N-TUPLE ALGEBRA-BASED PROBABILISTIC LOGIC

B. A. Kulik

Institute of Problems in Machine Science, Russian Academy of Sciences,
Vasil'evskii Ostrov, Bol'shoi pr. 61, St. Petersburg, 199178 Russia

Abstract

The concept of "probabilistic logic" known in artificial intelligence needs a more thorough substantiation. A new approach to constructing probabilistic logic based on the N-tuple algebra developed by the author is proposed. A brief introduction is given to the N-tuple algebra and its properties that provide efficient paralleling of algorithms for solving problems of logical analysis of systems in computer implementation are generalized. Methods for solving direct and inverse problems of probabilistic simulation of logical systems are considered.

INTRODUCTION

The *N*-tuple algebra based on the known properties of Cartesian products [1] was developed for solving certain problems in artificial intelligence, in particular, for simulating logical systems and in order to reduce the complexity of algorithms of logical inference [2, 3]. The foundations of the N-tuple algebra and potentialities of its application in probabilistic simulation were presented in [4–6]. Further investigations of this system have shown that the class of problems solved on its ground can be extended substantially. In addition, the structures of the N-tuple algebra can be programmed relatively easily and have a natural parallelism; therefore, their application in software–hardware support of logical and logical–probabilistic analysis of systems allows one to reduce the cost of development of programs and the required computational resources.

In this paper, we give a brief introduction to N-tuple algebra taking into account the correction of certain terminology introduced earlier and its capabilities in solving the inverse problem of logical–probabilistic analysis are also considered, i.e., the restoration of probability distributions of simple events based on data on the probabilities of complex events. Such problems were posed within the framework of probabilistic logic [7–9].

1. BASIC CONCEPTS AND STRUCTURES OF THE N-TUPLE ALGEBRA

The *N*-tuple algebra contains a number of definitions and more than 30 theorems, which are used in order to obtain the following results:

(1) It is substantiated that it is isomorphic to the system such as the theories of multiplace relations, of propositional calculus, and of predicate calculus.

(2) This system is embedded in the probability space.

(3) The algorithmic foundation for solving various problem of logical analysis of systems (logical inference, search for correct hypotheses and "hidden axioms", probabilistic analysis, etc.) is developed.

In order to avoid the consideration of original works [2–6], we present here in short the basic concepts and relationships of N-tuple algebra necessary for understanding probabilistic relations. Moreover, in the author's opinion, this section is useful because of the appropriate correction in the previous notation.

The *N*-tuple algebra is based on the concept of a flexible universe of discourse. Let a totality of different sets called *sorts* be given. We assign a certain set of attributes to every sort (in the previous publications, the term "coordinate", which does not seem to be completely adequate, was used). The *domain* of each attribute is the set that is equal to the corresponding sort. In mathematical logic, the domains of definition of variables correspond to attribute domains. A *flexible universe* consists of a totality of *partial universes*, Cartesian products of domains for a given sequence of attributes. The sequence of attributes that determines a given partial universe is called a *relation diagram*.

The N-tuple algebra contains five structures (N-tuple algebra objects) such as an elementary *n*-tuple, *C*-*n*-tuple, *C*-system, *D*-*n*-tuple, and *D*-system. Objects of the N-tuple algebra formed in the same partial universe are called similar.

Suppose that a partial universe in the form of a Cartesian product of arbitrary sets is given $S = X_1 \times X_2 \times \ldots \times X_n$. Clearly, *S* can be represented as a space of features with *attributes X*. The domains of these attributes correspond to feature *scales*. Then, we can form in the space *S* the following substructures:

(1) Projections, which are subspaces in which only certain attributes from the set of attributes generating *S* are used.

(2) The Cartesian products in the given relation diagram; certain subsets of the sets *X* represented in the given relation diagram are components of these Cartesian products.

Consider examples of these substructures. Let $S = X \times Y \times Z$, where $X = \{a, b, c, d\}$, $Y = \{f, g, h\}$, and $Z = \{a, b, c\}$. The Cartesian products $X \times Y$, $X \times Z$, etc. or particular sets, e.g. *X*, may be projections of this space. For simplicity, we assume that in this system a unique attribute corresponds to each sort.

Within the limits of the space *S* or some its projection, we can give the corresponding substructures in the form of Cartesian products. For example, the Cartesian product

$R[XYZ] = \{b, d\} \times \{f, h\} \times \{a, b\}$

is an example of such a substructure. Here, the expression [*XYZ*] is a relation diagram. It can be easily tested that $R \subseteq S$ (a property of Cartesian products). Similarly, a certain subset of elementary *n*-tuples of the projection $Y \times Z$ can be represented as the Cartesian product $Q[YZ] = \{f, g\} \times \{a, c\}$. Cartesian products represent the sets of elementary *n*-tuples. If necessary, these sets can be listed, although it is not necessary in performing operations with the structures of N-tuple algebra.

An *elementary n-tuple* is an element of a Cartesian product or its projection; i.e., a sequence of elements, each of which belongs to the domain of the corresponding attribute. For example, the Cartesian product $Q[YZ] = \{f, g\} \times \{a, c\}$ contains the following set of elementary *n*-tuples: $\{(f, a), (f, c), (g, a), (g, c)\}$.

A *C*-*n*-*tuple* is an *n*-tuple given in the complete space or in some its projection with components generated by subsets of the corresponding domains of attributes. A *C*-*n*-tuple is interpreted as the Cartesian product of these components; i.e., as a certain subset of elementary *n*-tuples. Square brackets are employed for denoting *C*-*n*-tuples. For example, the relations *R* and *Q* presented above can be represented as *C*-*n*-tuples

$R[XYZ] = [\{b, d\}\ \{f, h\}\ \{a, b\}]$; $Q[YZ] = [\{f, g\}\ \{a, c\}]$.

A C-n-tuple that has at least one empty component is empty. In the N-tuple algebra, if we deal with models of propositional or predicate calculus, this proposition is taken as an axiom, which has an interpretation based on the properties of Cartesian products.

To generalize operations that are applied frequently to structures with various relation diagrams, we introduce *dummy components*. These components have two types. One of these components is used in *C*-*n*-tuples and is designated "∗". Another dummy component (∅) involved in *D*-*n*-tuples is considered in what follows.

Dummy components "∗" designate the sets equal to the domains of the corresponding attributes; they can be inserted in the corresponding *C*-*n*-tuple instead of missing attributes and thus introduce new attributes in it. For example, the *C*-*n*-tuple $Q[YZ] = [\{f, g\}\ \{a, c\}]$ can be represented in the relation diagram [*XYZ*] in the form of the *C*-*n*-tuple $[*\ \{f, g\}\ \{a, c\}]$ using a dummy component. Since the dummy component in *Q* corresponds to the attribute *X*, we have the equality

$[*\ \{f, g\}\ \{a, c\}] = [\{a, b, c, d\}\ \{f, g\}\ \{a, c\}]$.

The *intersection* of similar *C*-*n*-tuples is performed componentwise. The result of intersection is the *C*-*n*-tuple that contains the intersection of the components of the source *C*-*n*-tuples related to the same attribute, e.g.,

$[\{b, d\}\ \{f, h\}\ \{a, b\}] \cap [*\ \{f, g\}\ \{a, c\}] = [\{b, d\}\ \{f\}\ \{a\}]$.

The result of intersection of *C*-*n*-tuples may be an empty set (empty *C*-*n*-tuple)

$[\{b, d\}\ \{f, h\}\ \{a, b\}] \cap [*\ \{g\}\ \{a, c\}] = \varnothing$,

since the intersection of the second components of these *C*-*n*-tuples is an empty set.

Many relations given as subsets of a Cartesian product cannot always be represented by a single *C*-*n*-tuple. Therefore, it is reasonable to introduce a universal structure that is the union of similar *C*-*n*-tuples.

A *C*-*system* is a structure that is the union of an arbitrary number of similar *C*-*n*-tuples. As *C*-*n*-tuples, *C*-systems are confined by square brackets. For example, for the space *S* given above, we can define a certain relation *P* as a *C*-system

$$R[XYZ] = \begin{bmatrix} \{a,d\} & * & \{b,c\} \\ \{b,d\} & \{f,h\} & \{a,c\} \\ \{b,c\} & \{g\} & \{b\} \end{bmatrix}.$$

The fact that a *C*-*n*-tuple (C_m) is included in another *C*-*n*-tuple (C_n) is tested componentwise, $C_m \subseteq C_n$ if and only if all components C_m are included in the corresponding components of C_n. Based on the properties of Cartesian products, we are able to find conditions for which the union of two *C*-*n*-tuples C_m and C_n can be transformed into a single *C*-*n*-tuple. There are two such conditions

(1) if $C_m \subseteq C_n$, then $C_m \cup C_n = C_n$;

(2) if C_m and C_n differ only in the *i*th component, then $C_m \cup C_n$ can be represented as a single *C*-*n*-tuple that have all component the same except for the *i*th component, which becomes equal to the union of the corresponding components from C_m and C_n.

If we know how to obtain the intersection of *C*-*n*-tuples, we can formulate the algorithm for finding the intersection of a *C*-*n*-tuple with a *C*-system and a *C*-system with a *C*-system. For this purpose, we should represent *C*-*n*-tuples as conventional sets whose elements are similar elementary *n*-tuples. Then, the *C*-system that contains the *C*-*n*-tuples *A*, *B*, …, *L* is the union of these sets. On this ground, using the law of algebra of sets, in particular, the distributive law, we can easily obtain the corresponding algorithms for calculating the intersection of the corresponding structures.

Algorithm 1. The calculation of the intersection of a *C*-*n*-tuple *P* with a *C*-system *Q*:

(1) calculate the intersection of the *C*-*n*-tuple *P* with each *C*-*n*-tuple from *Q*;

(2) eliminate empty *C*-*n*-tuples from the obtained results;

(3) form a *C*-system from the remaining *n*-tuples;

(4) terminate the algorithm.

Algorithm 2. Calculation of the intersection of a *C*-system *P* with a *C*-system *Q*:

(1) calculate the intersection of the *C*-*n*-tuple from *P* with each *C*-*n*-tuple from *Q*;

(2) eliminate empty *C*-*n*-tuples from the obtained results;

(3) form a *C*-system from the remaining *n*-tuples;

(4) terminate the algorithm.

As an example, we calculate the intersection of two *C*-systems given on the space *S* defined above (this means that the symbol "∗" in the second position of *C*-*n*-tuples corresponds to the set $X_2 = \{f, g, h\}$)

$$P[XYZ] = \begin{bmatrix} \{a,b,d\} & \{f,h\} & \{b\} \\ \{b,c\} & * & \{a,c\} \end{bmatrix},\ Q[XYZ] = \begin{bmatrix} \{a,d\} & * & \{b,c\} \\ \{b,d\} & \{f,h\} & \{a,c\} \\ \{b,c\} & \{g\} & \{b\} \end{bmatrix}.$$

(1) We calculate the intersection of all pairs of *C*-*n*-tuples belonging to different *C*-systems

$[\{a, b, d\}\ \{f, h\}\ \{b\}] \cap [\{a, d\} * \{b, c\}] = [\{a, d\}\ \{f, h\}\ \{b\}]$;
$[\{a, b, d\}\ \{f, h\}\ \{b\}] \cap [\{b, d\}\ \{f, h\}\ \{a, c\}] = \varnothing$;
$[\{a, b, d\}\ \{f, h\}\ \{b\}] \cap [\{b, c\}\ \{g\}\ \{b\}] = \varnothing$;
$[\{b, c\} * \{a, c\}] \cap [\{a, d\} * \{b, c\}] = \varnothing$;
$[\{b, c\} * \{a, c\}] \cap [\{b, d\}\ \{f, h\}\ \{a, c\}] = [\{b\}\ \{f, h\}\ \{a, c\}]$;
$[\{b, c\} * \{a, c\}] \cap [\{b, c\}\ \{g\}\ \{b\}] = \varnothing$.

(2) From the remaining nonempty *C*-*n*-tuples, we form the *C*-system

$$P \cap Q = \begin{bmatrix} \{a,d\} & \{f,h\} & \{b\} \\ \{b\} & \{f,h\} & \{a,c\} \end{bmatrix}.$$

Even this relatively simple example shows certain opportunities to reduce the complexity of the algorithms using the *N*-tuple algebra. The same result can be obtained if we convert preliminarily the source *C*-systems to the set of elementary *n*-tuples. However, this increases the complexity of computations since the *C*-system *P*, *C*-system *Q*, and the *C*-system $P \cap Q$ contain 24, 20, and 8 elementary *n*-tuples, respectively.

The union of *C*-*n*-tuples and *C*-systems is computed much simpler. For this purpose, we need to form a new *C*-system from the united structures that contains all *C*-*n*-tuples of these structures. Then, we may unite certain *C*-*n*-tuples in particular cases. It is necessary to remember that the implemented algorithms of union and intersection, as well as testing the inclusion of the structures of the N-tuple algebra, make sense only when these structures are similar or are transformed into similar structures with the help of addition of dummy attributes.

If it is required to compute the *complement* of a *C*-*n*-tuple, then, using conventional methods from the theory of multiplace functions, we should perform the following operations:

(1) to split into elementary n-tuples the *C*-*n*-tuple *R* and the partial universe *S* corresponding to it;

(2) eliminate elementwise all elementary *n*-tuples belonging to *R*.

It is clear that this operation is laborious in general. However, it is simplified essentially if we employ the following relations. Let us first define the notion of complement to the component of a *C*-*n*-tuple. If a multiplace relation is defined in the space such that each its attribute is represented by a certain set, then it is obvious that the universe for the component of the *C*-*n*-tuple is the domain of the attribute that corresponds to it (partial universe), and the set that contains all elements of this partial universe that do not belong to this component is the complement of the component. For example, assume that, in the space $S = X \times Y \times Z$, a *C*-*n*-tuple $R = [R_1\ R_2\ R_3]$ is given. Then, correspondingly, we have $\overline{R_1} = X \backslash R_1$; $\overline{R_2} = Y \backslash R_2$; and $\overline{R_3} = Z / R_3$.

The following theorem can be proved based on the properties of the Cartesian product [1].

Theorem 1. The complement of the *C*-*n*-tuple $T = [R_1\ R_2\ \ldots\ R_n]$ is the C-system

$C = \begin{bmatrix} \overline{R_1} & * & \dots & * \\ * & \overline{R_2} & \dots & * \\ \dots & \dots & \dots & \dots \\ * & * & \dots & \overline{R_n} \end{bmatrix}$ of dimension $n \times n$, in which each diagonal component is the complement of the corresponding component of C-n-tuple T, and the other components are dummy.

Consider an example. Assume that in the space $S = X \times Y \times Z$ mentioned above a C-n-tuple $T = [\{b, d\}\ \{f, h\}\ \{a, b\}]$ is given. By Theorem 1, its complement is the C-system

$$\overline{T} = \begin{bmatrix} X \setminus \{b,d\} & * & * \\ * & Y \setminus \{f,h\} & * \\ * & * & Z \setminus \{a,b\} \end{bmatrix} = \begin{bmatrix} \{a,c\} & * & * \\ * & \{g\} & * \\ * & * & \{c\} \end{bmatrix}.$$

By Theorem 1, the C-systems that represent the complement of a C-n-tuple can be represented as a single n-tuple of sets using for designation inverted square brackets. Then, we obtain the equality

$$\overline{T} = \begin{bmatrix} \{a,c\} & * & * \\ * & \{g\} & * \\ * & * & \{c\} \end{bmatrix} =]\{a, c\}\ \{g\}\ \{c\}[.$$

This brief representation of the diagonal C-system generates a new structure of the N-tuple algebra, which is called a *D-n-tuple*. It turns out that this structure not only allows one to represent briefly diagonal C-systems, but is also used independently in certain operations and retrieval requests. The terms "C-n-tuple" and "D-n-tuple" are not chosen randomly. In the simplest case, C-n-tuple and D-n-tuple correspond to conjunction and disjunction of one-place predicates with different variables. Using D-n-tuples, we can formulate one more (the fifth) structure of the N-tuple algebra, a D-system.

A D-system is a structure similar to a matrix whose rows contain similar D-n-tuples, which is interpreted as the intersection of the sets of elementary n-tuples belonging to these D-n-tuples.

The representation of a D-system is similar to the representation of a C-system, but, instead of square brackets, we use inverted ones. For example, the complement to the C-system

$$F[XYZ] = \begin{bmatrix} \{a,b,d\} & \{f,h\} & \{b\} \\ \{b,c\} & * & \{a,c\} \end{bmatrix},$$

given in the space S can be represented as the D-system

$$\overline{F} = \begin{bmatrix} X \setminus \{a,b,d\} & Y \setminus \{f,h\} & Z \setminus \{b\} \\ X \setminus \{b,c\} & Y \setminus * & Z \setminus \{a,c\} \end{bmatrix} = \begin{bmatrix} \{c\} & \{g\} & \{a,c\} \\ \{a,d\} & \varnothing & \{b\} \end{bmatrix}.$$

Thus, the D-system is *dual* (according to the de Morgan laws) to the C-system and is the *intersection* of two C-n-tuples. The algorithms for converting D-n-tuples and D-systems into C-systems equal to them have been developed.

The complete analogy between the structures of the N-tuple algebra and formulas of predicate calculus can easily be established. Consider the main ideas. In the predicate calculus, *conjunction* of one-place predicate with different variables corresponds to the C-n-tuples in the trivial case (when particular attributes are not related to multiplace relations). For example, the C-n-tuple $P[XYZ] = [P_1\ P_2\ P_3]$, where $P_1 \subseteq X$; $P_2 \subseteq Y$; and $P_3 \subseteq Z$, corresponds to the logical formula $H = P_1(x) \wedge P_2(y)\ \wedge P_3(z)$. The negation of the formula H (*disjunction* of one-place predicates) $\neg H = \neg P_1(x) \vee \neg P_2(y)\ \vee \neg P_3(z)$ corresponds to the D-n-tuple $\overline{P} =]\ \overline{P_1}\ \ \overline{P_2}\ \ \overline{P_3}\ [$. An empty object of the N-tuple algebra corresponds to an *identically false formula*.

An object of the N-tuple algebra corresponds to a *satisfiable formula*. An elementary *n*-tuple that is involved in the composition of a nonempty object of the N-tuple algebra corresponds to a *satisfying substitution* of the logical formula.

If a dummy attribute is introduced in a *C*-*n*-tuple or in a *C*-system, then this procedure corresponds to the inference rule known in the predicate calculus which is called the *generalization rule*. For example, if the object of the N-tuple algebra $G[XZ] = \begin{bmatrix} \{a,c\} & * \\ \{a,c,d\} & \{b,c\} \end{bmatrix}$ corresponds to a formula $F(x, z)$ in the predicate calculus, then, appending to this object a dummy attribute Y, we obtain the object of the N-tuple algebra $G_1[XYZ] = \begin{bmatrix} \{a,c\} & * & * \\ \{a,c,d\} & * & \{b,c\} \end{bmatrix}$, which corresponds to the formula $\forall yF(x, z)$ obtained from $F(x, z)$ by the generalization formula.

Together with operations of the algebra of sets on objects of the N-tuple algebra, the following three additional operations on attributes are introduced:

(1) the transposition of attributes and the their corresponding columns of the matrix of the object of the N-tuple algebra;

(2) addition of a new dummy attribute; and

(3) attribute elimination.

We also use two quantifier operations $\exists x(P)$ and $\forall x(P)$, which not only recognize identical falsity or satisfiability of the corresponding structure, but also, in the case of satisfiability, allows one to obtain the object of the N-tuple algebra corresponding to this expression. Certain combinations of operations with attributes and operations of the algebra of sets give the opportunity to perform negation, composition, and join of relations, and logical inference and operations with quantifiers. In detail, see this in [4–6].

S. RESOURSE SAVING IN COMPUTER IMPLEMENTATION OF STRUCTURES OF THE N-TUPLE ALGEBRA

Computational complexity of operations of the algebra of sets and testing inclusion depends on the class of structures the employed objects of the N-tuple algebra belong to. For example, an inclusion of a *C*-*n*-tuple into a *C*-system is tested in general with the help of an algorithm of exponential computational complexity, while the algorithm for testing of an inclusion of a *C*-*n*-tuple and even *C*-system into a *D*-system has a polynomial complexity. To fulfill certain operations and tests, it is required to transform an object of the N-tuple algebra into an object of the *alternative class* that is equivalent to it (for example, a *C*-system into *D*-system and vice versa), which is achieved for the *C*-system or the *D*-system by algorithms of exponential complexity. The operation of complement of an object of the N-tuple algebra in all cases is fulfilled by an algorithm of polynomial computational complexity, but, in this case, the system is transformed into the alternative class. The operations of intersection and union of objects of the N-tuple algebra that belong to the same class are fulfilled by algorithms of polynomial complexity, but if they belong to different classes, then, to fulfill these operations, it is necessary to transform one of them into another class.

In problems that are known in logic as problems of deductive inference, frequently, it is required to test that one object of the N-tuple algebra is included into another, as well as to fulfill quantifier operations. Table 1 presents various combinations of objects of the N-tuple algebra and the sign "+" label the combinations for which the execution algorithms of the corresponding operations are polynomial under the condition that all domains of attributes are simple sets (i.e., not multiplace relations). Note that, in all cases, to test whether a given elementary *n*-tuple belongs to any structure, we need an algorithm of polynomial complexity.

Table 1

Action	*C*-n-tuple	*C-system*	*D*-n-tuple	*D*-system
Testing of inclusion of a C-n-tuple into	+		+	+
Testing of inclusion of a C-system into	+		+	+
Testing of inclusion of a D-n-tuple into	+		+	+
Testing of inclusion of a D-system into				
Quantifier operation $\forall x$	+		+	+
Quantifier operation $\exists x$	+	+	+	

However, within the framework of the N-tuple algebra, methods for reducing complexity of computationally exponential algorithms, as well as methods for recognizing particular cases of structures were developed that allow one to fulfill the corresponding operations, transformations, and testing for polynomial time. Even in the cases when it is not possible to use an algorithm of polynomial computational complexity, the required computational resources can be reduced by using natural parallelism inherent in objects of the N-tuple algebra.

In contrast to conventional data structures applied in the computer implementation of logical and logical-probabilistic analysis, the structures of the N-tuple algebra are *matrixwise*, which, using the corresponding software–hardware implementation, makes it possible to reduce relatively easily the computational resources by paralleling the operations.

The computer implementation of objects of the N-tuple algebra employs parallelism at the levels of (1) of components; (2) rows; and (3) matrices. *At the level of components*, we can represent domains and their subsets in the form of a totality of logical vectors. To implement operations of the algebra of sets and of tests of inclusion, we can apply logical operations with integer vectors. *At the level of rows*, we are able to fulfill simultaneously operations or tests of inclusion with all pairs of components of *C*-*n*-tuples and *D*-*n*-tuples. *At the level of matrices*, we can fulfill simultaneously operations of the algebra of sets and the test of inclusion for a set of pairs elements of which are rows (*C*-*n*-tuples and *D*-*n*-tuples from different objects of N-tuple algebra. For example, in the computation of the intersection of two *C*-systems (see algorithm 2), all operations of intersection of *C*-*n*-tuples employed in this algorithm can be executed in parallel.

T. LOGIC AND PROBABILITY

The term "probabilistic logic" has been widely applicable in AI since the publication of work [7] by Nilsson. His idea was extended and developed by other researchers [8, 9]. In these and other publications on probabilistic logic, the following problem was posed: given estimates of the probabilities of a certain set of event represented by formulas of propositional calculus, it is necessary to find a probabilistic estimate of the event represented by a logical formula different from the initial ones. Another aspect of the combination of probability and logic, e.g., the aspect that was implemented in logical-probabilistic methods (LPM) [10], where the probability of formulas is calculated based on the probabilistic values of logical variables have not been considered in those works. Moreover, the analysis of papers [7–9] has shown that the combination of classical concepts of "probability" and "logic" results in certain nonclassical logics. However, in this paper, we consider the concept of probabilistic logic within the framework of the N-tuple algebra.

The combination of concepts of "logic" and "probability" is rather difficult. At the first glance, it is completely simple if we take as a ground the system of axioms proposed by A.N. Kolmogorov [11], in which the algebra of events embedded into a probabilistic measure corresponds to the algebra of sets. For example, for events represented by sets A and B, the probabilistic measure of their union can be calculated as

$$p(A \cup B) = p(A) + p(B) - p(A \cap B).$$

Thus, together with the probabilities $p(A)$ and p(B), to compute the probabilities of the event $A \cup B$, it is necessary to know the probability $p(A \cap B)$, which, within the limits of certain constraints, in particular $p(A \cap B) \leq \min(p(A), p(B))$, does not depend on $p(A)$ and $p(B)$. If in addition, we have $A \cap B = \varnothing$, then the events A and B are dependent. However, if we assume that A and B are different logical variables rather than sets, then the probability of disjunction of these events can be calculated by the formula

$$p(A \vee B) = p(A) + p(B) - p(A)pB).$$

To calculate this formula, it is sufficient to set only the probability of the events A and B.

The question is why, for logical relations, another methodology takes place for computing probability, although it seems reasonable that the algebra of sets and the Boolean algebra are isomorphic. The answer to this question is a key point in the combination of the concepts of "logic" and "probability". It is the matter of fact that, in classical logic, elementary events corresponding to different logical variables are inconsistent; therefore, any logical formula of n free variables is isomorphic to a certain n-place relation, and the events that correspond to the discriminate variables belong to different attributes. In other words, logical variables may be dependent but not initially and only by the fact that they are involved into a certain logical formula, which determines the dependence between them.

Absurdity (from the point of view of mathematical logic) of another approach can be seen from the following example, which is presented sometimes in papers on probabilistic logic: for logical variables (but not for formulas!) X and Y, the probabilities $p(X)$, $p(Y)$, and $p(X \wedge Y)$ are given, and the last probability is not necessarily equal to the product of the preceding ones.

This clearly implies that *in the embedding of logical systems into the probabilistic space, it is necessary to take into account that we deal with the system that is isomorphic to the algebra of sets according to the Kolmogorov system of axioms, but, in structure, the sets themselves are sets of n-tuples involved in multiplace relations.*

It is this circumstance, which is taken into account in the N-tuple algebra explicitly or implicitly, is not taken into account in different versions of "probabilistic logic." The assumptions that events that correspond to different logical variables can be dependent in themselves, i.e., without taking into account the logical formula that relates them, means that the laws of mathematical logic are violated. It is not the same when we deal with the formulas, in which the dependence between different variables is established or with different logical formulas, which can be dependent only under the assumption that they contain at least one free variable that is common for them.

4. N-TUPLE ALGEBRA-BASED PROBABILISTIC ANALYSIS OF SYSTEMS

Consider methods of probabilistic simulation that use the N-tuple algebra in greater detail. The basic cross-linking concept of the N-tuple algebra is the concept of C-n-tuple. If we know the probabilistic measures of components of the C-n-tuple, then the measure of the C-n-tuple can be calculated as the product of the measures of its components. For example, when the C-n-tuple $R = [A\ B\ C]$ is given in measurable attributes and the measures of its components are equal to $\mu(A)$, $\mu(B)$, and $\mu(C)$, respectively, then we have

$\mu I = \mu(A) \bullet \mu(B) \bullet \mu(C)$.

If we deal with the embedding of logical formulas in the probabilistic space, then all attributes of the space in which the totality of objects of the N-tuple algebra is given have measure one, and all objects of the N-tuple algebra have measures that do not exceed one. This corresponds to the probabilistic measure not only in numerical relations, but also by the fact that the system of events simulated by the N-tuple algebra is isomorphic to the algebra of sets.

To compute the measures of objects from the N-tuple algebra that are different from *C*-*n*-tuples, it is necessary to *orthogonalize* them, i.e., to transform into an equivalent *C*-system in which the intersection of any pair of *C*-*n*-tuples is an empty set. The methods of orthogonalization of arbitrary objects of the N-tuple algebra have been developed; the results in detail can be found in [2–6]. Note that the measure of an orthogonal *C*-system is equal exactly to the sum of the measures of *C*-*n*-tuples that belong to it. In addition, the following regularity has been established: *the orthogonalization not only allows one to prepare an object of the N-tuple algebra for calculating its probability, but also, in many cases, reduces the computational cost substantially in solving other problems* (*e.g., in solving the satisfiability problem*).

If an object of the N-tuple algebra is a representation of formulas of propositional calculus, then it is given in the universe $\{0, 1\}^n$, where n is the number of logical variables of the formula. Each column of a *C*-*n*-tuple or a *C*-system is related to a certain logical variable. The variable x_k corresponds to the kth column, the state 1 in the object of the N-tuple algebra corresponds to the literal x_k, and the state 0 corresponds to the literal $\bar{x}_k$. Any row (*C*-*n*-tuple) in a *C*-system corresponds to the conjunction of the formula expressed as the disjunctive normal form (DNF). If some clause misses the variables that are involved in the composition of formula, then, instead of them, the corresponding dummy variable "$*$" is inserted in the *C*-*n*-tuple.

Example 1. Assume that the formula of propositional calculus

$$F_Q = (x_1 \wedge \neg x_3) \vee (\neg x_2 \wedge x_3) \vee (\neg x_1 \wedge x_2). \qquad (4.1)$$

is given. Since there are three logical variables here, this formula can be represented as an object of the N-tuple algebra Q in the universe $\{0, 1\}^3$

$$Q = \begin{bmatrix} \{1\} & * & \{0\} \\ * & \{0\} & \{1\} \\ \{0\} & \{1\} & * \end{bmatrix}.$$

This formula and the object of the N-tuple algebra that corresponds to it are orthogonal; therefore Q can be expressed directly in terms of the probabilistic measure. Assume that, in the object Q of the N-tuple algebra, the probabilities of events are given as follows: pi is the probability of the event 1 in the ith attribute, and $1–p_i$ is the probability of the event 0 in the ith attribute. Taking into account that the measure of a *C*-*n*-tuple is equal to the product of the measures of its components, and the measure of an orthogonal *C*-system is the sum of the measures of *C*-*n*-tuples belonging to it, we obtain the formula

$$p(Q) = p_1(1–p_3)+(1–p_2)p_3+(1–p_1)p_2. \qquad (4.2)$$

In the LPM, formula (4.2) is called the probabilistic function (PF) of formula (4.1). This function can also be derived from the orthogonal *C*-system using the change of elements of components by the probabilities corresponding to them and by the transformation of the system into a polynomial. At the first glance, it seems that the structures of the N-tuple algebra provide only a different way for expressing logical formulas. However, when the models get complicated (in particular, in the transition to many-state systems), using the N-tuple algebra, it turns out to be possible to simplify essentially the algorithms for solving a number of problems considered in LPM. In addition, in the embedding into the probabilistic space, the concept of "regression equation" is introduced in the N-tuple algebra, which allows one to pose and solve the problem of probabilistic logic in accordance with the Nilsson statement. *If in the probabilistic functions of type* (4.2)*, we suppose that p_i are variables rather than fixed numbers, then these formulas are an exact*

regression equation of the corresponding logical formula. The proof of this proposition can be found in [5]. Consider an example of many-state system.

Example 2. Let $R = \begin{bmatrix} \{a_1, a_2\} & \{b_1, b_3\} \\ \{a_3\} & \{b_1, b_2\} \end{bmatrix}$ be an orthogonal *C*-system with three states given in the space $\{a_1, a_2, a_3\} \times \{b_1, b_2, b_3\}$ with probabilities $p(a_i)$ and $p(b_i)$, and $p(a_3) = 1 - p(a_1) - p(a_2)$ and $p(b_3) = 1 - p(b_1) - p(b_2)$. Then, the probability of the event expressed by the object *R* of the *N*-tuple algebra if the required probabilities are substituted is

$$p\mathrm{I} = (p(a_1) + p(a_2))(1 - p(b_2)) + (1 - p(a_1) - p(a_2))(p(b_1) + p(b_2)).$$

The presented approach corresponds to the direct problem of logical–probabilistic analysis when, for given probabilities of elementary events, the probability of a complex event is calculated. In the *inverse problem*, the statement is different. The problem is, based on the data on the probabilities of certain complex events, we should calculate the probabilities of elementary events. After this, we can calculate the probabilities of other complex events. Problems solved in probabilistic logic are of this type. Consider the example presented in the paper by Nilsson [7].

Example 3. Given a totality of events specified by formulas *A* and $A \supset B$ of propositional calculus, and $p(A) = p_1$ and $p(A \supset B) = p_2$. It is necessary to estimate the probability $p(B)$ of the event *B*.

We solve this problem by the methods of the N-tuple algebra. There are only two logical variables *A* and *B* that are also elementary events in this case. Assume that the probabilities of these events are $p(A)$ and $p(B)$, respectively. The conditions of the problem imply that $p(A) = p_1$. Let us express the given formulas in the structures of the N-tuple algebra using the universe $\{0, 1\}^2$:

$$A = [\{1\}\ *];\ B = [*\ \{1\}];\ \ A \supset B = \overline{A} \vee B =]\{0\}\ \{1\}[= \begin{bmatrix} \{0\} & * \\ \{1\} & \{1\} \end{bmatrix}$$

(here the *D*-*n*-tuple corresponding to formula $A \supset B$ is transformed into an orthogonal *C*-system).

On this ground, we write the probabilistic formulas for the events *A* and $A \supset B$

$$P(A) = p_1; \qquad P(A \supset B) = (1 - p(A)) + p(A)p(B) = p_2.$$

We obtain the system of two equations

$$p(A) = p_1;$$
$$(1 - p(A)) + p(A)\,p(B) = p_2.$$

This easily implies that

$$p(B) = \frac{p_1 + p_2 - 1}{p_1}.$$

In this problem, we obtain an exact solution, while in [7] the solution is obtained as the inequality $p_2 + p_1 - 1 \le p(B) \le p_2$.

In general, the algorithm for solving problems of probabilistic logic is as follows. Assume that we have initial logical formulas F_i with given probabilities $p(F_i)$ and a formula G whose probability $p(G)$ has to be calculated. Then, it is necessary to fulfill the following sequence of operations:

(1) formulas F_i and *G* are transformed into orthogonal *C*-systems;

(2) for each of these systems, the regression equations $E(F_i)$ and $E(G)$ are derived;

(3) the system of equations $\{ E(F_i)\}$ is formed and solved;

(4) if the system of equations $\{ E(F_i)\}$ has a unique solution, then the obtained values of variables are substituted into the formula $E(G)$ and an exact solution is found.

In the Nilsson problem, an exact solution was obtained by the methods of the N-tuple algebra. However, this situation is not possible in all cases. Consider an example.

Example 4. Given probabilities of the events described by the logical formulas

$$p(A \vee B) = a; \quad p(A \wedge B) = b.$$

Find estimates of $p(A)$ and $p(B)$. Let us express the given events in the system as orthogonal C-systems

$$A \vee B \Leftrightarrow]\{1\} \quad \{1\}[= \begin{bmatrix} \{1\} & * \\ \{0\} & \{1\} \end{bmatrix};$$

$$A \wedge B \Leftrightarrow [\{1\} \quad \{1\}].$$

We derive the system of equations

$p(A) + (1 - p(A))\, p(B) = a$;

$p(A)\, p(B) = b$.

Solving this system, we obtain

$$p(A) = \frac{a + b \pm \sqrt{(a+b)^2 - 4b}}{2}; \quad p(B) = \frac{a + b \mp \sqrt{(a+b)^2 - 4b}}{2}.$$

It is clear that the obtained solutions do not give a unique solution in the cases when the radicand is not equal to 0 (this is possible for $p(A) \neq p(B)$). If we take into account that both initial formulas are symmetric, then this uncertainty was caused by the conditions of the problem.

For the presented examples, we can test the calculation numerically, if the probabilistic models corresponding to them are constructed. For example, for example 4, the probabilistic model is as follows: assume that two coins are tossed, note that the probability that at least once heads have occurred (the formula A .B corresponds to this event), and the probability that heads have occurred in two tosses (formula $A \vee B$) are known. If we know the probability of the event that heads have occurred (for correct coins, it is equal to 0.5), by the laws of probability theory, we can calculate the probability of these complex events $p(A \vee B) = 0.75$ and $p(A \wedge B) = 0.25$. The substitution of these values into the formulas for $p(A)$ and $p(B)$ presented above gives a correct answer. A similar test can be performed for incorrect coins, when the probability that heads occur differs from 0.5.

The probabilistic relations obtained based on the ntuple algebra allow one not only to estimate the probability of complex events for given distribution functions of events in each attribute, but also to solve the inverse problem of probabilistic analysis, i.e., to estimate the types and parameters of marginal distributions (distributions in attributes and certain projections).

First, we consider the following statement of the problem: a system is represented either in structures of the N-tuple algebra or in the form of a system of logical functions, and, for any variable, the probability distribution is known. In a multidimensional space, a distribution in either attributes or in certain projections of this space is called marginal distribution. It is necessary to calculate the probability distribution of the system and to estimate the stability of this distribution. The problems of this type arise in evaluating the reliability and safety of systems with complex structure and logical–probabilistic risk management in business and industry [12]. Using the relations derived above, logical systems in which these problems are solved can be conversed into measurable systems in the N-tuple algebra.

Assume that every attribute of a certain set of objects of the N-tuple algebra is represented by a finite system of events. The following two variants of specifying a system of events in the attribute X_i:

(1) in the form of an elementary system (i.e., a system of pairwise inconsistent events);

(2) on a continuous probability distribution $p(x_i)$ in the form of a finite set of intervals (a_i, b_i) nonoverlapping in general, where a_i and b_i are the values of the parameter x_i and $a_i < b_i$.

For the first variant, it is sufficient to assign to each event its probability. The second variant can be reduced to the first one by the following procedure:

(1) in the system $\{(a_i, b_i)$ of intervals of the parameter x_i, the system of events is split into the set of pairwise nonoverlapping intervals–quanta;

(2) the initial system of events is transformed into a discrete one assigning to each initial event a certain set of quanta such that their union is equal to this event;

(3) for each quantifier e_r, compute the value of the probability p_r using as the lower and upper integration limits for $p(x_i)$ the endpoints of this quantum equal to the values of the parameter x_i.

If this procedure has been performed for each attribute of the object of the N-tuple algebra, then its probability is computed in the following order:

(1) the object of the N-tuple algebra is orthogonalized;

(2) the object of the N-tuple algebra is transformed into a polynomial in which we assign to each quantum the corresponding value of the probability.

Example 5. The system is given in the space $X \times Y$, where the attributes are represented in the form of intervals, and X= [0, 7] and Y= [0, 5]. The densities of probability distributions $f_1(x, d_1, e_1)$ and $f_2(y, d_2, e_2)$ on attributes are also known, where d_1, e_1, d_2, and e_2 are the parameters of the distributions. In this system a certain event is given in the form of an object of the N-tuple algebra

$$R[XY] = \begin{bmatrix} \{a_1\} & \{b_1\} \\ \{a_2, a_4\} & \{b_2\} \\ \{a_3\} & \{b_3\} \end{bmatrix},$$

where a_i and b_j are intervals given in Table 2. It is necessary to determine the order of calculations for computing the probability of the event R.

It is sufficient to use only open intervals in order to solve this problem. To simplify the system, we construct increasing series of endpoints of intervals in the attributes, for X, 0; 1.7; 2.8; 3.4; 4.3; 5.5; 6.4; and 7; for Y, 0; 1.4; 2.3; 3.2; and 5. Then, we obtain the following sets of elementary intervals for the attributes X (Table 3) and Y (Table 4).

Table 2

a_1	a_2	a_3	a_4	b_1	b_2	b_3
[0, 2.8]	[1.7, 3.4]	[3.4, 5.5]	[4.3, 6.4]	[0, 2.3]	[1.4, 3.2]	[2.3, 5.0]

Table 3

r_1	r_2	r_3	r_4	r_5	r_6	r_7
(0, 1.7)	(1.7, 2.8)	(2.8, 3.4)	(3.4, 4.3)	(4.3, 5.5)	(5.5, 6.4)	(6.4, 7)

Table 4

q_1	q_2	q_3	q_4
(0, 1.4)	(1.4, 2.3)	(2.3, 3.2)	(3.2, 5)

When we replace the intervals with the corresponding sets of quanta, we obtain

$a_1=\{r_1, r_2\}$; $a_2=\{r_2, r_3\}$; $a_3=\{r_4, r_5\}$; $a_4=\{r_5, r_6\}$;

$b_1=\{q_1, q_2\}$; $b_2=\{q_2, q_3\}$; $b_3=\{q_3, q_4\}$.

After the substitution into the initial C-system, we obtain

$$R = \begin{bmatrix} \{r_1, r_2\} & \{q_1, q_2\} \\ \{r_2, r_3, r_5, r_6\} & \{q_2, q_3\} \\ \{r_4, r_5\} & \{q_3, q_4\} \end{bmatrix}.$$

For each quantum r_i or q_j, we calculate the corresponding probability. For example,

$$p(r_3) = \int_{2,8}^{3,4} f_1(x, d_1, e_1)dx.$$

Now, we can orthogonalize the corresponding complex events. For the event R, we compute $\overline{R}$, and after the transformation of $\overline{R}$ into an orthogonal C-system, we find $p\mathrm{I} = 1 - p(\overline{R})$. Then, we obtain (the intermediate calculations are eliminated)

$$\overline{R} = \begin{bmatrix} \{r_3, r_4, r_5, r_6, r_7\} & \{q_1\} \\ \{r_4, r_7\} & \{q_2\} \\ \{r_1, r_7\} & \{q_3\} \\ \{r_1, r_2, r_3, r_6, r_7\} & \{q_4\} \end{bmatrix}.$$

Next, substituting the probabilities of quanta and using the theorems of the N-tuple algebra, we arrive at the expression

$$p\mathrm{I} = 1 - p(\overline{R}) = 1 - ((p(r_3)+p(r_4)+p(r_5)+p(r_6)+p(r_7))p(q_1)+(p(r_4)+p(r_7))p(q_2)+$$
$$+(p(r_1)+p(r_7))p(q_3) +(p(r_1)+p(r_2)+p(r_3)+p(r_6)+p(r_7))p(q_4)).$$

When solving the inverse problem for systems with many states, it is not always possible to solve the system of equations exactly since the number of variables in the regression equations is comparable with the number of all quanta and may exceed the number of equations. For example, in example 5, the number of quanta in the attribute *X* is seven; therefore, the number of unknown parameters of just this attribute is smaller by one, i.e., six. However, the problem can be solved approximately, if it is represented as an approximation problem. Assume that the attribute X is split into k_i quanta ($k_i > 2$). Then, we take as unknowns the types and parameters of continuous distributions for each attribute, rather than the magnitudes of quanta. Usually, the number of parameters of distributions does not exceed two — they will be unknown quantities. To estimate them, we can use optimization methods, in which the control actions are the types and parameters of marginal distributions, and the goal function is a generalized parameter, e.g., the mean value of the absolute deviations of the calculated values of the probabilities of the investigated complex events from the actual ones.

CONCLUSIONS

The application of the N-tuple algebra allows one to solve the direct and inverse problems of probabilistic analysis of logical systems in a multidimensional space not restricting ourselves to a particular class of distributions. As marginal distributions in solving the direct and inverse problems, we can use not only a normal distribution, but also any other distribution.

REFERENCES

1. N. Bourbaki, *Theorie des Ensembles* (Hermann, Paris, 1956–1958; Mir, Moscow, 1965).

2. .B. A. Kulik, “A System for Logical Design Based on the n-tuple Algebra,” Izv. Ross. Akad. Nauk, Tekh. Kibern., No. 3, 226–239 (1993).

3. B. A. Kulik, “New Classes of CNF with Polynomially Recognized Satisfiability Property,” Avtom. Telemekh., No. 2, 111–124 (1995).

4. B. A. Kulik, “Representation of Logical Systems in the Probabilistic Space Based on the n-tuple Algebra, I, Foundations of the n-tuple Algebra,” Avtom. Telemekh., No. 1, 126–136 (1997).

5. B. A. Kulik and M. V. Naumov, “Representation of Logical Systems in the Probabilistic Space Based on the ntuple Algebra, II, Measurable Logical Systems,” Avtom. Telemekh., No. 2, 169–179 (1997).

6. B. A. Kulik, “Reliability Analysis of Systems with Many States Based on the n-tuple Algebra,” Avtom. Telemekh., No. 7, 13–18 (2003).

7. N. J. Nilsson, “Probabilistic Logic,” Artif. Intell., No. 28, 71–87 (1986).

8. R. Fagin, J. Y. Halpern, and N. Megiddo, "Logic for Reasoning and Probability," Report RJ, No. 4, 1–41 (1988).

9. R. Fagin and J. Y. Halpern, "Uncertainty, Belief, and Probability," in Proceedings of 11th International Conference on Artificial Intelligence, Detroit, USA, 1989 (Morgan Kaufmann, Michigan, 1989), pp. 1161–1167.

10. I. A. Ryabinin, *Reliability and Safety of Structurally Complex Systems* (Politekhnika, St. Petersburg, 2000) [in Russian].

11. A. N. Kolmogorov, *Foundations of the Theory of Probability*, 2nd ed. (2nd ed., Nauka, Moscow, 1974; Chelsea, New York, 1956).

12. E. D. Solozhentsev, *Scenario Logical–Probabilistic Control of the Risk in Business and Industry* (Biznes-Pressa, St. Petersburg, 2004) [in Russian].