R in the form of a natural connection of their projections (4). In the theory of relational databases, it is usually this procedure that is called *decomposition of* R. Based on the expression (4), decomposition itself is carried out by designing the initial relation for different groups of its attributes. This suggests that the projection operator is the decomposition operator, while the composition (reassembly) operator is a natural join.

However, splitting a relation can be carried out in different ways: along with projection, for example, sampling can be used as a decomposition operator. Then the composition operator will no longer be a natural join, but a join operation. When in database theory we are talking about the decomposition of relations, then almost always we mean a decomposition of the form (4). Other types are practically not used. Nevertheless, for accuracy, the decomposition of relations in the form (4) will sometimes be called *projection-connective decomposition*.

A *predicate decomposition of* P is the derivation of some predicates $P_1, P_2, \dots, P_l$ based on a predicate P with the possibility of its reverse recovery from these predicates, which is called the *predicate composition of* P. An important particular case of decomposition is the so-called *binary predicate decomposition of* P, characterized in that each of the predicates $P_1, P_2, \dots, P_l$ resulting from the decomposition of the predicate P has exactly two significant arguments. If the conjunction serves as the composition (restoration) operator, then decomposition is called *conjunctive*.

The predicate analogues of relational projection operations are existence quantifiers. Then the predicate analogue of the projection-connective decomposition will be the quantifier-conjunctive form of decomposition.

Let $T \subseteq V \times U$ be a spanning tree with a finite set of variable names $V = \{x_1, x_2, \dots, x_n\}$, $S = \text{Dom } x_1 \times \text{Dom } x_2 \times \dots \times \text{Dom } x_n$ Let P be some predicate from $\text{Pre}(S)$, and the family of sets $\{X_1, X_2, \dots, X_l\}$ form a covering of the set V. We say that a predicate P *satisfies conjunction dependences* $\wedge\{X_1, X_2, \dots, X_l\}$ if

$$\exists Y_1(P) \wedge \exists Y_2(P) \wedge \dots \wedge \exists Y_l(P) = P, \tag{5}$$

where $Y_i = V \backslash X_i$, $i = 1, \dots, l$.

Representation of the predicate P in the form of (5) will be called a *quantifier-conjunctive decomposition*.

Some methods for decomposing predicates boil down to replacing the original predicate P with some auxiliary predicate P', which is built based on the original predicate P by introducing one or more auxiliary variables $y_1, y_2, \dots, y_k$. The idea of these methods is that the auxiliary predicate P' is constructed in advance so that it satisfies some required quantifier-conjunctive dependence. Then, using the existential quantifiers, the predicate P' is decomposed directly. This predicate is related to the original predicate P by equality

$$P = \exists y_1 \exists y_2 \dots \exists y_k(P'), \tag{6}$$

Although in fact a quantifier-conjunctive decomposition of the auxiliary predicate P' is performed, the original predicate P is restored using equality (3). Therefore, they say that the predicate is decomposed.

Thus, the predicate decomposition can be performed either without the help of auxiliary variables, or with the help of them. In the first case, decomposition is called *equivalent*, in the second – *nonequivalent*. So, the decomposition

of the mentioned predicate P' is equivalent, and the decomposition of the original P is nonequivalent. Quantifier conjunctive decomposition is equivalent.

Let us consider the concepts of dependencies and some statements regarding them. The latter allow for a given relationship to resolve the question of the admissibility of a projection-connective decomposition. Consider special cases: functional and multi-valued dependencies.

Let a relation $R \in Re(T)$ be given and X, Y be arbitrary subsets of the set $\text{at}(R)$. It is said that Y *depends functionally* on X if for any two functional data $f_1, f_2 \in R$ from equality $f_1[X] = f_2[X]$ follows the equality $f_1[Y] = f_2[Y]$. If Y depends functionally on X, it is written $X \to Y$ and said that the relation R satisfies the functional dependence $X \to Y$ or that the functional dependence $X \to Y$ is fulfilled in the relation R

It follows directly from the definition that if the dependence $X \to Y$ is satisfied in R, then it is executed in any of its projections containing many attributes $X \cup Y$.

If $R \in Re(T)$ is an arbitrary relation, then the set of pairs (X, Y) such that $X \to Y$ in R is denoted by $\text{FD}(R)$. It is called the *structure of the functional dependencies of the relationship* R.

Let be an arbitrary relation R. It is proved that the structure of functional dependencies $\text{FD}(R)$ has the following properties (X, Y, Z, W is a subset of a set $\text{at}(R)$).

$$\text{If } X \supseteq Y, \text{ then } X \to Y. \tag{7}$$

$$\text{If } X \to Y \text{ and } Y \cup Z \to W, \text{ then } X \cup Z \to W. \tag{8}$$

For a deductive system defined by rules (7) - (8), the problem of derivability or logical sequence is algorithmically solvable, which means that the analysis of the structures of functional dependencies can be carried out automatically. There are algorithms that allow for a given set of functional dependencies Σ to find the set of all dependencies $\phi(\Sigma)$ that logically follow from the set according to the rules (7) - (8). In addition, there is an algorithm to determine whether some functional dependence σ is a consequence of many dependencies Σ or not. To analyze relationships with functional dependencies, all these algorithms are based on the apparatus of Boolean functions.

Statement 1. Haag's theorem. Let R be some relation from $Re(T)$, and the family of sets $\{X, Y, Z\}$ forms a partition of the set $at(R)$. If $X \to Y$ in R, then

$$R = R[X \cup Y] \circ R[X \cup Z], \tag{9}$$

In other words, Haag's theorem says that functional dependence $X \to Y$ in relation $R\{X, Y, Z\}$ implies connection dependence for it.

Let us consider the concept of multi-valued dependence. Let a relation $R \in Re(T)$ be given and let X and Y be arbitrary subsets of the set $\text{at}(R)$. It is said that X ambiguously determines Y if $R = R[X \cup Y] \circ R[X \cup Z]$, where $Z = \text{at}(R) \backslash (X \cup Y)$. If X defines Y ambiguously, it is written $X \twoheadrightarrow Y$ and said that the relation R satisfies the ambiguous dependence $X \twoheadrightarrow Y$ or that the ambiguous dependence $X \twoheadrightarrow Y$ is fulfilled in the relation R.

The above definition shows that a multi-valued dependence is a special case of a connection dependence and is also determined through an algebraic integrity constraint.

If $R \in Re(T)$ is an arbitrary relation, then the set of pairs (X, Y) such that $X \twoheadrightarrow Y$ in R is denoted by $\text{MVD}(R)$. It is called the *structure of multi-valued relationship dependencies* R.

Let R be an arbitrary relation. It is proved that the structure of multivalued dependencies $\text{MVD}(R)$ has the following properties (X,Y,Z,W - subsets of a set $\text{at}(R)$).

$$\text{If } X \supseteq Y, \text{ then } X \twoheadrightarrow Y, \tag{10}$$

$$\text{If } X \twoheadrightarrow Y \text{ and } Y \cup Z \twoheadrightarrow W, \text{ then } X \cup Z \twoheadrightarrow W \backslash (Y \cup Z), \tag{11}$$

$$\text{If } X \twoheadrightarrow Y \text{ and } X \twoheadrightarrow Z, \text{ then } X \twoheadrightarrow Y \cup Z, \tag{12}$$

$$\text{If } X \twoheadrightarrow Y, \text{ then } X \twoheadrightarrow \text{at}(R) \backslash (X \cup Y), \tag{13}$$

Property (13) suggests that multi-valued dependencies always form connected pairs; therefore, the notation is used $X \twoheadrightarrow Y|Z$ to relate R to $\text{at}(R) = \{X, Y, Z\}$ and the dependence $X \twoheadrightarrow Y$. As for functional dependencies, for the deductive system defined by rules (10) - (13), the derivability problem is algorithmically solvable, therefore, the analysis of the structures of multi-valued dependencies can also be carried out automatically using Boolean functions.

Statement 2. Let $R \in Re(T)$ be an arbitrary relation, and X and Y be arbitrary subsets $\text{at}(R)$ of the set $Z = \text{at}(R) \backslash (X \cup Y)$. The relation R satisfies the dependence $X \twoheadrightarrow Y$ if and only if

$$\sigma_{(X=A) \wedge (Z=C)}(R)[Y] = \sigma_{X=A}(R)[Y], \tag{14}$$

for every $\{X: A\} \in R[X]$ and every $\{Z: C\} \in \sigma_{X=A}(R)[Z]$.

Equality (14) shows, that for every $\{X: A\}$ from $R[X]$ set $\sigma_{(X=A) \wedge (Z=C)}(R)[Y]$ does not depend on $\{Z: C\} \in \sigma_{X=A}(R)[Z]$. It means that multivalued dependency $X \twoheadrightarrow Y$ in R defines the displaying of $\{X: A\} \to \sigma_{(X=A) \wedge (Z=C)}(R)[Y]$ from $R[X]$ to Boolean $P(R[Y])$ that can be related as multivalued function from $R[X]$ in $R[Y]$.

This statement in less formal form is chosen as a definition to multivalued dependency by Date. Except for the notation, this definition is as follows.

Let there be some relation $R \in Re(T)$, $X, Y, Z \subseteq \text{at}(R)$ and $X \cup Y \cup Z = \text{at}(R)$, and the sets X, Y, Z do not intersect in pairs. Y multivalued depends on X if and only if the set of values Y corresponding to a given pair (value X and value Z) of the ratio R depends only on X but does not depend on Z.

At the same time, Date cites the definition of multivalued dependence adopted in this paper as Feigin's theorem, which is formulated in terms of “if and only if”.

Statement 3. Let $R \in Re(T)$ be an arbitrary relation, and X and Y be arbitrary subsets of the set $\text{at}(R)$, $Z = \text{at}(R) \backslash (X \cup Y)$. The relation R satisfies the dependence $X \twoheadrightarrow Y$ if and only if

$$\sigma_{X=A}(R)[Y \cup Z] = \sigma_{X=A}(R)[Y] \circ \sigma_{X=A}(R)[Z], \quad (15)$$

for every $\{X: A\} \in R[X]$.

Note that the natural combination of two relations that have no common attributes can be viewed as an extended version of the Cartesian product. Equality (15) expresses just such a case. In fact, Statement 3 expresses another way that multivalued dependency can be defined.

According to Statement 3, the presence of a multivalued relationship $X \twoheadrightarrow Y$ in a relation $R\{X, Y, Z\}$ means that its attributes Y and Z are mutually independent: the set of values of each of them depends only on the value of the attribute X. In other words, multivalued dependencies $X \twoheadrightarrow Y|Z$ appear when, for a meaningful group of characteristics X, the set of values of characteristics from groups Y and Z must occur in any combinations with a fixed value of the X-component.

Each functional dependence is a multivalued dependence, i.e. if $X \to Y$ at R, then $X \twoheadrightarrow Y$ at R. Therefore $\mathrm{FD}(R) \subseteq \mathrm{MVD}(R)$. Thus, multivalued dependencies are a generalization of functional dependencies.

However, there is a significant difference between functional and multivalued dependencies. According to (13) dependences $X \twoheadrightarrow Y$ and $X \twoheadrightarrow \mathrm{at}(R)\backslash(X \cup Y)$ always form connected pairs, which is denoted as $X \twoheadrightarrow Y|Z$, where $Z = \mathrm{at}(R)\backslash(X \cup Y)$. Property (13) has no analogue for functional dependencies, therefore, in the general case, they do not form connected pairs.

The following statement generalizes Statement 2, and this time uses the notion of connection dependency.

Statement 4. Let R be some relation from $Re(T)$, and the family of sets $\{X, Z_1, Z_2, \dots, Z_m\}$ forms a partition of the set $\mathrm{at}(R)$. The relation R satisfies the connection $[X \cup Z_1, X \cup Z_2, \dots, X \cup Z_m]$ dependence if and only if

$$\sigma_{X=A}(R)[Z_1 \cup Z_2 \dots \cup Z_m] = \sigma_{X=A}(R)[Z_1] \circ \sigma_{X=A}(R)[Z_2] \circ \dots \circ \sigma_{X=A}(R)[Z_m], \quad (16)$$

for each $\{X: A\} \in R[X]$.

Statement 5. Generalization of Feigin's theorem. Let R be some relation from $Re(T)$, and the family of sets $\{X, Z_1, Z_2, \dots, Z_m\}$ forms a partition of the set $\mathrm{at}(R)$. A relation R satisfies the connection dependency $[X \cup Z_1, X \cup Z_2, \dots, X \cup Z_m]$ if and only if it simultaneously satisfies multivalued dependencies $X \twoheadrightarrow Z_1, X \twoheadrightarrow Z_2, \dots, X \twoheadrightarrow Z_m$.

According to Statement 3, the multivalued dependence $X \twoheadrightarrow Y|Z$ performed in relation to R with $\mathrm{at}(R) = X \cup Y \cup Z$ expresses the mutual independence of attributes Y and Z in relation to R considering that the set of values of each of them is determined by the value of the attribute X. In the same way, Statement 3 allows us to consider the dependence $X \twoheadrightarrow Z_1|Z_2| \dots |Z_m$ performed in R, $\mathrm{at}(R) = X \cup Z_1 \cup \dots \cup Z_m$: it expresses the mutual independence of attributes $Z_1, Z_2, \dots, Z_m$, from each other, given that the set of values of each of them is determined by the value of the attribute X.

According to Haag's theorem, functional dependence acts as a sufficient condition for decomposition of a given relation into two of its projections. Statements 2 and 3 can be considered as criteria for the presence of a certain multivalued dependence in a given relation. Statement 4 is a criterion for the presence of a connection dependence of a certain type, and Statement 5 establishes the equivalence of such a dependence to a group of multivalued dependencies with the same determinant. Due to this, Statement 4 can be viewed as a generalization of Statement 3 to the case of a group of multivalued dependencies.

The functional dependence of the argument $\mathrm{x_n}$ on the arguments $\mathrm{x_1}, \dots, \mathrm{x_{n-1}}$ in the predicate $\mathrm{P(x_1}, \dots, \mathrm{x_n)}$ is expressed by the following formula:

$$\forall x_1 \forall x_2 \dots \forall x_n \forall x_n' (P(x_1, \dots, x_{n-1}, x_n) \wedge P(x_1, \dots, x_{n-1}, x_n') \Rightarrow x_n = x_n'), \quad (17)$$

Any functional dependence of one group of predicate variables $P(x_1, \dots, x_n)$ on another group can be specified either by several formulas of the form (17), or by one such formula in which, after implication, there will be a conjunction of several equality predicates. If X and Y are subsets of predicate variables $P(x_1, \dots, x_n)$ and Y functionally depends on X, then we the predicate $P(x_1, \dots, x_n)$ *satisfies the functional dependence* $X \to Y$.

The multi-valued dependence for predicates can be expressed in terms of the conjunction dependence. We say that a predicate $P \in \mathrm{Pre}(S)$ *satisfies a multi-valued dependence* $X \twoheadrightarrow Y$ if it satisfies the dependency of the conjunction $\wedge\{X \cup Y, X \cup Z\}$, where $Z = V\backslash(X \cup Y)$. In this case the dependence $X \twoheadrightarrow Z$ will also be fulfilled, which together can be denoted as $X \twoheadrightarrow Y|Z$.

As with relational relationships, dependency structures can also be defined for predicates. Let be $P \in \mathrm{Pre}(S)$, $X, Y \subseteq V$. Then the set of pairs (X, Y) such that $X \twoheadrightarrow Y$ in P, is denoted by $\mathrm{MVD}(P)$ and will be called the *structure of the functional dependences of the predicate*. Similarly, the set of pairs such that in, is denoted by and will be called the structure of multivalued dependencies of the predicate.

Apparently, sets $\mathrm{FD}(P)$ and $\mathrm{MVD}(P)$ will have the same properties as dependency structures for relationships. But to verify this assumption, additional research is required, which is beyond the scope of this paper.

Connection dependency properties are derived from the properties of a natural join operation. An analogue of the natural connection of relations is the conjunction of predicates, which, like the natural connection, is idempotent, commutative, and associative. Two other properties of the natural compound are easily extended to the conjunction operation. If we show the fulfillment of the latter for the conjunction, then it can be argued that the dependencies of the conjunction have the same properties as the dependencies of the connection.

The only justified binary predicate decomposition method that guarantees the correctness of the result is the Cartesian decomposition, which allows decomposition of any finite predicates, not just functional ones. Although the disadvantages of this method were discussed earlier, we will consider the Cartesian decomposition as a starting point for finding a more rational method in the sense of saving hardware resources.

To perform the Cartesian decomposition of a predicate $P \in \mathrm{Pre}(S)$, it is necessary to explicitly represent its region of truth. You can use the PDNF for predicate P. Let P_T be the region of truth of the predicate P, $P_T \subseteq S$. Then

$$P(X,Z) = \bigvee_{(a_1,a_2,\dots,a_m,c_1,c_2,\dots,c_n)\in P_T} x_1^{a_1}x_2^{a_2}\dots x_m^{a_m}z_1^{c_1}z_2^{c_2}\dots z_n^{c_n} == \bigvee_{i=1}^{r} x_1^{a_{i1}}x_2^{a_{i2}}\dots x_m^{a_{im}}z_1^{c_{i1}}z_2^{c_{i2}}\dots z_n^{c_{in}}, \quad (18)$$

where r is the number of tuples of the relation P_T (which will be due to the finiteness of the predicate P).

According to the Cartesian decomposition method, an auxiliary variable is introduced with values $B = \{b_1, b_2, \dots, b_r\}$ that are interpreted as unique names of relationship tuples.

Cartesian decomposition $P(X,Z)$ is performed by constructing an auxiliary predicate $Py(X,Z,y)$ and subsequent binary decomposition $Py(X,Z,y)$ using existential quantifiers.

As noted above, the disadvantage of this decomposition is that the variable y takes as many values within the predicate Py and its projections as the number of tuples contains the region of truth of the predicate P. We assume that this is the maximum number of values of the auxiliary variable that is required for binary decomposition of the predicate, since such a quantity is always sufficient. In real language models, this number is usually large. It was previously explained that this complicates the circuitry implementation of such models on a microcircuit. In addition, during Cartesian decomposition, no structural features of the predicate are considered in order to reduce the number of values of the auxiliary variable.

The task is to use a binary function (and other dependencies if they exist in the predicate) to perform binary decomposition either directly using existence quantifiers or with the preliminary introduction of one auxiliary variable with a minimum number of values.

The principle of the proposed method of binary decomposition consists in the methods of constructing auxiliary predicates, based on which with the help of existential quantifiers (by eliminating variables) many binary predicates are formed. The latter are the result of binary decomposition of the original predicate. The construction of auxiliary predicates can be performed in different ways, or it may not be necessary at all. From the point of view of the structure of the initial functional predicate, three different situations are possible that determine the sequence of actions to achieve binary decomposition at a minimum price.

When binary decomposition of a functional predicate $P(X,Z)$ is necessary, firstly, it is necessary to check whether the arguments $\{x_1, x_2, \dots, x_m\}$ of the described function $p: A \to C$ are independent of each other. Mutual independence of variables $x_1, x_2, \dots, x_m$ in a predicate $P(X,Z)$ means that the dependency $Z \twoheadrightarrow x_1|x_2|\dots|x_m$ is fulfilled. To strictly check the predicate $P(X,Z)$ for a dependency $Z \twoheadrightarrow x_1|x_2|\dots|x_m$, it is necessary to perform a disjunctive expansion of the predicate $P(X,Z)$ by variables Z. After that, the form of this decomposition will allow us to establish exactly whether the dependence $Z \twoheadrightarrow x_1|x_2|\dots|x_m$ in the predicate $P(X,Z)$ is fulfilled or not.

If the predicate $P(X,Z)$ satisfies the dependencies $Z \twoheadrightarrow x_1|x_2|\dots|x_m$, then binary decomposition is quite simple. There are two possible situations.

The first situation occurs when a group of variables Z consists of one variable – z. Then an auxiliary variable is not required: binary decomposition $P(X,z)$ is performed using existence quantifiers directly based on a given conjunction dependence, which can be represented as $\wedge\{\{x_1,z\},\{x_2,z\},\dots,\{x_m,z\}\}$. This is the simplest case. A slightly more complicated case is when a group Z consists of two or more variables; then the second situation takes place.

In the second situation, it is necessary to replace the group Z of variables with one auxiliary variable y, and then make the transition to two auxiliary predicates – $P_1(X,y)$ and $P_2(y,Z)$, which are the result of intermediate decomposition of the predicate $P(X,Z)$. The binary decomposition of predicates $P_1(X,y)$ and $P_2(y,Z)$ is carried out using existence quantifiers based on conjunction dependencies $\{\{y,x_1\},\{y,x_2\},\dots,\{y,x_m\}\}$ and $\{\{y,z_1\},\{y,z_2\},\dots,\{y,z_n\}\}$ respectively, which, as will be shown below, are performed by virtue of the construction of these predicates.

When $P(X,Z)$ does not satisfy the dependency $Z \twoheadrightarrow x_1|x_2|\dots|x_m$, then the most difficult situation of the three possibly takes place. Here, the transition to auxiliary predicates $P_1(X,y)$ and $P_2(y,Z)$ also follows, followed by binary decomposition of the latter. However, the method of generating these auxiliary predicates in this case is much more complicated.

The last and most difficult situation from the point of view of binary decomposition of a functional predicate $P(X,Z)$ occurs when the dependence $Z \twoheadrightarrow x_1|x_2|\dots|x_m$ is not satisfied. In this case, an auxiliary variable is also required. It will be introduced after the system of special subsets of the set A is formed. The values of the auxiliary variable in this case will act as the names of these subsets. With this approach, on the one hand, dependencies will be built in auxiliary predicates that will allow their binary decomposition to be performed. On the other hand, the minimum required number of the indicated subsets will be formed, which will minimize the number of values of the auxiliary variable.

As with Cartesian decomposition, the advantage of this algorithm for executing the binary decomposition of functional predicates is the ability to process the formula or relation not in whole but in parts: for each complete inverse image, its minimum parallelepiped coverage is selected separately.

All available methods of nonequivalent decomposition, i.e., decompositions involving auxiliary variables oriented to static predicates or relations. The described method is no exception. Meanwhile, relationships in databases are dynamic objects that are constantly updated. Therefore, nonequivalent decomposition methods are not suitable for use in databases. However, they are good at describing natural language grammar, which (in comparison with industrial or commercial databases) can be considered a static object.

Research results and their discussion

On the way to creating a logical network that simulates a significant part of the natural language, there are several goals that can be described as follows. The first goal is to present the entire model of inflection by the logical network, and not just the individual parts of speech. The next goal is to provide a logical model of a word; this model should accurately express the entire morphology of the language, so the size of such a network seems quite impressive. The main goal is to provide the logical network with a general model of semantic-syntactic processing of phrases, which should cover most of the language.

In order to consistently move towards achieving your goals, you need to start with a model of inflection. So, to date, models of logical networks for the inflectional processing of some nominal parts of the speech of the language are already available: a model of inflections (end declension mechanism) of full non-possessive adjectives, a model of inflections of nouns of substantive declension. Other parts of speech, from existing models of inflectional processing, are not yet presented in the form of a logical network.

All formulas are presented in the basis of the disjunctive conjunctive predicate algebra (DCPA); therefore, the size or length of the formula is determined by the number of predicates of recognition of the subject contained in it. The binary predicate formulas obtained by decomposing Xi, Eta and Phi are written in compact form, which we will call the disjunctive-bracket form. It is in this form that the DCPA binary formulas form the basis of the circuitry implementation. The amount of hardware resources for the implementation of a model presented in the DCPA language is proportional to the number of recognition predicates in it. Therefore, the number of recognition predicates in the formulas of a certain model makes it possible to approximately estimate the amount of hardware resources that will be spent on its implementation.

Predicates serve as a means of describing finite discrete relations; therefore, one can also evaluate the result of decomposition by the size of relations corresponding to the obtained formulas. The size of a finite relationship is determined, like the size of any finite set, by the number of its elements. The dimension of relations (the number of attributes) can be ignored, since only binary relations are to be compared. Therefore, it is proposed for each method to calculate the total number of binary tuples in the obtained relations and, thereby, compare these methods with each other.

Such a calculation will be carried out by calculating the constituent units in the formulas, since the number of these constituents in the PDNF predicate coincides with the number of tuples of the corresponding relation. To move from the disjunctive-bracket form of a formula to its PDNF, you just need to open all the brackets in this formula. Therefore, to determine the number of constituents in the PDNF of a certain formula, it is necessary to calculate how many terms will be contained in the formula after the brackets are opened.

Based on the obtained binary formulas, one can calculate the length of each of them and the number of tuples in the corresponding relation. Cartesian decomposition of predicates Xi, Eta and Phi was not carried out, but the above parameters can be accurately calculated by determining the number of tuples in the relations corresponding to these predicates.

Cartesian decomposition is a rather trivial way to decompose, so the structure of the resulting binary predicates is very simple. Each such predicate is everywhere defined, surjective, and contains an auxiliary variable that uniquely determines the value of another variable. The length of the formula of such a predicate in the minimum disjunctive-bracket form is determined as the sum of the areas of variation of its variables.

Conclusion and perspectives of further development

In this research, the relationship between operations in relational and predicate algebras was examined, enabling a mathematical apparatus for the rational representation of complex predicate models. The objective of the work was achieved: an efficient method for the binary decomposition of predicates was developed, which reduces the domain cardinality of auxiliary variables and transforms multidimensional predicate formulas into a compact system of binary relations suitable for hardware implementation as logical networks. Model throughput was increased and hardware complexity reduced. The analysis encompassed functional, multivalued, and join dependencies; conditions for the existence of a quantifier–conjunctive decomposition were established; and cases were identified in which a predicate can be represented as a system of binary formulas. The limitations of Cartesian decomposition were determined, and it was shown that dependency structures can eliminate these drawbacks, yielding more compact and structurally justified models. Special attention was devoted to structural properties of formulas that govern the efficiency of the decomposition.

The results demonstrate that the proposed method produces a more compact structure of binary models compared with the Cartesian approach. Application to a morphological inflection model confirmed a substantial efficiency gain in the hardware implementation of a logical network and improvements in performance metrics.

The use of dependency structures constitutes an effective instrument for optimizing formal models and opens opportunities for further automation of constructing predicate structures in tasks of logical analysis and knowledge processing.

References

1. Nikitchenko M. S. Logics of general non-deterministic predicates: semantic aspects [Electronic resource] / M. S. Nikitchenko, O. S. Shkilniak, S. S. Shkilniak // Problems in programming. – 2018. – No. 2-3. – P. 031–045. – Mode of access: https://doi.org/10.15407/pp2018.02.031 (date of access: 15.12.2025). – Title from screen.

2. Jurafsky D. Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition with language models [Electronic resource] / Daniel Jurafsky, James H. Martin. – [S. l. : s. n.], 2025. – Mode of access: https://web.stanford.edu/~jurafsky/slp3/. – Title from screen.

3. Fong B. Seven sketches in compositionality: an invitation to applied category theory [Electronic resource] / Brendan Fong, David I. Spivak. – Mode of access: https://arxiv.org/abs/1803.05316. – Title from screen.

4. Shubin I. Formalization and application of algebraic methods in automated intelligent systems [Electronic resource] / Igor Shubin, Stanislav Snisar, Svitlana Litvin // 2021 IEEE 8th international conference on problems of infocommunications, science and technology (PIC S&T), Kharkiv, Ukraine, 5–7 October 2021. – [S. l.], 2021. – Mode of access: https://doi.org/10.1109/picst54195.2021.9772174 (date of access: 15.12.2025). – Title from screen.

5. Neural engineering: computation, representation, and dynamics in neurobiological systems [Electronic resource] // IEEE transactions on neural networks. – 2004. – Vol. 15, no. 2. – P. 528–529. – Mode of access: https://doi.org/10.1109/tnn.2004.826381 (date of access: 15.12.2025). – Title from screen.

6. Shubin I. Reuse of information based on the interpretation of knowledge [Electronic resource] / Ihor Shubin, Oleksandr Karataiev // Innovative technologies and scientific solutions for industries. – 2023. – No. 2 (24). – P. 62–71. – Mode of access: https://doi.org/10.30837/itssi.2023.24.062 (date of access: 15.12.2025). – Title from screen.

7. Shubin I. Categorical analysis of logical networks in application to intelligent radar systems [Electronic resource] / Igor Shubin, Stanislav Snisar, Svitlana Litvin // 2020 IEEE international conference on problems of infocommunications. science and technology (PIC S&T), Kharkiv, Ukraine, 6–9 October 2020. – [S. l.], 2020. – Mode of access: https://doi.org/10.1109/picst51311.2020.9467893 (date of access: 15.12.2025). – Title from screen.

8. Karataiev O. A method for investigating links between discrete data features in knowledge bases in the form of predicate equations [Electronic resource] / Oleksandr Karataiev, Dmytro Sitnikov, Nataliia Sharonova // Computational Linguistics and Intelligent Systems (COLINS 2023). Volume I: Machine Learning Workshop : Proceedings of the 7th International Conference, Kharkiv, 20–21 April 2023. – [S. l.], 2023. – P. 224–235. – Mode of access: https://ceur-ws.org/Vol-3387/paper17.pdf (date of access: 15.12.2025). – Title from screen.