I T H E A

International Journal

INFORMATION CONTENT & PROCESSING

2017 Volume 4 Number 1

International Journal
INFORMATION CONTENT & PROCESSING
Volume 4 / 2017, Number 1

IJ ICP is official publisher of the scientific papers of the members of the ITHEA® International Scientific Society

IJ ICP rules for preparing the manuscripts are compulsory.
The **rules for the papers** for ITHEA International Journals as well as the **subscription fees** are given on www.ithea.org .
The papers should be submitted by ITHEA® Submission system http://ij.ithea.org .
Responsibility for papers published in IJ IMA belongs to authors.

International Journal "INFORMATION CONTENT AND PROCESSING" Volume 4, Number 1, 2017

Edited by the **Institute of Information Theories and Applications FOI ITHEA**, Bulgaria, in collaboration with
Institute of Mathematics and Informatics, BAS, Bulgaria,
V.M.Glushkov Institute of Cybernetics of NAS, Ukraine,
Universidad Politechnika de Madrid, Spain,
Hasselt University, Belgium
Institute of Informatics Problems of the RAS, Russia,
St. Petersburg Institute of Informatics, RAS, Russia
Institute for Informatics and Automation Problems, NAS of the Republic of Armenia.

Publisher: **ITHEA®**
Sofia, 1000, P.O.B. 775, Bulgaria. www.ithea.org, e-mail: info@foibg.com
Technical editor: Ina Markova

Printed in Bulgaria

ISSN 2367-5128 (printed) **ISSN 2367-5152 (online)**

FURTHER RESULTS ON DOUBLE ±1 ERROR CORRECTING CODES OVER RINGS Z_m

Gurgen Khachatrian, Hamlet Khachatryan

***Abstract**: In this paper further results on double ±1 error correcting codes over rings are presented. In particular optimal linear codes correcting ±1* type of errors over *rings Z_7 and Z_9 are constructed. A method allowing to construct (2N, 2N-6) double ±1error correcting codes over rings Z_m from the given (N, N-4) double ±1 error correcting codes over rings Z_m is also developed.*

***Keywords**: Error correcting codes, Codes over the rings* Z_7 and Z_9, Asymmetrical errors, Optimal codes.

1. Introduction

From practical point of view the codes over rings Z_{2m} or Z_{2m+1} are interesting, because they can be used in 2^{2m} – QAM **(**Quadrature amplitude modulation) schemes**.** Codes over finite rings, particularly over integer residue rings and their applications in coding theory have been studied for a long time**.** Errors happening in the channel are basically asymmetrical; they also have a limited magnitude and this effect is particularly applicable to flash memories.

There have been couple of papers regarding to optimal *±1* single error correcting codes over alphabet Z_m [Martirossian, 1996; Kostandinov, 2010]**.** Also there are many linear codes capable to correct up to two error of type *±1* for different alphabets which have been found by computer search, but there are not optimal. The optimality criteria for the linear codes over fixed ring Z_m can be considered in two ways. First of all, recall that the code of the length n is optimal-1 if it has a minimum possible number of parity check symbols. Secondly, optimality-2 criteria for the code is that for a given number of parity check symbols, it has a maximum possible length. The linear code (12, 8) correcting double error over ring Z_5 of value *±1* was presented in [Khachatrian, 2014] satisfies the optimality criteria -1:

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 1 & 2 & 3 & 4 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 & 2 & 2 & 2 & 2 & 2 & 1 & 1 \\ 3 & 2 & 4 & 4 & 2 & 3 & 2 & 4 & 4 & 2 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 3 & 2 & 4 & 4 & 2 & 0 & 4 \end{bmatrix},$$

this code was given by the parity check matrix *H*, which has 8 information and 4 parity check symbols.

At this point we do not know any codes that satisfy the optimality criteria-2. In [Khachatrian, 2014] a method how to compare two code constructions over different size of alphabets when both satisfy the optimality – 1 criterion has been presented. Two factors are considered, namely:

1) The first factor should be the rate of the code i.e. the ratio of the number of information symbols over the length of the code.
2) Second, the ratio between the number of possible amplitude errors corrected by the code over the size of alphabet minus 1, which corresponds to the number of all possible amplitude errors.

The product of these two factors is chosen as a merit to compare optimal codes over different size of alphabets.

For the code over ring Z_5 mentioned above the product is: (8/12) * (2/4) = 0.3333.

In this paper constructions of the optimal-1 codes (16, 12) and (20, 16) over the rings Z_7 and Z_9 correcting double ± 1 errors is presented. For these codes the products will be (12/16) * (2/6) = 0.25 for code over Z_7 and (16/20) * (2/8) = 0.2 for Z_9 . These products are a little bit smaller then for the code (12, 8) over ring Z_5 , although there are much better comparing with codes over Z_{16} and Z_{128} in [Kostandinov, 2010; Han Vinck, 1998; Kostandinov, 2008].

Moreover, in this paper a constructions of codes C (N, N-6), which are 2 times longer than previous ones, and have 6 parity check symbols are presented.

2. Construction of Optimal (16, 12) code over ring Z_7

Our purpose is to construct an optimal linear code over ring Z_7 correcting double errors of the type ± 1. It is well known, that a linear code given by the parity check matrix H , can correct up to two errors of the type ± 1, only when H has a property according to which all the syndromes resulting from adding and subtracting operations between any two columns of the matrix H are different $\left(\pm h_i \pm h_j , where \ (h_i \neq \pm h_j)\right)$. For constructing this kind of matrix H , at first we will find a difference set in Z_7. For example, a difference set for a linear code (12, 8) constructed in [Khachatrian, 2014] is the set {3, 2, 4, 4, 2}. A difference set is defined to have a property that the differences for any 2 components in the set are different in Z_5 given that difference is taken for the elements located at the same distance from each other where the distance itself can be from the set (1, 2, 3, 4). Note also that the distance between positions of elements is calculated modulo 5 in this case. For an example if the distance is chosen to be three, we have to take a difference between 4-th and 1-th positions of the set which is equal = 1, a difference between 5-th and 2-th positions of the set will be 0, a difference between 1-th(6-th) and 3-th positions of the set will be -1(4), a difference between 2-th(7-th) and 4-th positions of the set will be -2(3), and finally a difference between 3-th(8-th) and 5-th positions of the set will be 2.

For the ring Z_7 it is easy to check that the difference set is a set – {4, 3, 6, 6, 3, 4, 2} of the length 7.

For instance, for the distance equal to 1 all corresponding differences resulting from (3 – 4 = **6**, 6 – 3 = **3**, 6 – 6 = **0**, 3 – 6 = **4**, 4 – 3 = **1**, 2 – 4 = **5**, 4 – 2 = **2** (**mod7**)) are different. 4-corresponds to the

position with index 0 and the last position 2 corresponds to the position with index 6 and 0 – 6 = 1 (mod 7) (all operations are in Z_7).

A linear (16, 12) code over ring Z_7 is given by the following parity check matrix H:

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 1 & 1 \\ 6 & 5 & 4 & 3 & 2 & 1 & 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 \\ 4 & 3 & 6 & 6 & 3 & 4 & 2 & 4 & 3 & 6 & 6 & 3 & 4 & 2 & 1 & 6 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 4 & 3 & 6 & 6 & 3 & 4 & 2 & 0 & 0 \end{bmatrix}. \quad (2.1)$$

An approach how this matrix is designed is similar to one in [Khachatrian, 2014]. It consists of three parts, namely the first 7 columns, the next 7 columns and a tail of two last columns. The first two rows of the first and second parts is a code correcting one error of the type ± 1 third rows of the parts 1 and 2 as well as the fourth row of the second part is a difference set for Z_7. It can be checked that a linear code over Z_7 , given by the parity check matrix H in (1) can correct up to two errors of the type ± 1. This can be done in the similar manner demonstrated in [Khachatrian, 2014] and of course also by computer.

Lemma 1. *A linear code (16, 12) correcting up to two errors of the type ±1 is optimal in the sense that it has a minimal possible number of parity check symbols.*

Proof: In this case the number of combinations for each code word that can be corrected is

$$(1 + 16*2 + (16 \text{ choose } 2) * 4) = 513.$$

Thus, we have that $513 * 7^{12} \leq 7^{16}$ and the cardinality of the best possible code is $7^{16} / 513 < 7^{13}$.

3. Construction of Optimal (20, 16) code over ring Z_9

In this section we will construct an optimal (20, 16) linear code over ring Z_9. As same as in previous construction we need to find a difference set of length 9 for Z_9**.** In this case we could not find a difference set of length 9. So, to fix this problem we find a difference set of the length 8: {7,3,2,4,4,2,3,7}. Similarly in this set, for all distances (1, 2, 3, 4…) the differences of any 2 components should be different in Z_9. For instance, for the distance-1 all corresponding differences resulting from (3 - 7 = **5**, 2 - 3 = **8**, 4 - 2 = **2**, 4 - 4 = **0**, 2 – 4 = **7**, 3 – 2 = **1**,7-3=**4**(mod9)) are different (all operations are in Z_9).

In the previous construction, sequences consisting of all integers in Z_7 - {0, 1, 2, 3, 4, 5, 6} have been used in rows of the matrix. Since we have for Z_9 a difference set with only 8 components, we should take either a sequence {0,1,2,3,4,5,6,7} or {1,2,3,4,5,6,7,8}.

In this case the parity check matrix for an optimal linear code (20, 16) correcting double errors of the type ±*1* has the following form:

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 1 & 1 & 2 & 4 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 & 2 & 4 \\ 7 & 3 & 2 & 4 & 4 & 2 & 3 & 7 & 7 & 3 & 2 & 4 & 4 & 2 & 3 & 7 & 1 & 1 & 2 & 4 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 7 & 3 & 2 & 4 & 4 & 2 & 3 & 7 & 6 & 3 & 7 & 2 \end{bmatrix}. \quad (3.1)$$

As for previously constructed codes a corresponding parity check matrix (3.1) also consists of three parts. Since a difference set in this case has only 8 elements a corresponding split between those parts will be (8, 8, 4). This is because our target is to have a code of the length 20 which will be optimal and therefore we will need for the tail part to have 4 columns which is in fact the last four columns of the matrix (3.1). A linear code over Z_9, given by the parity check matrix *H* (3.1), can correct up to two errors of the type ±*1*. The proof of this statement can be done similarly by manner demonstrated in [Khachatrian, 2014], as well as by computer.

Lemma 2. *A linear code (20, 16) code given by* (3.1) *correcting up to two errors of the type ±1 is optimal in the sense that it has a minimal possible number of parity check symbols.*

Proof: In this case the number of combinations for each code word that can be corrected is

$$(1 + 20*2 + (20 \text{ choose } 2) * 4) = 801.$$

Thus, we have that $801 * 9^{16} \leq 9^{20}$ and the cardinality of the best possible code is

$$9^{20} / 801 < 9^{17}.$$

4. Construction of C (2N, 2N-6) codes based on optimal C (N, N-4) codes

In this section we will describe a method, which allows us to construct new codes with a double length at the expense of just two parity check symbols. We will assume that we have at our disposal (N, N-4) double error correcting code like codes constructed in part 2 in this paper as well as in [Khachatrian, 2014]. In this paper we construct codes of length N with 4 parity check symbols C (N, N-4). Using method, which will be described below, we can construct codes of length 2N with 6 parity check symbols C (2N, 2N-6). Further in this paper by codes we mean double error correcting code of the type *±1*.

Let C (N, N-4) be a code over ring Z_m . Our construction of a new code will have a parity check matrix with 6 rows and 2N columns where the first 4 rows will be just a repetition of 4 rows of the code C (N, N-4). Now we will describe how we add 2 additional rows for the parity check matrix. The first N columns and the last N columns of two additional rows will be referred as group 1 and a group 2 respectively. In the first row of group 1 we put integer 2 repeated n times then integer 1 repeated n times and then (N -2n) times any integer x from Z_n . In the second row of the group 1 we put all integers from Z_n{0, 1 ,…, n-1} repeated twice and then but the first (N-2n) elements of {0,1,…, n-1}. Consequently, in the first row of group 2 we put the second row of group 1, and in the second row of group 2 accordingly we put integers 3 and 4 repeated n times, and in the rest of positions the same integer x from group 1. Note that an integer x should differ from (1, 2, 3, 4) and must satisfy the condition ((*x + x ≠ 1 (mod n*)),

Group 1		Group 2
C (N, N - 4)		C (N, N - 4)

$$\begin{bmatrix} 2 & 2 & \dots & 2 & 1 & 1 & \dots & 1 & x & x & \dots & 0 & 1 & \dots & n-1 & 0 & 1 & \dots & n-1 & 0 & 1 & \dots \\ 0 & 1 & \dots & n-1 & 0 & 1 & \dots & n-1 & 0 & 1 & \cdots & 3 & 3 & \dots & 3 & 4 & 4 & \dots & 4 & x & x & \dots \end{bmatrix}$$

Now we can prove the following theorem.

Theorem 1: *For a given C (N, N-4) code over ring Z_n correcting double ±1 errors, it is possible to construct a code with 6 parity check symbols of a length C(2N, 2N-6) correcting ±1 double errors.*

Proof. In order to prove this theorem it must be shown that all corresponding syndromes resulting from operations *±1* between all columns of both groups should be all different. Let us split all columns of the group 1 into 3 subgroups, namely the first subgroup (subgroup 1.1) contains the first columns of group 1, which first component is 2, starting from the left, the second subgroup (subgroup 1.2) contains next n columns, which first component is 1, and the third subgroup (subgroup 1.3) contains the last columns with x. Accordingly, we can do the same with columns of group 2 and split them into three subgroups

(2.1, 2.2 and 2.3) in accordance with (3.1). We need to consider only those cases, when the first four components have the same syndromes. We will demonstrate the proof for the case, when the first error has upward direction (+1), and second downward (-1). Let us perform the proof by 3 cases:

1) Let's suppose that two errors occurred in the first group. Because the both parts of parity check matrix H consist of the same code in first 4 rows, we do not know in which part the errors occur: whether in the first part of matrix H or in the second one. We can check it using next 2 rows. There can be only 3 possible subcases:

1.1) If errors are in subgroup 1.1 the first position will always be 0, otherwise in group 2 it cannot be 0 (due to the property of the set {0,1,…, n-1}), and the syndromes will be different.

1.2) If one error occur in subgroup 1.1 and the second in subgroup 1.2 in group 1 the first position will be 1, but in group 2 the second position will be -1, and the syndromes will always be different, because we have the same components in two other positions (the second row of group 1 and the first row of group 2 are the same).

1.3) If one of the errors occur in subgroup 1.3, then if next is in subgroup 1.1 resulting syndromes will always be different, because in subgroup 1.1 the first position is 2 and in subgroup 2.1 the second position is 3, but we have the same components as in subgroups 1.3 and 2.3. Like the case b) the other two components of the syndrome always will be the same (the second row of group 1 and the first row of group 2 are the same). If second error occur in subgroup 1.2 (2.2) the way of the proof is the same.

Thus, the first case of the proof is complete.

2) Let one error occur in group 1 and second in group 2. We need to check whether both errors are in the same group or not. Again there can be 3 possible subcases:

2.1) Let an error occur in subgroup 1.1 and second error in subgroup 2.1. As the first four components of matrix H are in these subgroups we have the same columns the errors might be in the same subgroup. How we can distinguish these cases? If both occur in subgroup 1.1, then the first component will be 0, otherwise, in our case 0 can be only with column 3 of subgroup 2.1. In this case, in the third column of subgroup 1.1 the second position is 2, and in the same column of subgroup 2.1 it is 3, consequently the syndromes will be different (if both occur in subgroup 2.1 the proof is the same (the second component will be 0)).

2.2) Let one error occur in subgroup 1.1 and the second in subgroup 2.2. If both errors occur in group 1, then the first component will be 1, in our case 1 can be only with column 2 of subgroup 2.2, here the second component is 4, but in subgroup 1.2 it is 1, consequently the syndromes will be different. If both occur in group 2, then the second component will be -1, in our case -1 can be only with column 4 of subgroup 1.1, here the first component is 2, but in subgroup 2.1 it is 3, and syndromes will

be different. (For the case when errors occur in subgroup 2.1 and subgroup 1.2 the way of proof is the same).

2.3) Let one error occur in subgroup 1.1 or in subgroup 1.2 and the second in subgroup 2.3. If both errors occur in group 1 (subgroup 1.3) the resulting syndromes will be different, because in subgroups 1.3 and 2.3 the corresponding rows are swapped like $\begin{pmatrix} x & x & \dots \\ 0 & 1 & \dots \end{pmatrix}$ and $\begin{pmatrix} 0 & 1 & \dots \\ x & x & \dots \end{pmatrix}$. Consequently, when we subtract them from the same subgroup 1.1 or 1.2 the resulting syndromes will be different. (For the case when errors occur in subgroup 1.3 and in subgroups 2.1 or 2.2 the way of proof is the same).

Thus, the second case of the proof is complete.

3) In this case both of the errors occur in the same columns of different groups. Accordingly, the first four components of the syndromes will be (0 0 0 0). In this case the number of all possible syndromes will be 2N . Due to the selection of last two rows of matrix (group 1 and group 2), it can be shown that all 2N syndromes will be different. In this case the difference between the same columns for the first two subgroups of groups 1 and 2 will be if the second element of the last column is the same first elements will be different by two, while the difference between the same columns in corresponding third subgroups will be $\begin{pmatrix} x-i \\ i-x \end{pmatrix}$ and will be different for all i's unless 2x ≠1 (mod **n**) - which is the condition for x. This analysis completes the proof of the theorem.

Here we will show some results, which we have gotten using the method described by the theorem 1:

The code (24, 18) correcting double ±*1* errors, given by parity check matrix H_5 was resulted from the optimal code (12, 8) over ring Z_5 by adding 2 parity check symbols:

$$\begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 1 & 2 \\ 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 1 & 2 & 3 & 3 & 3 & 3 & 3 & 4 & 4 & 4 & 4 & 4 & 0 & 0 \end{bmatrix}$$

$$H_5 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 1 & 2 & 3 & 4 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 2 & 3 & 4 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 & 2 & 2 & 2 & 2 & 2 & 1 & 1 & 0 & 1 & 2 & 3 & 4 & 2 & 2 & 2 & 2 & 2 & 1 & 1 \\ 3 & 2 & 4 & 4 & 2 & 3 & 2 & 4 & 4 & 2 & 1 & 1 & 3 & 2 & 4 & 4 & 2 & 3 & 2 & 4 & 4 & 2 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 3 & 2 & 4 & 4 & 2 & 0 & 4 & 1 & 1 & 1 & 1 & 1 & 3 & 2 & 4 & 4 & 2 & 0 & 4 \\ 2 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 1 & 2 \\ 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 1 & 2 & 3 & 3 & 3 & 3 & 3 & 4 & 4 & 4 & 4 & 4 & 0 & 0 \end{bmatrix}.$$

The code (32, 26) correcting double ±*1* errors, given by parity check matrix H_7 was resulted from the optimal code (16, 12) over ring Z_7 by adding 2 parity check symbols:

$$\begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 5 & 5 & & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 5 & 5 \end{bmatrix}$$

$$H_7 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 1 & 1 \\ 6 & 5 & 4 & 3 & 2 & 1 & 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 \\ 4 & 3 & 6 & 6 & 3 & 4 & 2 & 4 & 3 & 6 & 6 & 3 & 4 & 2 & 1 & 6 & 4 & 3 & 6 & 6 & 3 & 4 & 2 & 4 & 3 & 6 & 6 & 3 & 4 & 2 & 1 & 6 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 4 & 3 & 6 & 6 & 3 & 4 & 2 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 4 & 3 & 6 & 6 & 3 & 4 & 2 & 0 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 5 & 5 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 5 & 5 \end{bmatrix}.$$

And lastly, the code (40, 34) correcting double ±*1* errors, given by parity check matrix H_9 was resulted from the optimal code (20, 16) over ring Z_9 by adding 2 parity check symbols:

$$\begin{bmatrix} \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{6} & \mathbf{6} & & \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{0} & \mathbf{1} & & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{6} & \mathbf{6} \end{bmatrix}$$

$$\mathbf{H_9} \begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{8} & \mathbf{7} & \mathbf{6} & \mathbf{5} & \mathbf{4} & \mathbf{3} & \mathbf{2} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{2} & \mathbf{4} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{8} & \mathbf{7} & \mathbf{6} & \mathbf{5} & \mathbf{4} & \mathbf{3} & \mathbf{2} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{2} & \mathbf{4} \\ \mathbf{7} & \mathbf{6} & \mathbf{5} & \mathbf{4} & \mathbf{3} & \mathbf{2} & \mathbf{1} & \mathbf{0} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{1} & \mathbf{1} & \mathbf{2} & \mathbf{4} & \mathbf{7} & \mathbf{6} & \mathbf{5} & \mathbf{4} & \mathbf{3} & \mathbf{2} & \mathbf{1} & \mathbf{0} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{1} & \mathbf{1} & \mathbf{2} & \mathbf{4} \\ \mathbf{7} & \mathbf{3} & \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} & \mathbf{3} & \mathbf{7} & \mathbf{7} & \mathbf{3} & \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} & \mathbf{3} & \mathbf{7} & \mathbf{1} & \mathbf{1} & \mathbf{2} & \mathbf{4} & \mathbf{7} & \mathbf{3} & \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} & \mathbf{3} & \mathbf{7} & \mathbf{7} & \mathbf{3} & \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} & \mathbf{3} & \mathbf{7} & \mathbf{1} & \mathbf{1} & \mathbf{2} & \mathbf{4} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{7} & \mathbf{3} & \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} & \mathbf{3} & \mathbf{7} & \mathbf{6} & \mathbf{3} & \mathbf{7} & \mathbf{2} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{7} & \mathbf{3} & \mathbf{2} & \mathbf{4} & \mathbf{4} & \mathbf{2} & \mathbf{3} & \mathbf{7} & \mathbf{6} & \mathbf{3} & \mathbf{7} & \mathbf{2} \\ \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{2} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{6} & \mathbf{6} & \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{0} & \mathbf{1} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{3} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{6} & \mathbf{6} \end{bmatrix}$$

5. Conclusion

In this paper further results on construction of linear double ±*1 error correcting codes are presented. In particular a technique is developed which allows to double the length of the codes at the cost of just two parity check symbols. That technique will allow to construct codes with higher rates.*

Bibliography

[Han Vinck, 1998] A.J.Han Vinck and H.Morita. Codes over the ring of integers modulo m // *IEICE Trans.Fundamentals*, vol. E81-A, pp.2013-2018, 1998.

[Khachatrian, 2014] G.Khachatrian and H. Morita. Construction of optimal ± 1 double error correcting linear codes over ring Z_5 // *3th International Workshop on Advances in Communications*, Boppard, Germany, pp. 10-12, May 2014.

[Kostandinov, 2008] H.Kostadinov, N.Manev and H. Morita. Double ± 1-error correctable codes and their applications to modulation schemes// *Proc. Elev. Intern. Workshop ACCT*, Pamporovo, June 16-22, pp. 155-160, 2008.

[Kostandinov, 2010] H.Kostadinov, N.Manev and H.Morita. On ± 1 error correctable codes// *IEICE Trans.Fundamentals//* vol.E93-A, pp.2578-2761, 2010.

[Martirossian, 1996] S.Martirossian. Single error correcting close packed and perfect codes // *Proc.1st INTAS Int. Seminar Coding Theory and combinatorics,* Armenia, pp.90-115,1996.

Authors' Information

Gurgen Khachatrian *- professor, American University of Armenia Yerevan, Armenia; e-mail: gurgenkh@aua.am*

Hamlet Khachatryan *- Ph.D. student at National Academy of Sciences, Junior Researcher at Institute for Informatics and Automation Problems of NAS Armenia, 1, P.Sevaki Str., 0014, Yerevan, Armenia; e-mail: hamletxachatryan08@gmail.com*

RESEARCH ON THE CRITERIA FOR THE STRUCTURE OF THE USED IN THE IDA ALGORITHM P FUNCTION

Ivan Ivanov, Stella Vetova, Krasimira Ivanova, Kiril Aleksiev, Lubov Ilieva

Abstract: *The following paper presents some conducted extensive research on the cryptographic algorithm IDA, concerning one of the basic properties of the block algorithms "diffusion." The object of the studies is the criteria for the structure of the function P used in this algorithm.*

Keywords: *cryptography, cryptographic algorithm, P function, S matrix, IDA algorithm.*

ITHEA Keywords: *E.3 Data Encryption – cryptosystems; F. Theory of Computation: F.2 Analysis of Algorithms and Problem Complexity; K. Computing Milieux: K.7 The Computing Profession: K.7.3 Testing, Certification, and Licensing.*

Introduction

The purpose of the presented paper is the exploration of the used in the IDA algorithm criteria for the structure of the P function [Ivanov, 2014] in the following main tasks: (1) introduction of the criteria for the structure of the P function; (2) research on the structure of the P function on the base of the set criteria; (3) results analysis.

There are three criteria used in the process of development of the P function as follows [Stallings, 2013; Schneier, 2013; Sokolov & Shangin, 2002]:

1. The four output bits obtained as a result of the S matrix in the i-th loop should be distributed so that two of them might affect the middle bits of the i + 1 loop, and the other two bits to affect the final ones.
2. The four output bits of the S matrix in the next loop should affect the results of six different S matrices, and none of the pairs of these four output bits should not come to the input of any S matrix.
3. For the two S matrices, Si and Sk, if any of the output bits of the Si matrix in the next loop affects the middle bits of Sk, then none of the output bits of Sk, should affect the middle bits of Si.

Using the described three criteria the algorithm satisfies the requirement for the property "diffusion" [Katz & Lindell, 2014].

The structure of the P function

Besides its application for bits rearranging, the P function (P) is applied for the determination of the encryption function setting the values in Table 1. In the starting sequence, the conversion runs in the following order:

1. Bit with number sixteen takes first position;
2. Bit with number seven takes the second position;
3. Bit with number twenty takes the third position, etc.

Table 1. Conversion function (P)

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Research on the structure of the P function in the IDA algorithm, on the base of the set criteria

The next research uses IDA algorithm with the following input data:

P=11110010 11100101 11101011 11100101 11110100 11101110 11101101 11101000

K=11101010 11101110 11110000 11100101 11101010 11110010 11101110 11110000 00100000 11101101 11100000 00100000 11110010 11100101 11101011 11100101 11101010 11101110 11101100 11110011 11101101 11101000 11101010 11100000 11110110 11101000 11101110 11101101 11101101 11101000 11110010 11100101

In S_{0R} = 00010001 00001111 11011111 00011101 01100001 00101011, the obtained result is:

S_1(in)=000100; 00 - line number, 0010 - column number, S_1(out)=13=1101;

S_2(in)=010000; 00 - line number, 1000 - column number, S_2(out)=9=1001;

S_3(in)=111111; 11 - line number, 1111 - column number, S_3(out)=12=1100;

S_4(in)=011111; 01 - line number, 1111 - column number, S_4(out)=9=1001;

S_5(in)=000111; 01 - line number, 0011 - column number, S_5(out)=12=1100;

S_6(in)=010110; 00 - line number, 1011 - column number, S_6(out)=4=0100;

S_7(in)=000100; 00 - line number, 0010 - column number, S_7(out)=2=0010;

S_8(in)=101011; 11 - line number, 0101 - column number, S_8(out)=10=1010;

These results represent the output sequences of the eight S matrices (boxes). The sequences are applied to the input of the conversion function P (Table 1).

$S_{0R,32}$ = 11011001 11001001 11000100 00101010

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$S_{0R,32}$	1	1	0	1	1	0	0	1	1	1	0	0	1	0	0	1	1	1	0

№	20	21	22	23	24	25	26	27	28	29	30	31	32
$S_{0R,32}$	0	0	1	0	0	0	0	1	0	1	0	1	0

After applying the function P on $S_{0R,32}$, the obtained result P_{0R} is:

P_{0R} = 10001001 10001111 11000101 01001010

In order to continue to the second loop, it is necessary that the XOR-based adder should be used first on the base of $P_{0R,32}$ and K_4, to obtain R_0:

for $P_{0R,32}$ and K_4=11101010 11101110 11101100 11110011 the result is:

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_{0R}	1	0	0	0	1	0	0	1	1	0	0	0	1	1	1	1	1	1	0
K_4	1	1	1	0	1	0	1	0	1	1	1	0	1	1	1	0	1	1	1
R_0	0	1	1	0	0	0	1	1	0	1	1	0	0	0	0	1	0	0	1

№	20	21	22	23	24	25	26	27	28	29	30	31	32
P_{0R}	0	0	1	0	1	0	1	0	0	1	0	1	0
K_4	0	1	1	0	0	1	1	1	1	0	0	1	1
R_0	0	1	0	0	1	1	0	1	1	1	0	0	1

R_0 = 01100011 01100001 00101001 10111001

Second loop

Since at each next loop the left and the right parts replace, the right sequence R_0 leaves labeled L_1:

L_1 = 01100011 01100001 00101001 10111001

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
L_1	0	1	1	0	0	0	1	1	0	1	1	0	0	0	0	1	0	0	1

№	20	21	22	23	24	25	26	27	28	29	30	31	32
L_1	0	1	0	0	1	1	0	1	1	1	0	0	1

This sequence goes through the expansion function E. The result is the sequence E_{1L} [Ivanov et al., 2014]:

E_{1L} = 10110000 01101011 00000010 10010101 00111101 11110010

Then, addition using XOR-based adder on the base of E_{1L} and K_5 is applied to obtain S_{1L}:

Exclusive OR of the sequences E_{1L} and K_5:

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
E_{1L}	1	0	1	1	0	0	0	0	0	1	1	0	1	0	1	1	0	0	0
K_5	1	1	1	0	1	1	0	1	1	1	1	0	1	0	0	0	1	1	1
S_{1L}	0	1	0	1	1	1	0	1	1	0	0	0	0	0	1	1	1	1	1

№	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
E_{1L}	0	0	0	1	0	1	0	0	1	0	1	0	1	0	0	1	1
K_5	0	1	0	1	0	1	1	1	0	0	0	0	0	1	1	1	1
S_{1L}	0	1	0	0	0	0	1	1	1	0	1	0	1	1	1	0	0

№	37	38	39	40	41	42	43	44	45	46	47	48
E_{1L}	1	1	0	1	1	1	1	1	0	0	1	0
K_5	0	1	1	0	1	1	1	0	1	0	0	0
S_{1L}	1	0	1	1	0	0	0	1	1	0	1	0

S_{1L} = 01011101 10000011 11101000 01110101 11001011 00011010

The sequences of S_{1L} are submitted to the inputs of the eight S boxes:

S_1(in)= 010111; 01 - line number, 1011 – column number, S_1(out)=11=1011;

S_2(in)= 011000; 00 - line number, 1100 - column number, S_2(out)=12=1100;

S_3(in)= 001111; 01 - line number, 0111 - column number, S_3(out)=10=1010;

S_4(in)= 101000; 10 - line number, 0100 - column number, S_4(out)=12=1100;

S_5(in)= 011101; 01 - line number, 1110 - column number, S_5(out)=8=1000;

S_6(in)= 011100; 00 - line number, 1110 - column number, S_6(out)=5=0101;

S_7(in)= 101100; 10 - line number, 0110 - column number, S_7(out)=7=0111;

S_8(in)= 011010; 00 - line number, 1101 - column number, S_8(out)=0=0000;

The obtained results represent the output sequences of the eight S matrices (boxes). These sequences are applied to the input of the conversion function P (Table 1).

$S_{1L,32}$ = 10111100 10101100 10000101 01110000

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$S_{1L,32}$	1	0	1	1	1	1	0	0	1	0	1	0	1	1	0	0	1	0	0

№	20	21	22	23	24	25	26	27	28	29	30	31	32
$S_{1L,32}$	0	0	1	0	1	0	1	1	1	0	0	0	0

After applying the P function on $S_{1L,32}$, the obtained result P_{1L} is:

P_{1L} = 00000011 10011000 00110111 01011110

To continue to the third loop, it is necessary to go through the XOR-based adder on the base of $P_{1L,32}$ and K_7 to receive L_1:

For $P_{1L,32}$ and K_7 = 11001011 11010101 11100101 11011101 the result is:

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_{1L}	0	0	0	0	0	0	1	1	1	0	0	1	1	0	0	0	0	0	1
K_7	1	1	0	0	1	0	1	1	1	1	0	1	0	1	0	1	1	1	1
L_1	1	1	0	0	1	0	0	0	0	1	0	0	1	1	0	1	1	1	0

№	20	21	22	23	24	25	26	27	28	29	30	31	32
P_{1L}	1	0	1	1	1	0	1	0	1	1	1	1	0
K_7	0	0	1	0	1	1	1	0	1	1	1	0	1
L_1	1	0	0	1	0	1	0	0	0	0	0	1	1

L_1 = 11001000 01001101 11010010 10000011

Research results

Results on the base of criteria 1

Taking into consideration the performed research work, the following results are obtained: in $S_{0R,32}$ = 11011001 11001001 11000100 00101010, the result is P_{0R} = 10001001 10001111 11000101 01001010, and in $S_{1L,32}$ = 10111100 10101100 10000101 01110000 the result is P_{1L} = 00000011 10011000 00110111 01011110. These results indicate that this criterion is one hundred percent satisfied. This comes from the fact that the distribution of the four separate output bits obtained as a result of the S matrix is such that two of them affect the middle bits of the i + 1 loop, and the other two affect the final ones.

Results on the base of criteria 2

On the base of the performed research work, the following results are obtained: in $S_{0R,32}$ = 11011001 11001001 11000100 00101010, the result is P_{0R} = 10001001 10001111 11000101 01001010, and in $S_{1L,32}$ = 10111100 10101100 10000101 01110000 the result is P_{1L} = 00000011 10011000 00110111 01011110. These results indicate that this criterion is one hundred percent satisfied. The reason is that the four output bits of the S matrix in the next loop, affect the results of six different matrices S and no pair of these four output bits comes to the input of the S matrix.

Results on the base of criteria 3

The performance of the research work produces the following results: in $S_{0R,32}$ = 11011001 11001001 11000100 00101010, the result is P_{0R} = 10001001 10001111 11000101 01001010, and in $S_{1L,32}$ = 10111100 10101100 10000101 01110000 the result is P_{1L} = 00000011 10011000 00110111 01011110. These results indicate that this criterion is one hundred percent satisfied. This comes from the fact that for any two S matrices, S_i and S_k, if any of the output bits of the S_i matrix in the next loop affect the middle bits of S_k, then no output bit of S_k, affects the middle bit of S_i.

Conclusion

The described research work and obtained results lead to the following three conclusions:

1. The set criteria for the P function used in the IDA algorithm are one hundred percent satisfied;
2. One of the basic properties of the block algorithms called “diffusion” is realized. This means that if there is a key and two clear texts which differ by only one bit, the produced encrypted texts are totally different. Therefore, each bit of the ciphertext depends on all the bits of the clear text;
3. High degree of protection against differential cryptographic analysis is applied.

Acknowledgements

The paper is published with partial financial support from the "Scientific Research Fund" of University of Telecommunications and Posts, Sofia, Bulgaria, by the research project “Methods for development and estimation of cipher functions in block encryption algorithms”.

Bibliography

[Ivanov et al., 2014] Ivanov I, Arnaudov R, Dikov D, Stanchev G, Patent application 111513: Method for increasing data security in storage and during information transmission in special purpose telemetry systems, Bulgarian patent office, Official Bulletin, Issue 12, pp.14, Dec 2014.

[Katz & Lindell, 2014] Katz J., Lindell Y. Introduction to Modern Cryptography, Second Edition (Chapman & Hall/CRC Cryptography and Network Security Series), CRC Press, 2014.

[Schneier, 2013] Schneier B., Appliet Cryptography Protocols, Algorithms, and Source Code in C, Wiley, 2013.

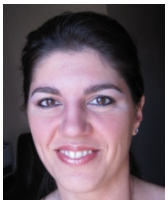
[Sokolov & Shangin, 2002] Sokolov V., Shangin F. Information protection distributed corporate networks and systems. DMK Press, Moskva, 2002.

[Stallings, 2013] Stallings W. Cryptography and Network Security: Principles and Practice (6th Edition), Hardcover, 2013.

Authors' Information

Ivan Ivanov *– Assist. Prof. PhD; University of Telecommunications and Posts, Sofia, Bulgaria; e-mail: i.ivanov@utp.bg;*

Major Fields of Scientific Research: Information and Network Security, Cryptographic Methods and Algorithms, Cyber security.



Stella Vetova *– Scientific Researcher, e-mail: vetova.bas@gmail.com;*

Major Fields of Scientific Research: Databases and security, Artificial Intelligence, Computer networks.

Krassimira Ivanova *- Assoc. prof. Dr.; University of Telecommunications and Posts, Sofia, Bulgaria; Institute of Mathematics and Informatics, BAS, Bulgaria;*
e-mail: krasy78@mail.bg;
Major Fields of Scientific Research: Software Engineering, Business Informatics, Data Mining, Multidimensional multi-layer data structures in self-structured systems

Kiril Aleksiev *– Assoc. Prof. PhD; University of Telecommunications and Posts, Sofia, Bulgaria; e-mail: kiril.m.alexiev@gmail.com;*
Major Fields of Scientific Research: Information Technology, Computer networks, Project management.

Lubov Ilieva *– Assoc. Prof. PhD; University of Telecommunications and Posts, Sofia, Bulgaria; e-mail: l.ilieva@utp.bg;*
Major Fields of Scientific Research: Psychology, Philosophy, Social Psychology, Business communications, Management Communicational Skills.

'GAME OF LIFE' WITH MODIFICATIONS: NON-REGULAR SPACE, DIFFERENT RULES AND MANY HYERARCHICAL LEVELS

Lois Facchetti, Alexander Makarenko

***Abstract**: The results of computer investigations of CA with 'game of 'Life' models, but on the cellular space with the 'whole', which have different shapes and different distribution in the space. We give results of experiments and discussed some correlations in behavior with parameters of non-homogeneities. Also the results of different rules of CA influence are presented. And finally the results for some hierarchical CA are described. Proposed results may be useful for understanding of non-classical CA behavior which may be interesting for some problems of cellular computing.*

***Keywords**: cellular automata; non-regular space; non-classical rules; hierarchical CA; computing; anticipation.*

1 Introduction

Cellular automata (CA) since past century were one of the very important fields in theoretical computer science [1-8]. Also cellular automata have many very important applications, for example, in physics [9], in crowds movement modeling [10-12], in pattern recognition [13], in brain investigations [14-16], in quantum mechanics [17, 18], in computation theory [19].

Usually all investigations focused around the most know CA – the game 'Life' by J. Conway, or some their modification, for example with probability accounting in the rules [2, 8, 11, 15]. But recently as in the theory as in the practice more essential improvement in CA arise: evolving and adaptive rules [20], asynchrony in the rules operation [21], decline from absolute homogeneities in rules for all cells [22], using of fixed similar neighbors for all cells and relies from full regularity of basic cellular space.

The letter property corresponds to common non-homogeneity of real physical spaces, problems, processes in the Nature – for example of percolation [23, 24]. CA models for such problems should correctly account such non-homogeneity. Presumable introduction non-homogeneity of basic space

follows for unknown changes s in CA behavior. It posed the problems of understanding qualitative and quantitative influence of non-regularity of basic space on CA properties.

So in given paper we propose the description of some our investigation of such problem. We describe the results of our computer investigations of CA with 'game of 'Life' models, but on the cellular space with the 'whole', which have different shapes and different distribution in the space. We give results of experiments and discussed some correlations in behavior with parameters of non-homogeneities. Also the results of different rules of CA influence are presented. And finally the results for some hierarchical CA are described. Proposed results may be useful for understanding of non-classical CA behavior which may be interesting for some problems of cellular computing.

2 Description of cellular automata model on non-regular space

In classical Game of Life cellular automata, we tried to determine a rule between, on the one hand, the percentage and distribution of initial wall cells and, on the other hand, the average time steps before reaching a temporal cycle in the cellular automata.

In the numerical simulations below, we adopted the model of classical Game of Life rules, with a classical Game of Life neighborhood for the cells. Yet, we introduced a certain amount of cells considered as "walls". These walls are assigned a value at the initialization of the cellular grid. This value determines the behavior of the cells right next to a wall cell. In most cases, these walls are frozen - although in two particular cases, the walls introduced could act as classical alive cells. When the distribution of walls is set, we study the number of steps up to which the cellular automata reach a cyclic evolution. For each value taken by the walls, we draw a two dimensional graph representing the average time before cycle. It enables studying the speed for cellular automata to reach a cycle in function of the initial distribution of the walls.

2.1. Modification of 'Game of Life'

2.1.1 'Game of Life' neighborhood

We adopted the Game of Life classical model for cellular automata. It consists of a Moore neighborhood. At each step, the cell considered evolves accordingly to state of its eight surrounding cells.

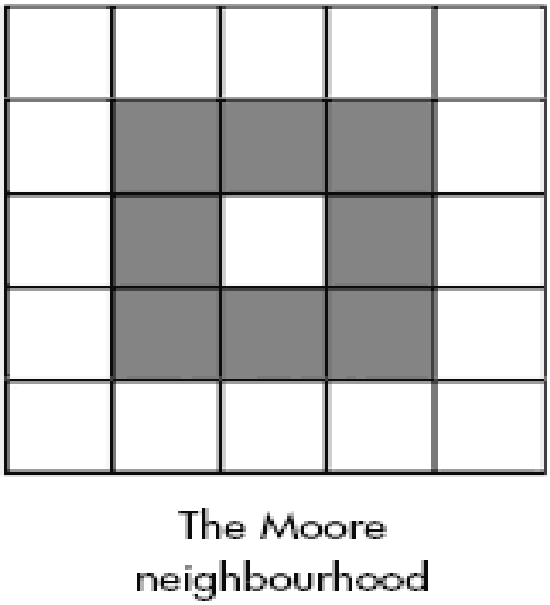


Fig. 1 Moor's neighbor of given cell (eight gray cells)

2.1.2 'Game of Life' classical rules

We adopted the Game of Life classical rule for every cell. At each time step, a cell can be in one of the two states 'dead' or 'alive' (or 0 or 1). Each cell evolves according to its own state and to its eight neighbor's states. Here is the rule to update a cell:

- an 'alive' cell at time t-1 remains alive at time t if 2 or 3 of its neighbors are also alive. Otherwise, it becomes 'dead' at time t.
- a 'dead' cell at time t-1 becomes alive at time t if 3 of its neighbors are alive. Otherwise, it remains 'dead' at time t.

In our model, a cell is represented by a Boolean value equal to 1 if the cell is alive and 0 if the cell is dead. Hence, the rule can be rewritten as a function of the sum of its neighbor's states. Let's consider that C(t) represents the state of one cell at time t, and NS(t) represents the neighborhood sum of values.

We can rewrite the Game of Life rule as:

- **if** C(t-1)=1 and 1<NS(t-1)<4 **then** C(t)=1, **else** C(t)=0
- **if** C(t-1)=0 and 2<NS(t-1)<4 **then** C(t)=1, **else** C(t)=0

Introducing strict inequalities will be necessary for the establishment of wall behavior.

2.1.3 Border conditions

The border conditions are 2D periodic, in other words our 2D cellular automata is a manifold of a 3D torus. For cellular automata composed of a NxN squared grid, considering C(i,j) the value of cell at t line i column j, we impose:

- C(N+1,k) = C(1,k) for k in {1,..,N}
- C(k,N+1) = C(k,1) for k in {1,..,N}
- C(0,k) = C(N,k) for k in {1,..,N}
- C(k,0) = C(k,N) for k in {1,..,N}

Defining a C(i,j) with i>N , j>N, i<0 , j<0 is necessary for the definition of the neighborhood of border cells.

2.1.4 Initial distribution of cell's states

For the numerical simulations, we will consider that the initial available grid -i.e. the whole grid deprived with the "wall" cells- is filled with uniformly-distributed fifty percents of alive cells.

2.1.5 Introduction of the walls into homogeneous cellular space

Now, we need to define the "walls" of our model (or distribution of the wholes).

A rule for the cells near the 'walls'.

Here we describe one of the possible rules for evolution for the cells near the walls. Remark that applications the rules for the cells near walls may correspond to physical nature of considered problems. Contrarily to normal evolving cells, "wall" cells can take any real value. This hypothesis describes entirely the model adopted, without having to change the previously enunciated rules. Keeping the previously introduced notations, the rules remain:

- **if C(t-1)=1 and 1<NS(t-1)<4 then C(t)=1, else C(t)=0**
- **if** C(t-1)=0 and 2<NS(t-1)<4 **then** C(t)=1, **else** C(t)=0

Both of the previous conditions are still implementable whether the values in the sum of neighbors are different from 0 or 1.

We could add the explicit condition: if the cell is a wall, then the cell remains a wall. Using, the previous notations:

— if C(t-1) ≠ 0 and C(t-1) ≠ 1, then C(t) = C(t-1)

It appears that this ultimate condition is optional. By setting such values to wall cells, the behaviors of the cells near to walls cells are changed. These local changes affect the whole cellular automata. Note that without changing the above conditions, if the walls have the value 0 or 1 at initial time step, they behave as normal cells in the following step.

2.1.6 Description of distribution and of wall's shape

In our simulation we for simplicity consider that the walls are square-shaped group of cells. Before each cellular automata simulation, we define a size of square and a percent of wall cells among the whole grid. The variable *s* represents the size of the square of surface containing s^2 wall cells. The percent represents the total amount of wall cells divided by the total number of cells in the grid.

At initialization time, the grid is full of empty cells. In given investigation we add successively one randomly placed *s*-sized square to the grid, till the percent of walls cells initially defined is exceeded. The effective percentage (in general different from the intended one) is eventually computed.

Here are presented the 6 first steps leading to the construction of the wall cells. The effective percent of wall cells is indicated. The intended percentage was 50%, the size of square is 10. The size of the grid 30x30.

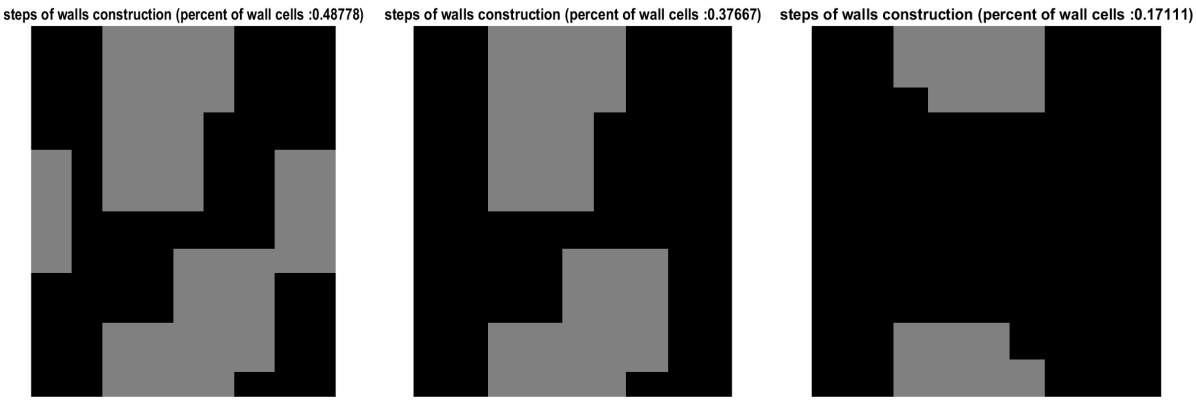


Fig. 2 Steps of wall construction

2.2. Numerical simulations

All the simulations were realized using Matlab software. In this section, we present the model, the parameters set and some results of the experiments.

2.3 Computing model

The cellular automata were modeled using matrix of size N^2xT, with **N** representing the side of the square grid, and *T* the maximal number of time step. The cell can take real values, 0 for dead cells, 1 for alive cells and arbitrary real number for walls different from 0 and 1. Each N^2 submatrix represents the state of the cellular automata at a particular time. The submatrix of the cellular automata at time t is represented by the cells of coordinates (*i,j,t*), with *i,j* in {1,...,N}, and *t* in *T*. By analogy with Matlab representation, we will note this matrix M(:,:,t).

Initialization During the initialization step, M(:,:,1) is randomly and equally filled with 0 and 1. Next, the wall cells are created according to the process presented in the model presentation. The wall cells values replaces the values of the cells in M(:,:,1) matrix. At this point, M(:,:,1) is filled with at most three different values : 0 (dead cells), 1 (alive cells), an arbitrary value (wall cells).

Update At each step, each cell of M(:,:,t) is updated according to its neighbors at time t-1, according to the Game of Life rules.

End The cellular automata can stop evolving for two reasons:

- the cellular automata reached the final time step $t=T$. *T* is returned.
- The cellular automata reached a cycle state. The time *t* at which the cyclic state was detected and the length l of the cycle is returned.

2.4 Parameters in the model

In this section, we will define the parameters used for the numerical computation.

a. size and space of the cellular automata

We used a 30x30x500 Matrix to model our cellular automata, i.e. a grid of 30x30 and a maximal time step of 500. These measures offer a good tradeoff between complexity of the evolution and computation speed. In the majority of the cases, the cellular automata reached a cyclic state before 500 iterations

step, which is necessary to truncate as few as possible the cellular automata evolution, which could perturb the data.

b. detect cycles in the model

We enable the algorithm to detect a cycle of at maximum 12 time step. In practical experiments, this number seems appropriate as most of cycle of length 8 and above is very rare, yet present.

2.5 Evolution of one CA over time

Here, we present the visual representation of a cellular automata evolving along the time. Some shots are represented, including the initial Matrix and the final one (i.e. the one at which we detected a cycle).

Black points represent dead cells, white one alive cell, and grey on wall cells. Initial intended percent of wall is 0.40.

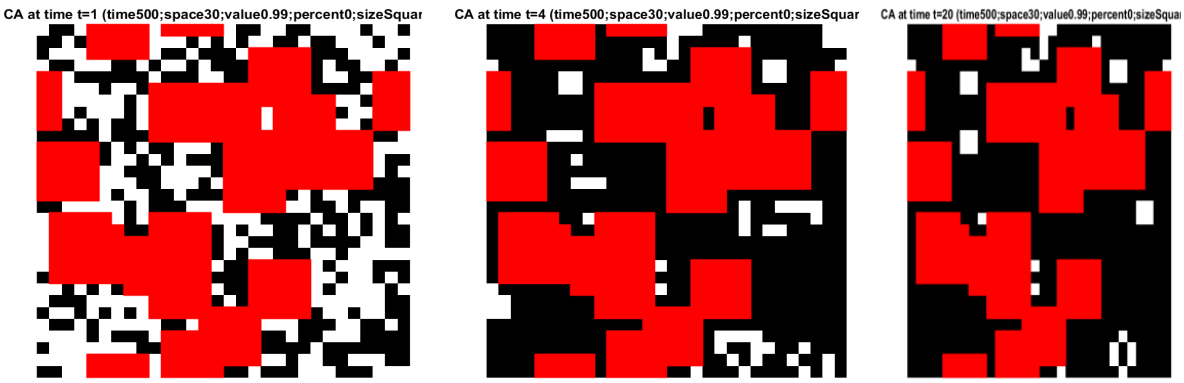


Fig. 3 Example of CA evolution during time

2.6 Parameters in the experiments

After having seen evolution of such cellular automata along time, we now intend to draw a rule binding the time to reach cycle and the initial distribution of walls. Therefore, we will draw a plot representing the average length time to reach cycle, in function of two parameters: size of square walls and percent of

wall cells. To do this, we will represent the length time with the intensity (black = 0, white = max) on an image with coordinates x representing the size of wall squares, coordinates y representing the percent of wall cells.

2.6.1 Number of simulation and values of the walls

We simulated cellular automata for eleven different values for the wall. These values have been chosen to observe behavior as diverse as possible. Here are the values studied: {-1, -0.01, 0, 0.01, 0.5, 0.90, 1, 1.01, 1.5, 1.99, 2}.

For each of these peculiar values, the simulation computed whether 20,000 or 40,000 different evolution of cellular automata, depending on the time the simulation was taking. The simulation returned as many values for *t* -the time before reaching cycle- and l -the length of the cycle.

2.6.2 Percent and size of wall squares

For every value of the walls, we obtained 20,000 or 40,000 couple of values (*t, l*) which were obtained by setting different parameters:

- *p* (percent of wall cells) : going from 0.001 to 1.000.
- *s* (size of the square walls) : integer values from 1 to *sqrt(p)*N*.

There are two nested loops: the first one runs on *p*, the second one runs on *s*. For a fixed p, we simulate all the integer size of square s, satisfying the condition: $s^2/N^2 < p$. Note that here, *p* is the intended percent, not the effective one. We need to return the effective percent p_e of walls at initial time.

When we obtained the 20,000 (or 40,000) values for the value wall (*t, l*) in function of (p_e, *s*), we classified them according to their length cycle l.

Next, we averaged them by grouping the value which had the same size s and the same integer part of effective percent. For example, if one value was computed with initial wall parameter *s*=2, p_e=10.68 and another for *s*=2, p_e=10.55, then they are averaged together.

At this point, for each value of walls, we have matrices M_l of values $M_l(p, s)$ with p in {1,...,100} and s in {1,...,N} representing the average time for cellular automata to reach cycle for *p* percent and *s* size of walls. Once these matrix M_l are obtained, we need to visualize them.

2.6.3 Representation

For each value of walls and for each cycle, the matrices have been visualized thanks to gray scaled images, where the intensity of the pixels represents the relative average time steps. “Relative” means that for each Matrix, we extracted the highest time step value t_m and divided every time step value t by this value, to obtain a ratio r going from 0 to 1, representing the relative average time steps. In order to grant highest values a sharper importance, we also applied a function to these [0,1] values to get r_f value: $r_f = r^{3/4}$. Finally, on an image of size 100x100, we represent the r_f intensities.

2.7 Results

In this section we present and explain some of the images simulated. Here is the examples image representing the 30x30 cellular automata for walls value of 0.01, ending by a cycle of length 1. We can read that the maximum average time for these automata is 480. It is the white point on the image, reached for percent around 75% and size around 70). Note that percent is 0 at the top and 1 at the bottom of the image, while size of square is 0 on the left and 100 on the right. Here the cycle “-1” means that these cellular automata did not reach a cycle within the T=500 time steps. Here the global-cycle means all the cycles of the walls value 0.99 have been represented on the same graph.

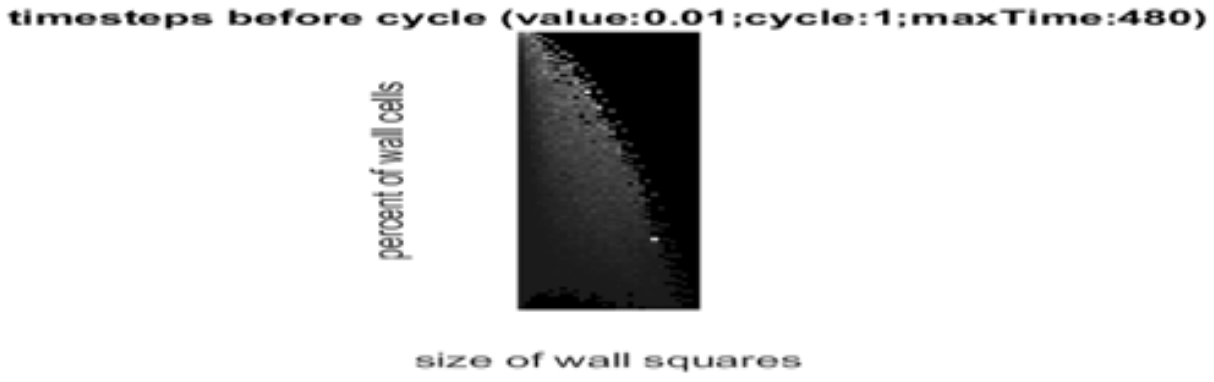


Fig. 4 Cycle

2.8 Analysis of the curves

In this section, we briefly discuss the global trend occurring in the graphs.

2.8.1 Global cycling

This subsection observes the global behaviors of the cellular automata without pointing out at precise cycle length, but taking all of them into account. Three main ideas can be noted.

First, there seems to be a global common distribution of average time steps before cycle stabilization. With a higher time step for small percent value of walls and small values of wall squares size. This image seems to be the archetype example of the typical global distribution.

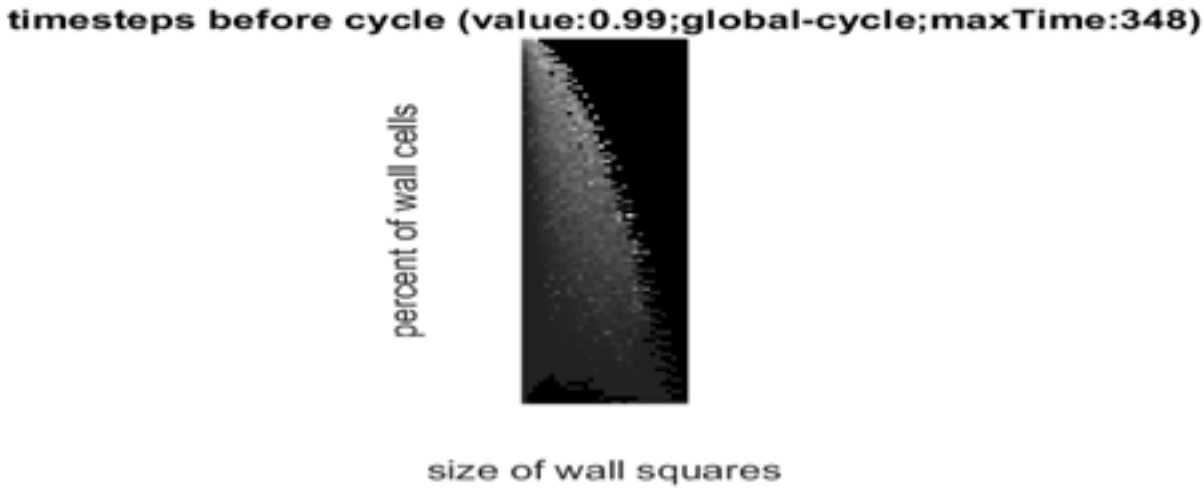


Fig. 5 Global cycling under value 0.99

Second, as we could expect, the behaviors observed for walls values of 0 and 1 and sharply different from the other ones. The average time step is far more homogeneous over the graph, and the average time step are higher than those for the other rules. Let's remind that these are not frozen wall cells as it is the case for the other walls values, but acts like evolving cells from time step 2. We can notice there is no continuity in function of wall values around 0 and 1. Lastly, some values such as 0.5 and 1.5 makes appear a high time step probability for values with intermediate percent and low square size.

2.8.2 Behavior in dependence of cycle length

In this section, we try to discriminate behavior in function of cycle length, at fixed values. First and foremost, what can be noticed is that most cellular automata end with a cycle of length 1 or 2. Their distribution is very similar. Next, automaton with cycle length is 3, 4 and 6, and those with no detected cycle in 500 steps, are the most frequent. Here we present examples of highly represented cycles.

Conclusion to Section 2

We were able to determine to some behaviors influenced by parameters of the wall distribution. We try to evolve in a context as constrained as possible (with square walls) to avoid coping with a huge number of parameter, which could have been overwhelming in the visualization step. We could see some rules appearing for global cycle, such as a higher average time step before cycle for low percentage and high size of wall squares. Some of the data found were presented in this report, but most of the data are still to be exploited from all the other data.

3. 2D Cellular automata: Study of Cellular Automata with inhomogeneous neighborhood and non-classical rules

3.1 Problem

We report a series of numerical experiments carried out on 2D cellular automata. We studied the behavior of cellular automata with different activation rules and with different inhomogeneous neighborhood. Rules implemented are inspired from classical rules of Conway's Game of Life. Neighborhood is a random function which takes in account user-set features. This series of experiments is not an exhaustive study of the whole problem. It intended to draw conclusion in a voluntarily constrained framework to obtain readable results. The whole study is empirically tackled.

3.2 General Model

In this section, we present the global model adopted for the simulation of the cellular automata. Each experiment has its own parameters and programming specificities. We won't present any of these

specific sets. Instead, we introduce the general iteration process of the cellular automata. The Conway's Game of Life classical rules had been described in the Section 2.

3.2.1 Generalized rules implemented

The rules implemented are based on the classical Game of Life rule idea. Here, we suppose that NS(t) represents the sum of the neighbor's value of the considered cell at t. Our rules can be modeled as a vector of four integer values (a, b, c, d). With the same notations that previously, the rule can be rewritten as:

- for C(t-1)=1 : if **a < NS(t-1) < b** then C(t)=1, else C(t)=0
- for C(t-1)=0 : if **c < NS(t-1) < d** then C(t)=1, else C(t)=0

For instance, classical Game of Life rule corresponds to the vector (1, 4, 2, 4). Note that we should have a < b-1 in order to enable cells being maintained alive, and have c < d-1 in order to enable them recover from death.

3.2.2 Neighborhood

To build the neighborhood, we made an analogy with graph theory. The neighborhood of our model is inhomogeneous, directed, and constant over time.

Initially, we attribute two parameters to the neighborhood of the grid:

- the number of neighbor's n of each cell.
- the radius r of the neighborhood, which can be seen as the integer value of the neighborhood scope. If we consider the cell at coordinate (i,j), another cell (k,l) can be in its neighborhood if:

$r <= \max(|i-k|, |j-l|)$

Here is a figure illustrating the neighborhood

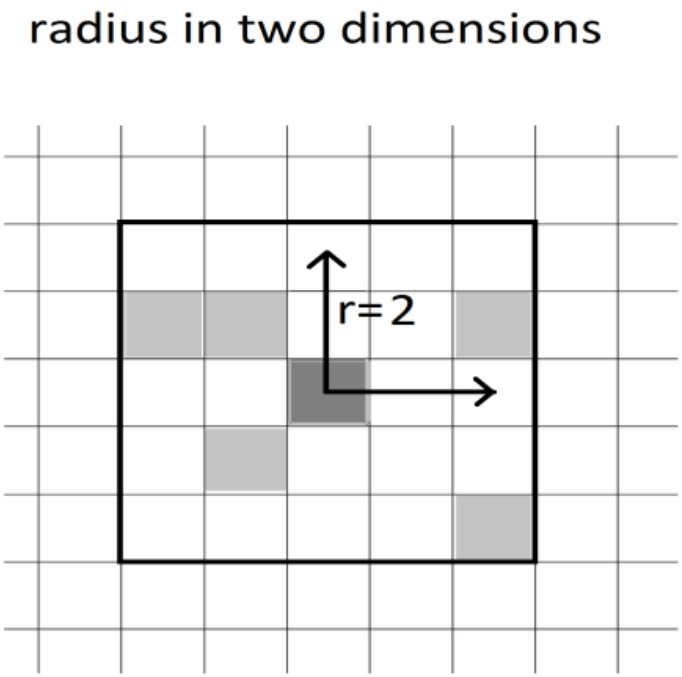


Fig. 6 Generalized neighbor and rules

Dark grey cell represents the cell we consider. Light grey cells are the neighbors of the cell. They are randomly chosen among the cells in the r-radius square.

At initialization step, each cell is given n neighbors among the $(2r+1)^2-1$ cells which subscribe the r radius condition. The choice of the cells is uniformly random. We also impose that cell cannot be their own neighbors. At updating time t-1→t, each cell computes **NS(t-1)** – the sum of its neighbor's value – and evolves in accordance with this value, and with its own state.

3.2.3 Border conditions

The border conditions are 2D periodic, in other words our 2D cellular automata is a manifold of a 3D torus. For a 2D cellular automaton composed of a NxN squared grid, considering C(i,j) the value of cell at t line i column j, we impose:

- C(N+m, k) = C(m, k) for k, m in {1,.., N}
- C(k, N+m) = C(k, m) for k, m in {1,.., N}
- C(1-m, k) = C(N-m, k) for k, m in {1,.., N}
- C(k, 1-m) = C(k, N-m) for k , m in {1,.., N}

Defining a C(i, j) with i>N , j>N, i<1 , j<1 is necessary for the definition of the neighborhood of border cells.

3.2.4 Initial distribution of life

At initial time step, every cell is given a fifty percents probability to be alive. Other cells are dead.

3.3 Numerical experiments

In this section, we present the series of experiments we conducted. The general aim is to study the overall evolution of the cellular automata. Hence, we used some global representation plot: evolution of the cellular automata over time, life rate of the cellular automata over time, average time steps to reach cyclic state in function of space.

For all the numerical experiments, we used Matlab software to simulate the cellular automata. The maximal duration – number of time steps simulated – of cellular automata was 5000. This is a convenient value in term of computation time. We used a grid of 30x30 cells.

Here, we present the average number of time steps required to reach a global cycle in function of the size of the grid. A global cycle is defined as a cycle in the temporal evolution of the whole grid. Here, this simulation indicates that we modeled cellular automata from size 3x3 to size 35x35, and we did this three times ("nbValues3" in the title of the graph). Then we averaged them for each size. For instance, the 30x30 grid reached a global cycle after 600 time steps in average. This preliminary experiment was useful to test whether the 30x30 size chosen was enough to enable the cellular automata to converge in a global cycle within less than 5000 time steps.

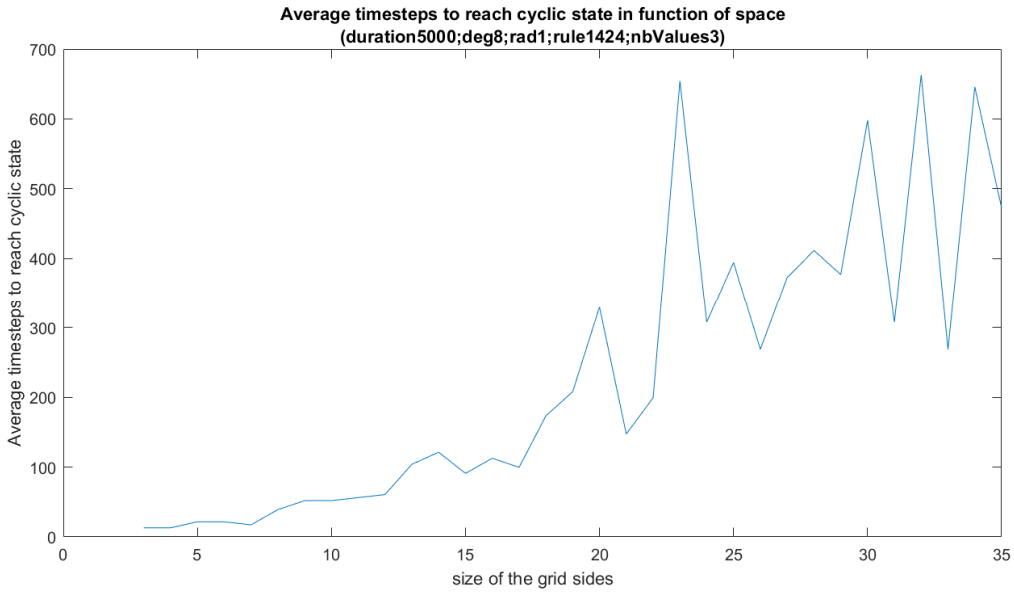


Fig. 6 Time of reaching cycles

3.3.1 Life rate over time in function of number of neighbors

In these first simulations, the aim was to detect particular comportments of the cellular automata in function of the number of neighbors we set. All the simulations consider the classical Conway's Game of Life rule – which we named (1, 4, 2, 4) in our model.

On each graph, we represented the percent of alive cells of the cellular automata over time. As we explained before, each graph represents different cellular automata: different initial distribution of life, and different neighborhood for each cells.

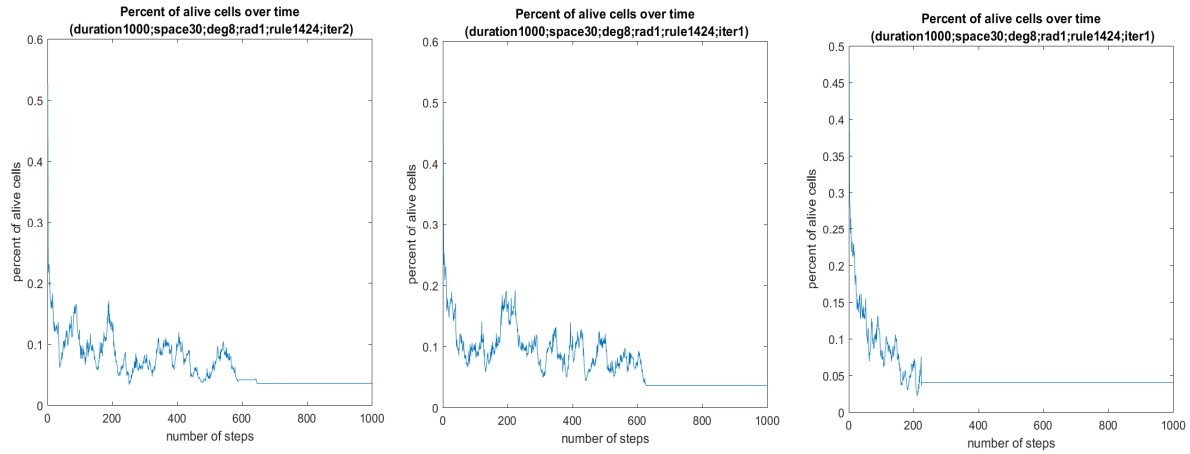


Fig. 7 Simulations for 8 neighbors (classical Game of Life).

As displayed on these nine different graphs, most cellular automata converge within a number of time steps inferior to 1000. What's more, when the number of neighbors decreases the number of time steps necessary to reach a global cyclic state decreases. Besides, the final rate of alive cells also seems to decrease with the number of neighbors.

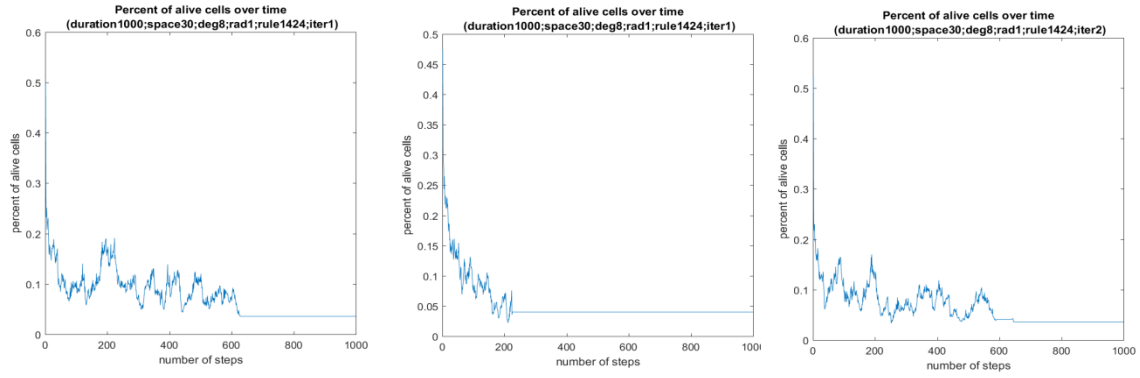
This last point can be must be due to the sum of the neighbors' states NS(t) whish drives the rule and which is lower when diminishing the number of neighbors. Hence, the condition 1<NS(t)<4 which enables a cell to remain alive and the condition 2<NS(t)<4 which enables a cell to get alive back is statistically harder to reach. Hence this fall of the life values over time.

3.3.2 Life rate over time in function of the rule

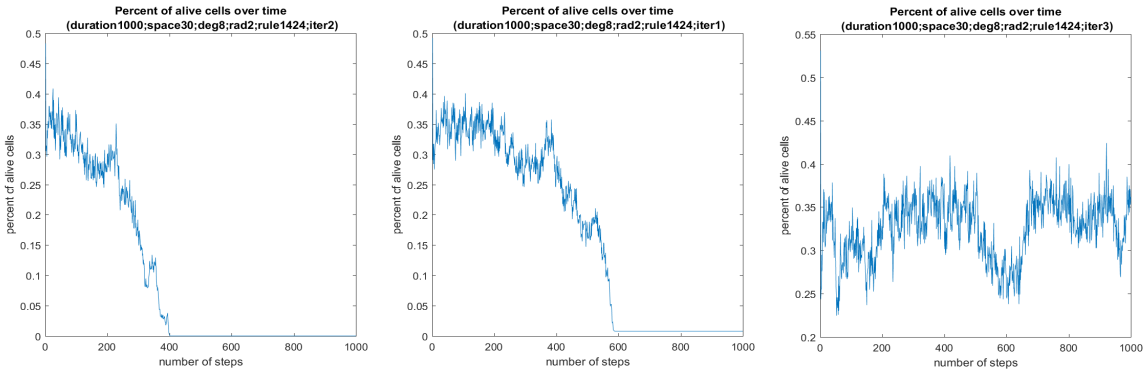
In the next simulations, the aim was to detect particular comportments of the cellular automata in function of the radius set. All the simulations consider the classical (1, 4, 2, 4) rule.

On each graph, we represented the percent of alive cells of the cellular automata over time.

Simulations with radius 1 (classical Game of Life):



Simulations with radius 2:



Simulations with radius 3 :

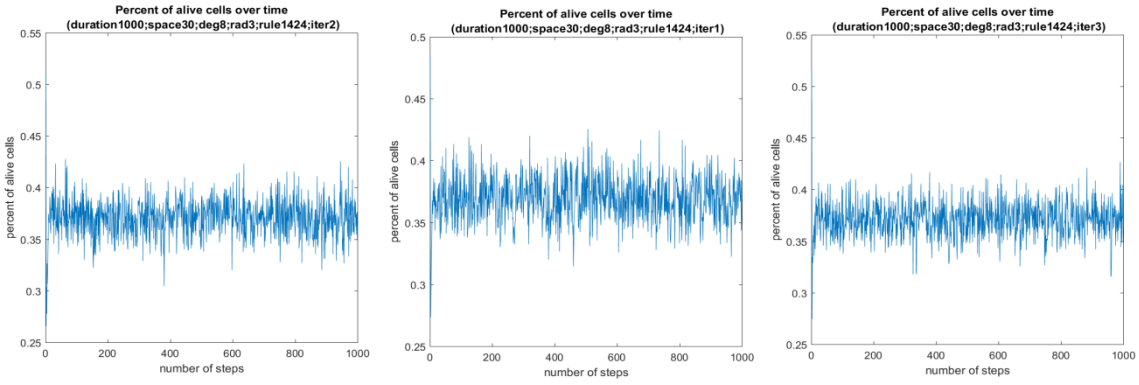


Fig. 8 Simulations with different size of neighbors

As displayed on these nine different graphs, when the radius increases the number of time steps necessary to reach a global cyclic state increases. Plus, the fast oscillations of the life rate increases as the radius grows. These oscillations of life seem to have an increasing amplitude and frequency. Besides, although the oscillations increase, the general trend of life over time seems more stable easier to forecast. We also notice that for a radius 2, the cellular automata converging in a cyclic state often converge in a global dead state. Unfortunately the maximum duration time steps does not enable us to draw the conclusion for radius=3.

3.3.3 Average time steps to reach cycle in function of number of neighbors

In order to draw more accurate figures on cellular automata – especially regarding the number of timestamps needed to reach a global cycle – we conducted other simulations. In the following simulation, we once again used the classical (1, 4, 2, 4) Conway's Game of Life Rule). The goal of the simulation was to study the average number of time steps before reaching a global cycle in function of the number of neighbors, going from 2 to 8. For each number of neighbors, we computed 41 different cellular automata and averaged their number of time steps required to reach a global cycle:

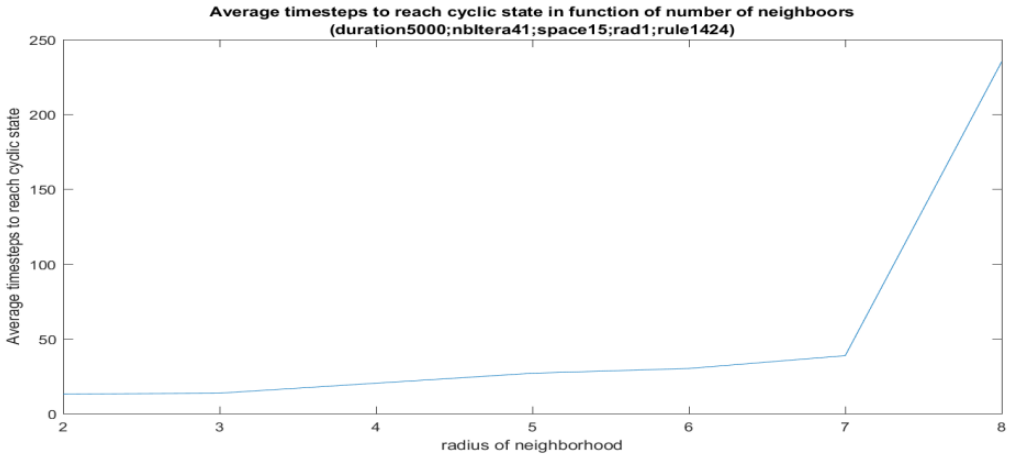


Fig. 9 Averag time for reaching cycles

The graph shows a very clear break for the critical value 8 of classical Game of Life rule. For value from 2 to 7, the average number of time steps looks like an increasing linear function of the number of neighbors. Note that it was worthless simulating the case with one neighbor since, following the Gale of Life rule, every cell would have died within the first simulated time step.

3.3.4 Average time steps to reach cycle in function of radius

Here, we represented the same type of graph, but changing the neighborhood scope, i.e. increasing the radius parameter.

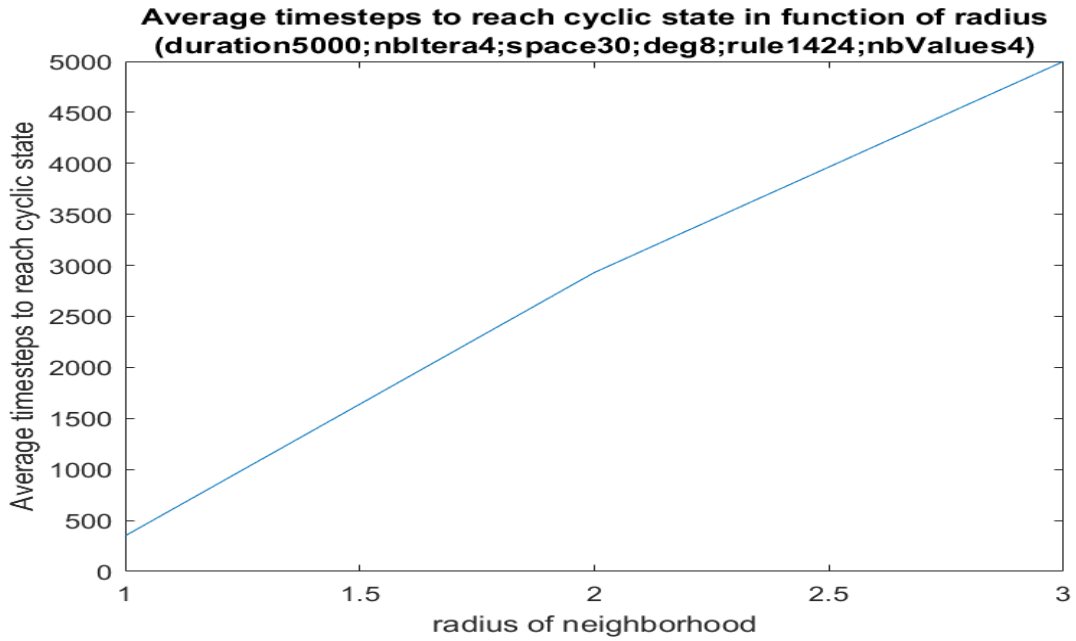


Fig.10 Average time for cycling in dependence on radius.

Here, the conclusion of the graph is limited by the maximum number of time steps we imposed on our cellular automata. Above a radius of 3, the average time steps overcome the 5000 time steps maximal duration. Yet, we have data about the radius 2 neighborhood's average time steps to reach global cycle, which is 6 times higher than the one for the radius 1 neighborhoods.

3.3.5 Determination of Game of Life-alike rule with different neighborhood

Some previous graph pointed out the limit of the study we conducted: the radius and the number of neighbors of each cell are closely linked to the classical Game of Life rule. Hence, changing one of these two parameters leads to sharply different behaviors. This emphasizes the link between neighborhood and rule in cellular automata behavior. This last experiment is a short empirical study aiming at determining some Game of Life-alike rules which lead to Game of Life-alike behaviors.

In order to determine such rules, we implemented algorithms based on random and mutation. Such algorithms are useful to determine "good" rule without examining each rule, due to the lack of computational power. Here, we do not have any objective function as it could be the case in genetic algorithm. Instead, we test the final life rate of the cellular automaton and if it belongs to the expected range, the rules are selected and then undergo some mutations. The conditions upon life attempt to find a life rate close to the 7% that the classical Conway's Game of Life converges to.

Here are the different steps of the algorithm:

Stage 0 : Fix a number of neighbors and a radius for the neighborhood.

Stage 1: Generate random (a,b,c,d) and select them if the cellular automaton state agrees with "loose" life conditions – life rate at time step 100 between 0.01 and 0.4.

Stage 2 : Make evolve randomly every (a,b,c,d) already selected at stage 1 and simulate the corresponding CA. Select the new rules if the cellular automata state agrees with "medium" life conditions – life rate at time step 200 between 0.01 and 0.4.

Stage 3 : Simulate CA with (a,b,c,d) already selected at stage 2 and select the rules if the CA agrees with tight life conditions – life rate at time step 1000 between 0.02 and 0.3.

We show the evolution of CA whose rule has been selected by the algorithm.

The cellular automata found with this rule enable a certain kind of life. The emergence of such life pattern is not trivial and most rules generated randomly do not enable finding such shapes. Most of the time they converge to a dead state.

This last experiment is an attempt to highlight the link between the neighborhood and the rules implemented in the evolution of life in such systems.

Conclusion to Section 3

Here we report the study of 2D cellular automata with inhomogeneous neighborhood and with new types of rules, inspired from classical Game of Life cellular automata. We built a model for the neighborhood, making analogies with network science through graph theory. We also defined a new model for the rules, and explained how we could transpose this model for Game of Life classical rule, and how we could derive new rules from this classical rule. Then, we simulated some cellular automata with classical Game of Life rule. We could highlight some behaviours of the life in the cellular automata. Eventually, we admit that instead of rewriting neighborhood of cells independantly of rule, which lead to significantly different cases from the classical Game of Life, we should attempt to discover new rules, more closely bound to the neighborhood adopted. They lead to some interesting self organizing pattern.

Some further studies could be carried out to extend the first graphs drawn, in particular those on which the maximal timestep was limiting the scope of the results. One might also consider other ways to discover the rule we shed light on, and thanks to more powerful computationnal means, try to investigate these rules in a more exchaustive way.

4. Study on 2D cellular automata: Multi-layer cellular automata, with classical Game of Life Rules

4.1 Problem

This report intends to study the behavior of two-layer cellular automata. Each layer of the cellular automata updates according to classical 2D Game of Life model. Some inter-layer links are introduced in the model. We studied the influence of such links upon the evolution of life in the cellular automaton.

4.2 General Model

In this section, we present the global model adopted for the simulation of the cellular automata. We make an analogy between network science lexicon and cellular automata one. In particular, an "edge" or "link" between two cells of cellular automata represents a relation of neighborhoods between these cells. In cellular automata lexicon, such cells (bound by an edge/link) are called "neighbors".

4.2.1 Multilayer model

Our model consists of two grids of size NxN. Each of these grids is a layer of the cellular automaton. These layers are connected by some inter-layer links. These inter-layer links are undirected edges. We only consider one type of inter-layer link: coupling edges. They are links connecting a cell A at a position (i,j) on the first grid, and a cell B with the same coordinate (i,j) on the other grid.

4.2.2 Updating rule

Here we remember classical Game of Life rule, which we applied to our model. At each time step, a cell can be in one of the two states 'dead' or 'alive'. Each cell evolves according to its own state and to its neighborhood's state – we will define its “neighborhood” later. Here is the rule to update a cell:

- an 'alive' cell at time t-1 remains alive at time t if 2 or 3 of its neighbors are also alive. Otherwise, it becomes 'dead' at time t.
- a 'dead' cell at time t-1 becomes alive at time t if 3 of its neighbors are alive. Otherwise, it remains 'dead' at time t.

In our model, a cell is represented by a Boolean value equal to 1 if the cell is alive and 0 if the cell is dead. Hence, the rule can be rewritten as a function of the sum of its neighbor’s states. Let's consider C(t) represents the state of one cell at time t, and NS(t) represents the neighborhood sum of states. We can rewrite the Game of Life rule as:

- in case C(t-1)=1 : if **1<NS(t-1)<4** then C(t)=1, else C(t)=0
- in case C(t-1)=0 : if **2<NS(t-1)<4** then C(t)=1, else C(t)=0

4.2.3 Neighborhood

Our model neighborhood consists of two types of neighbors:

- *intra-layer neighbors*. They represent the neighbors of the cell within the same layer. These neighbors follow the classical Moore neighborhood, i.e. they are the eight closest cells of the considered cell.
- *inter-layer neighbors*. We only consider inter-layer neighbors bound by coupling edges, i.e. cells with same coordinates but different layers.

Both type of neighbors - inter-layer and intra-layer - are considered equally regarding the updating rule. For a cell, the neighborhood sum at time t takes in accounts both intra-layer and inter-layer neighbors.

4.2.4 Border conditions

The border conditions of each layer are 2D-periodic, which is tantamount to saying that each 2D cellular automata layer is a manifold of a 3D torus.

We define the 2D-periodic condition. Every cellular automata layer composed of a NxN squared grid, considering C(i,j) the value of cell at t line i column j, we impose:

- C(N+m,k) = C(m,k) for k,m in {1,..,N}
- C(k,N+m) = C(k,m) for k,m in {1,..,N}
- C(1-m,k) = C(N-m,k) for k,m in {1,..,N}
- C(k,1-m) = C(k,N-m) for k ,m in {1,..,N}

Defining C(i,j) with i>N , j>N, i<1 , j<1 is necessary for the definition of the neighborhood of border cells.

4.2.5 Initial distribution of life

At initial time step and for each layer, every cell is given a fifty percents probability to be alive. Other cells are dead.

4.3 Numerical experiments and results

We realised all the following experiments with Matlab Software, using matrix as representation of the cells.

4.3.1 Parameters

For the numerical simulations, we adopted a two-layer cellular automata, with a size of 30x30. The cellular automata was encoded on a 30x30x2 Matrix. Each cell is assigned a boolean value in function of its state (1 for alive, 0 for dead). The last coordinate (of size 2) represents the layer of the cell. In all experiments, we simulated the cellular automata during 500 timesteps. We will consider that "final" state of the cellular automata is the state at the 500th step.

4.3.2 Life rate over time

In the next simulations, we intended to draw general a trend of the evolution of life – final life rate, time steps before reaching a global cycle – for different percents of inter-layer edges. First, we plotted the evolution of life rate – percentage of alive cells among all the grid's cells – over time for several values of p. The model is equivalent from one layer to another. Hence, to simplify the following graphs, we only displayed the rate of one of the two layers. On each plot, we displayed 10 different curves. They represent the life rates over time of 10 different cellular automata, changing slightly the initial inter-layer link probability p. On the following graph, we represented for example the life rates of 10 cellular automata with p between 0.005% and 1.000%.

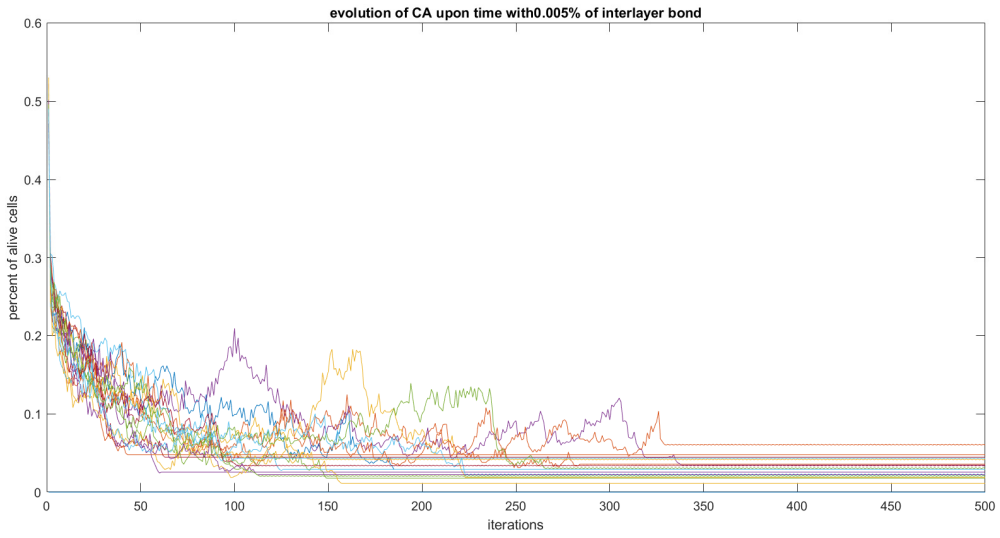


Fig. 11 Evolution of cells state

From analysis of such computations, it seems that p – the rate of inter-layer links – has an impact on the evolution of life. More accurately, it seems that:

- When p increases, the final rate of alive cells first decreases on average (until p=0.60%), before increasing.
- When p increases, the number of time steps required before the cellular automata enters a global cyclic state decreases.
- When p increases, the cellular automata tends to finish in a two-steps cycle, until p is high (around p=0.8%), where the cellular automata ends in a one-step cycle.

With the numerical experiments, we will try to confirm these predictions, and draw more precise laws.

4.3.3 Final life rate in function of interlayer link probability

In these numerical simulations, we computed the final life rate. We computed this final rate in function of the inter-layer link probability. For the simulations, p goes from 0.005% to 1.000% by steps of 0.005%. We have 200 different values. In this first graph we represent each couple (final life rate; inter-layer link probability).

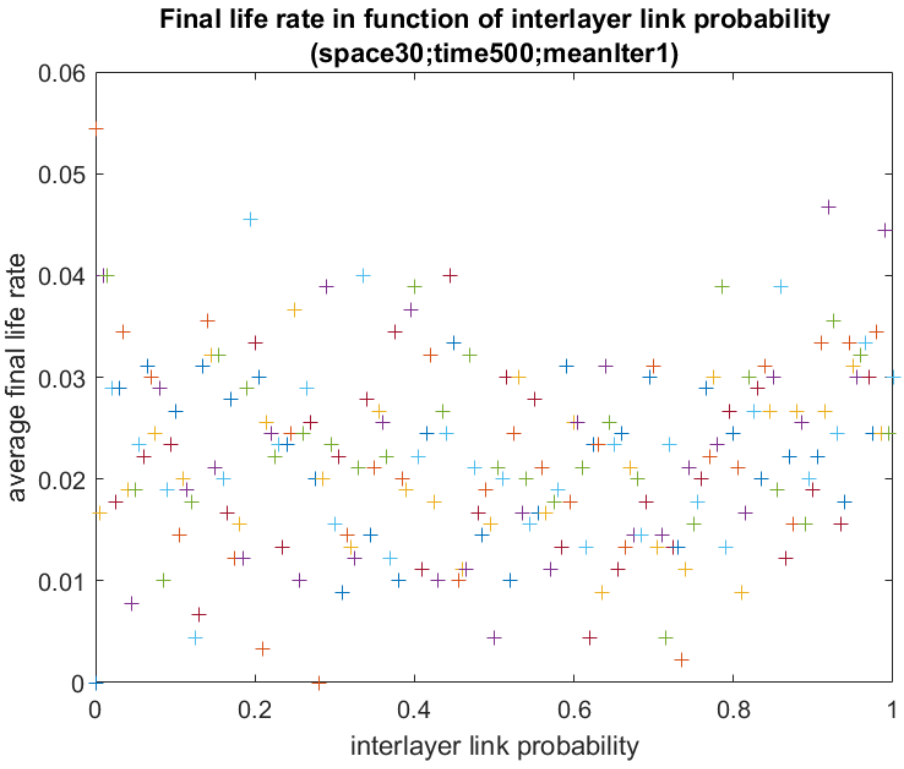


Fig. 12 Dependence on the links between layers

In this second graphs, we computed the bary-center of the previously computed couples. There are 20 bary-centers, each averaging 10 closest couples (final life rate; inter-layer link probability). Statistically, there is first a decrease of the final rate, going from around 0.030% for low p to 0.020 for p around 0.6%, followed by an increase which reaches back 0.030% for high values of p.

4.3.4 Time steps to reach cycle

We consider that the cellular automata reaches a *(global) cycle* when it is in a state it has already been in since the beginning of the simulation. A state of the cellular automata represents the state of every cell on both layers. Here, we implemented a basic algorithm which compares the state of the cellular automata at time t with all the previous states it was in. In these numerical simulations, we compute the time steps when the cellular automata are detected to be in a cycle. We computed these time steps in function of the inter-layer link probability. For the experiments, the p probability varies from 0.005% to 1.000% by 0.005%. Hence, we had 200 different values. We also had computed the bary-center of the previously computed coupled. There are 20 bary-centers, each averaging 10 closest couples (time steps to reach cycle; inter-layer link probability). From the computations it is clear that there is a relationship between the inter-layer link probability and the number of time steps to reach a cycle.

As the density of couples (time steps to reach cycle; interlayer) is low for $p<0.2$, we also simulated another 200 values for p between 0.001% and 0.200%. As we can see the number of time steps required to reach cycle is higher for low p. Hence, exceptionally, this simulation computed up to 1000 time steps for each cellular automaton. The graphs of computation to confirm the trend highlighted before.

4.3.5 Cycle length in function of inter-layer link probability

Here, we plotted the length of the cycle (within a cycle, minimum number of timesteps before reaching twice the same state). Some cellular automata did not reach a cycle within the 500 timesteps. They are given the length cycle 0 on the graphs. We simulated the cellular automata for the same values of p as previously.

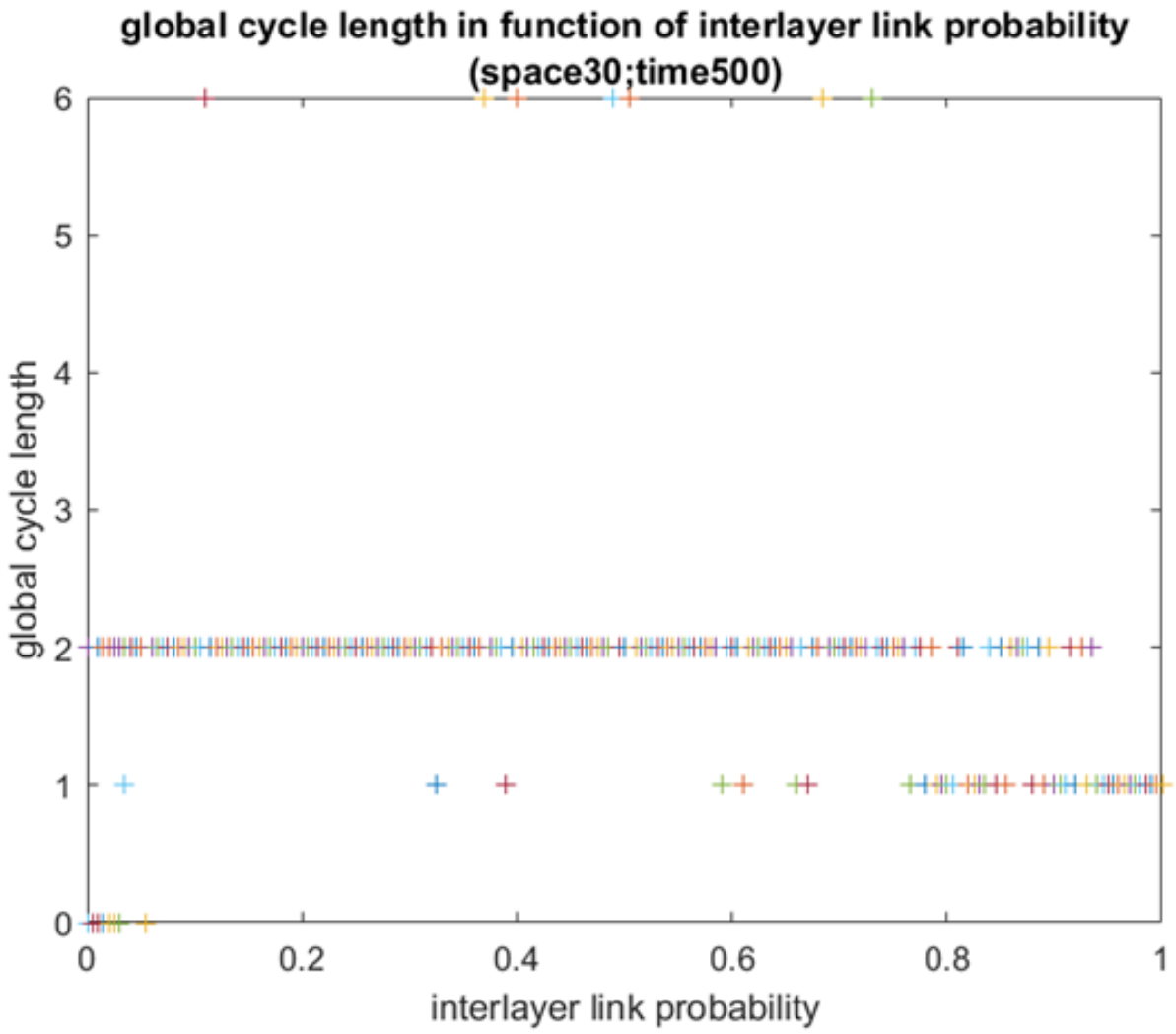


Fig. 13 The cycle length dependence of interlayer coupling.

The graphs enable us to draw a statistical trend of the cellular automata, which are highly prone to:

- end in a two-state global cycle for p<0.80%.
- end in a one-state or two-state cycle for 0.8%< p <0.93%.
- end in a one-state cycle for p>0.93%.

Conclusions for Section 4

This short study on multi-layer cellular automata give some general trends of life evolution in a constrained framework of cellular automata. We manage to draw statistical laws bewteen the inter-layer link probability on the one hand, and the number of timesteps to reach cycle, the final life rate, and the

cycle length on the other hand. We decided to restrict the study to two-layer cellular automata with exclusively coupling edges. This choice was imposed by the important number of parameters already existing, the presence of random conditions which might make the trend harder to draw. Besides, adding another layer would add the number of neighboors, which could affect the influence of classical game of life rule. It would also require to reflect on inter-layer linking. With some linking conditions, each layer might be no longer symmetric (and equivalent) from the system. Adding new layers, or introducing other types of inter-layer links might motivate some other studies.

General conclusions.

Thus in given paper we described the results of investigations by computer experiments with different kinds of the classical cellular automata generalization. Received results may be useful for understanding the behavior of cellular automata which may be useful for searching the architecture for cellular computing. Remark that deep further generalizations should be considered. The first is related with the accounting of networks structure especially the 'small world' property. The second prospect for cellular computing is accounting of strong anticipatory property (see [1, 16, 18]).

Acknowledgement

"The paper is published with partial support by the ITHEA ISS and the ADUIS".

References

1. *Makarenko A*. Cellular Automata with anticipation: Some new Research Problems // Int. Journal of Computing Anticipatory Systems (Belgium). — 2008. — **20**. P. 230 – 242.
2. *Von Neumann J.* Theory of Self-reproducing Automata. University of Illinois Press Urbana, 1966. Burks A. (ed.)
3. *Wolfram S*. A new kind of science, Champaign, IL, Wolfram Media, 2002.
4. *Illachinski A*. Cellular Automata. A Discrete Universe. World Scientific Publishing, Singapoure. — 2001.
5. *Toffoli T., Margolus N.* Cellular Automata Machines, MIT Press, 1987.

6. *Chua L*. A nonlinear Dynamic Perspective of Wolfram's New Kind of Science, World Science, 2011.
7. *Adamatzki A.* Game of Life Cellular Automata, Springer-Verlag, 2010.
8. https://en.wikipedia.org/wiki/Cellular_automaton
9. *Chopard B., Droz M.* Cellular Automata Modeling of Physical Systems. Cambr. Univ. Press, 1998.
10. *Bandini S., Mauri G.* Multilayered cellular automata. Theoret Comput. Sci. volume 217, Issue 1. 1997. Pp.99-113.
11. *Goldengorin B., Makarenko A., Smilianec N*. Some applications and prospects of cellular automata in traffic problems Cellular Automata, Proceed. Int. Conf. ACRI'06, LNCS 4173, Springer, 2006. — P. 532–537
12. *Makarenko A., Krushinski D., Musienko A., Goldengorin B*. Towards Cellular Automata Football Models with Mentality Accounting. LNCS 6350. Springer – Verlag, 2010. — P. 149 – 153.
13. *Wonghhanasu Sartra* Cellular automata for pattern recognition. In: Emerging Applications of Cellular Automata. Ed. Alejandro Saliedo. InTech, Croatia, 2013. Pp. 53-68.
14. *Tsoutsouras V., Siracoulis G.Ch., Parlos G.P., Iliopoulos A.S.* Simulation of healthy and epilepiform brain activity using cellular automata. Int. J. of Bifurcation and Chaos. 2012. Volume 22, Issue 09. Pp. 1250229-52.
15. *Vlassopoulos N., Fates N., Berry H., Girau B.* An FPGA design for the stochastic Greenberg-Hastings cellular automata. Int. Conf. on High Performance Computing and Simulation – HPCS2010, Caen, France, 2010. Pp. 565-574.
16. *Didkivskiy A.M Makarenko O.S., Osaulenko V.M., Redchuk B.V.* Cellular automata concept – from micro to global levels. Proc. Of 1st WolfgangKummer Int. School on Astroparticle Quantum Gravity and Quantum Information, Aug. 3-16, 2015. Ed. S.S. Moscaliuk. Kyiv,: TIMPANI, 2016. PP. 75-84. (in Ukrainian)
17. *T'Hooft* The Cellular Autmation Interpretation of Quantum Mechanics. ArXiv: [qiant-ph]? 1405.1548. v3, 21 Dec. 2015. 259 p.
18. *Makarenko A.* Cellular automata methods in quantum Physics. Proc. Of 1st Wolfgang Kummer Int. School on Astroparticle Quantum Gravity and Quantum Information, Aug. 3-16, 2015. Ed. S.S. Moscaliuk. Kyiv,: TIMP.ANI, 2016. PP. 97-110. (in Ukrainian)
19. *Cooper S.* Barry Computability Theory. Chapman Hall/CRC (2003)

20. *Sipper M.* Co-evolving non-uniform cellular automata to perform computation. Physica D, Vol.92, Issue 3-4, 1996. Pp. 193-208.

21. *Fates N.* A Guided Tour of Asynchronous Cellular Automata. Cellular Automata and Discrete Complex Systems. LNCS 8155. J. Kari, M. Kutrib, Malcher A. (eds.), Springer, 2013. Pp. 15-30.

22. *Bandini S., Manconi S., Simone C.* Heterogeneous agents situated in heterogeneous space. Applied Artificial Intelligence. 2001. Volume 16, Issue 9-10. Pp. 831-852.

23. *Fates N.* Directed percolation phenomena in asynchronous elementary cellular automata. LNCS 4173, Springer-Verlag, 2006. Pp. 667-675.

24. https://www.nobelprize.org/nobel_prizes/physics/laureates/2016/press.html

Authors' Information

Lois Facchetti - *Mines Nancy – ADTEM, Nancy, France*

Alexander Makarenko - ***Institute for Applied System Analysis at National Tech. University of Ukraine (KPI) Kyiv, Ukraine*

e-mail: makalex51@gmail.com , makalex@i.com.ua

HYBRID MODULAR MODEL FOR TIME SERIES FORECASTING BASED ON NEURO-FUZZY NETWORK AND FUZZY COGNITIVE MAPS

Sergey Yarushev, Alexey Averkin

Abstract*: In this paper, we consider a hybrid approach to forecasting time series using neuron-fuzzy prediction models and Fuzzy Cognitive Maps. Main idea of proposed approach is hybridization two different ways for time series forecasting. We can make quantitative and qualitative forecast. In addition, we describe the different approaches to learning and optimization of the network, such as the methods of particle swarm, evolutionary methods, as well as variants of the hybridization of these methods. Also, in a comparison of the results of the prediction for example, one of the indicators of the State Program of Development of Science and Technology is forecast schedule.*

Keywords*: forecasting, fuzzy cognitive maps, time series, hybrid-forecasting models.*

ITHEA Keywords*: I. Computing Methodologies: I.2 Artificial Intelligence: I.2.1 Applications and Expert Systems*

Introduction

Nowadays modelling and forecasting time series are among the most active areas of research. For example, depending on the historical data, situation on sales market, changes in prices for shares of population growth and banks deposits are forecast. Forecasting time series affects the lives of people around the world, so it has great practical value and perspectives of research in all areas of the modern society, which is also an important area in the field of computer application.

The solution of problems of identification of dynamic objects should be used in a variety of fields: it can simplify temperature controllers, or complex management and forecasting. It can also solve the forecasting problem, along with a number of different methods, for example, statistical analysis, neural networks [Haykin, 1994]. Identification of the object may be difficult if the exact structure of the model of

the object is unknown, some of the parameters of the object change due to obscure principles, or the exact number of parameters of the object is unknown. In such cases, the hybrid neural network can be used for identification of dynamic objects. There are many types of neural networks that are used for

identification of dynamic objects. Despite the large number of neural network methods for identification of dynamic objects, most of these algorithms have some limits, or do not provide the required accuracy.

Among all kinds of neural networks, architectures that can be used for identification of dynamic objects allocated a class of neural networks based on self-organizing maps of Kohonen with hybrid architecture. Hybrid neural networks of this type will get special attention in this article because they are becoming more widespread and successful applications for solving various problems of recognition [Efremova, 2012], identification [Trofimov, 2010], and forecasting. We will also consider a number of biomorphic neural networks applicable for solving identification problems and management.

In the development and future changes in time series, there is reflexivity between events, their participants and the actual prognosized process (time series), between the researcher and the process being studied [Lefevr, 1965]. The theory of reflexivity in the economic world suggests that the situation that has arisen affects the behavior of the participants in the process, and their thinking and behavior act on the development of the situation to which they are participants [Soros, 2003]. It is clear that using only a tool for forecasting time series, it would not be as powerful as any, it is impossible to reflect and take into account the situation and events affecting the process under study, since a neural network is allowed to work with historical data. The practical way out of the situation is to develop such methods that could operate both with a cause-effect relationship between events and the projected process, as well as with the numerical values of the time series, its historical data. Therefore, it is expedient to develop a hybrid forecasting system capable of operating both qualitative data and quantitative ones.

In this paper, we propose a new hybrid time-series forecasting model based on fuzzy relational cognitive maps and a hybrid neural-fuzzy network with regression analysis.

Modular Neural Networks

The core of the modular neural networks is based on the principle of decomposition of com-plex tasks into simpler ones. Separate modules make simple tasks. More simple subtasks are then carried through a series of special models. Each local model performs its own version of the prob-lem according to its characteristics. The decision of the integrated object is achieved by combin-ing the individual results of specialized local computer systems in a dependent task. The expan-sion of the overall problem into simpler subtasks can be either soft or hard-unit subdivision. In the first case, two or more subtasks of local computer systems can simultaneously assigned while in the latter case, only one local computing model is responsible for each of the tasks crushed.

Each modular system has a number of special modules that are working in small main tasks. Each module has the following characteristics:

— The domain modules are specific and have specialized computational architectures to recognize and respond to certain subsets of the overall task;

— Each module is typically independent of other modules in its functioning and does not influence or become influenced by other modules;

— The modules generally have a simpler architecture as compared to the system as a whole. Thus, a module can respond to given input faster than a complex monolithic system;

— The responses of the individual modules are simple and have to combine by some integrating mechanism in order to generate the complex overall system response.

The best example of modular system is human visual system. In this system, different modules are responsible for special tasks, like a motion detection, color recognition and shape. The central nervous system, upon receiving responses of the individual modules, develops a complete realization of the object which was processed by the visual system.

Review of Hybrid ANFIS Models for Time Series Forecasting

At the moment, there are many different learning algorithms ANFIS networks, each of them has its own advantages. Consider some of the studies that examined various hybrid-learning methods.

Chinese scientists, [Wang, 2015] presented its own model of forecasting of financial flows in the banking sector using a modified algorithm swarm optimization, which is called APAPSO (Adaptive Population Activity PSO). In view of the fact that the use of traditional methods do not give stable prediction results, researchers have proposed hybrid learning algorithm based APAPSO algorithm in combination with the method of least squares. In comparative experiments, the algorithm developed compared with the standard backpropagation technique in combination with the method of least squares (LMS), and with the traditional method swarm optimization-LMS. Results showed an increase in speed optimization, in comparison with conventional algorithms, as well as increase the accuracy of the prediction.

Indian scientists [Gunasekaran, 2011] offered a hybrid-forecasting model based on the integration of ANFIS and immune algorithm for the prediction of the Indian stock market. To create an effective model of prediction, the researchers decided to use an artificial immune algorithm to adjust the parameters of fuzzy system output functions. The data daily close of trading on the National Stock Exchange of India (NSE) were used as input data for system testing, as well as well-known technical indicators. The output

is a forecast of the future value of NSE index. The experimental results were compared with other models on the basis of soft computing and actual data from the auction. As a result, the experimental results have shown that the prediction model proposed gave much more accurate prediction results, compared to conventional models.

Artificial neural network (ANN) is a very good approximation method, which has the characteristics of adaptability and self-study [Dong, 2006]. However, using ANNs easily fall into a local minimum. By combining with fuzzy inference system has been proposed a new kind of non-linear prediction method, namely, adaptive neural fuzzy inference system (ANFIS) [Catalo, 2011]. This method can be used as the fuzzy rules and a neural network structure for implementing adaptive learning, so the prediction accuracy is higher than the one of the artificial neural network. In order to further improve the accuracy of predicting adaptive ANFIS system, you can use a variety of teaching methods, for example, the PSO algorithm (particle swarm optimization) or the method of particle swarm is used to optimize the structure of the network parameters. For example, a new hybrid approach combining particle swarm and ANFIS network is used for short-term forecasting of wind power in Portugal, as a result, it is possible to achieve the required accuracy of prediction using the proposed approach [Pousinho, 2011]. The radial basis function neural network (RBFNN) with non-linear evolutionary method swarm cha-particles, (NTVE-PSO) is proposed, and the simulation results show that the proposed NTVE-PSORBFNN has higher prediction accuracy and computational efficiency for predicting electricity consumption [Meng, 2012]. Improved PSO based on artificial neural network (ANN) was proposed by the researchers, results show that the proposed SAPSO based on ANN has a better ability to escape from local optimum and is more efficient than conventional PSO based on ANN [Cai, 2007]. Algorithm training based on a hybrid method of optimization of particle swarm (PSO) and the evolutionary algorithm (EA) for the prediction of 100 missing values of VRE-alternating series of 5000 data points that show the experimental D results that PSO-EA algorithm also proven effective in study [Wang, 2012].

Modular Hybrid System for Time Series Forecasting with ANFIS and Fuzzy Cognitive Maps

The developed forecasting system is based on a modular architecture that betrays the system additional stability when even if one of the modules crashed the remaining modules continue to perform their work.

The system itself has three main modules responsible for the prediction task. A hybrid neural-fuzzy network performs the forecast of a time series based on numerical indicators and gives us a so-called quantitative prediction, the results of which pass through a verification system (estimates of the adequacy of the forecast), if the prognosis corresponds to the required accuracy, then it is passed on to the next module. In parallel with the neural-fuzzy network, a module with a fuzzy cognitive map

operates, which receives data on the event-related effects on the time series, and constructs a cognitive map that takes into account all factors of influence on a specific predicted indicator. At the output, the cognitive map gives us a forecast with the probability of its implementation, that is, with the consonance of the factor that tells us whether the forecast will be fulfilled or not. Further, all data received from these modules is fed to the third module, which operates on the basis of the neural network, which aggregates the information obtained from the previous modules and outputs the final prognosis. In Figure 1 is a diagram of the forecasting system.

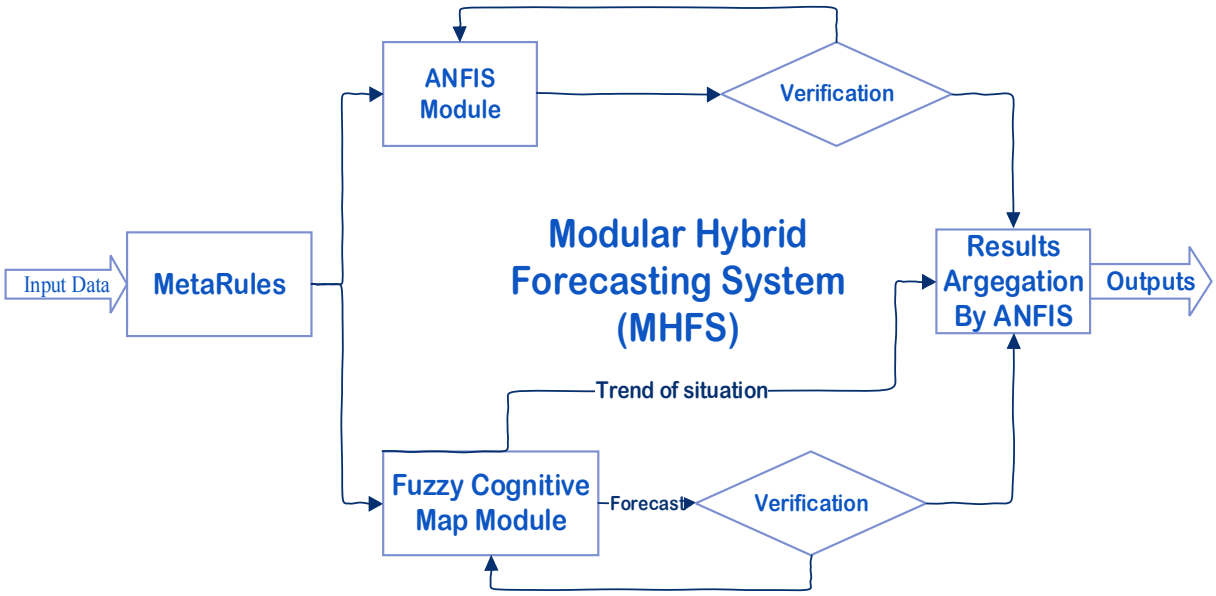


Figure 1. Modular Hybris Forecasting System

Further, we will dwell in more detail on forecasting based on fuzzy cognitive maps and their training, since a rather large number of studies are devoted to neural-fuzzy networks [Yarushev, 2016], with a slightly different situation with cognitive maps.

Fuzzy Cognitive Maps in Time Series Forecasting Area

The time series is governed by two main forces - time and events that affect the change over time of the values of the time series. Most of these events are characterized by some uncertainty. Each value of the time series can be associated with a fuzzy variable with some membership function. In this connection, the most interesting for our research are methods based on the theory of fuzzy sets. Lotfi Zadeh in 1965 introduced the concept of fuzzy set, due to which it is possible to describe qualitative, fuzzy concepts

and knowledge about the surrounding world, and then to operate them to obtain new information [Zadeh, 1976]. The application of this concept allows us to formalize linguistic information for constructing mathematical models [Rotshtein, 1997]. The notion of a fuzzy set is based on the proposition that the elements making up a given fuzzy set, as well as possessing common properties, can possess it and, consequently, belong to a given set in varying degrees. In this case, statements like "such and such an element belongs to a given set" lose their meaning, since it is still necessary to indicate the degree of belonging to a given set and its properties [Averkin, 1986].

To be able to operate with events that affect the time series, and events can be quite a lot and everyone can be related to each other, it makes sense to use fuzzy cognitive maps. They allow you to build a causal relationship between events and build a qualitative forecast of the development of the event, based on the strength of the influence of one event on another.

The cognitive map itself is an oriented graph, in which the vertices are the factors of the situation, and the weighted arcs are cause-effect relations, the weight of which reflects the force of the influence of the factors of the situation. Directional arcs of the graph are assigned the sign "+" or "-", i.e. they can be positive or negative. A positive relationship means that an increase in the value of the factor-cause leads to an increase in the value of the factor-effect, and a negative arc means that an increase in the value of the factor-cause leads to a decrease in the value of the factor-effect.

The tasks solved with the help of cognitive maps are to find and evaluate the influence of the factors of the situation, and to obtain, on the basis of the calculated influences, the forecasts of the development of the situation.

At present, for computation of the influences and forecasts of the development of the situation, fuzzy cognitive maps proposed by [Kosco, 1986] are widely used. In fuzzy cognitive maps, the force of influence between factors is given by means of linguistic meanings chosen from an ordered set of possible influence forces, and the values of the factors, their increments are also given in a linguistic form, and are chosen from the ordered sets of possible Values of the factor and its possible increments - scales of factors and incremental scales.

To construct a cognitive map that reflects the dynamic properties of the observed situation, it is necessary to determine the scales of the values of the factors and their increments.

To construct the scale of the factor, a lot of linguistic values of the factor are determined and structured. In determining the linguistic values, the absolute values of the factor are used, and not its evaluation of the type "large", "medium", "small". For example, the linguistic meaning of temperature can be the following: "so hot that you can barely put a hand on it" or the meaning "so cold that the hand

immediately freezes," and not just "Hot" or "Cold." With this definition of the linguistic values of the situation factors, an objective standard of its value is set - a reference point. Setting an objective standard of the value of a factor facilitates the work of experts in determining the strength of the influence of factors and reduces expert errors.

The prediction problem is reduced to the matrix-matrix composition of the matrix of weights and the vector of initial increments of characteristics.

This algorithm works for positively defined matrices, while in our case the elements of the adjacency matrix and increment vectors can take negative and positive values.

Learning Algorithm for Fuzzy Cognitive Map

Suppose that we have a set of 3N historical data lines (hereinafter - training material) about the status of concepts in the system. From the point of view of the problem of forecasting based on increments of concepts (see "Method of obtaining a forecast"), increments of concepts from i-th iteration to (i + 1) iterations will constitute the initial increment vector. In this case, the fuzzy cognitive map should show that with such an initial increment vector, the values of the concepts will change in such a way that the resulting increments will lead to values on the (i + 2) iteration.

Let $A_i(t)$ be the value of the concept e_i at time t. Based on the specification of the learning material given above, we will consider triples of rows $A_i(t)$, $A_i(t+1)$, $A_i(t+2)$.

Define $x_i = \frac{A_i(t+1)-A_i(t)}{A_i(t)}$, $y_i = \frac{A_i(t+2)-A_i(t)}{A_i(t)}$. Here, x are the initial increment vectors, and y is the resultant increment vectors.

Let $o_i(t)$ be the increment e_i, obtained as a result of the prediction on the initial vector $x(t)$.

The learning task is to minimize the error of the fuzzy cognitive map, but with the values of x, y, o introduced in this paragraph.

To solve the learning problem, a genetic algorithm is proposed. As a chromosome, a one-dimensional array of values is allocated, into which a two-dimensional array of weights of the fuzzy cognitive map is decomposed. Each value in this array is called a gene. Let's define the basic steps of the algorithm:

For all non-zero values of the weights of the initial map, a new non-zero weight value is defined, given by a small random number (the sign is not important). The initial non-zero values of the weights are

determined by the expert (a non-zero value can be any, its only purpose is an indication that, according to the expert, there is a causal relationship between the two selected concepts).

Item 1 repeats PopulationSize times. Thus, the initial population of random solutions is formed.

The fitness function is defined for each chromosome (see below for the form of fitness function).

The pool of parents is determined by the method of "roulette".

In the pool of parents, "elite individuals" are added. The elite individuals in genetic algorithms are individuals who have shown the best value of fitness function on the last few generations (one individual per generation).

There is a crossing of chromosomes that fall into the parents' pool. The crossing of chromosomes A and B occurs as follows. The crossing boundary of l is randomly determined. Let A_{l+} be the part of chromosome A, consisting of genes located from l, and A_{l-}- part of the chromosome, located up to l. Then the result of crossing will be two chromosomes $A_{l-}B_{l+}$ and $B_{l-}A_{l+}$. The probability of crossing is determined in advance. If crosses do not occur, both parental chromosomes change without change into a population of offspring.

From the descendants obtained in step 6, a new population is formed (its size is exactly the same as the size of the population in the previous step of the algorithm).

There are mutations in the population of descendants. When mutating, a random gene is selected and replaced with a new random value. The probability of a mutation is determined in advance. If the mutation does not occur, the chromosome passes to the next iteration of the algorithm unchanged.

The following generation parameters are determined: an elite specimen (an individual with the best fitness value) to preserve its gene pool; the average fitness of the population (only relevant for evaluating the convergence of the algorithm); the value of fitness of an elite individual.

If the fitness value of an elite specimen is greater than a predetermined value of maximum fitness, the algorithm stops, and the selected chromosome is decomposed into the adjacency matrix of the fuzzy cognitive map (the training is considered complete). Otherwise, go to step 3.

The concept of elite individuals was introduced into the algorithm to accelerate the convergence of the algorithm. The number of elite individuals is taken equal to 60, while the size of the population is 100 (thus, at each step after the 60th generation only 40 chromosomes from the current population have chances of crossing - the rest is filled by an elite gene pool inherited from previous populations) .

The maximum fitness value is defined as 0.99. The results of the training are rounded to the nearest hundredths.

The probability of crossing is defined as 0.9, and the probability of mutation is 0.5. Such a high probability of a mutation (usually uncharacteristic for genetic algorithms) is justified in this case, since mutations introduce genetic diversity into the population. At the same time, since an elite gene pool is used, there is no risk of irretrievably "losing" useful genes from previous generations.

Conclusion

In this paper, we presented a modular time series prediction system, which is based on a neural-fuzzy network and fuzzy cognitive maps. Such a prediction system allows you to include all the factors that influence the development of the situation, this is the actual numerical time series that is predicted on a fuzzy neural network and events that directly affect the future development of the time series. Developed is a genetic algorithm for learning a cognitive map that allows you to speed up the process of developing and adjusting the links of fuzzy cognitive maps.

Acknowledgements

This work was supported by the Russian Foundation for Basic Research (Grant No. 17-07-01558).

Bibliography

[Averkin, 1986] Аверкин А.Н., Батыршин И.З., Блишун А.Ф., Силов В.Б., Тарасов В.Б. Нечет-кие множества в моделях управления и искусственного интеллекта / под ред. Д.А.Поспелова. - М.: Наука, 1986. - 312 с.

[Cai, 2007] Cai, X.; Zhang, N.; Venayagamoorthy, G.K.; Wunsch, D.C. Time series prediction with recurrent neural networks trained by a hybrid PSO-EA algorithm. Neurocomputing 2007, 70, 2342–2353.

[Catalo, 2011] Catalao, J.P.S.; Pousinho, H.M.I.; Mendes, V.M.F. Hybrid wavelet-PSO-ANFIS approach for short-term electricity prices forecasting. IEEE Trans. Power Syst. 2011, 26, 137–144.

[Dong, 2006] Dong, Z. Study on the time-series modeling of China’s per capita GDP. Contemp. Manag. 2006, 11, 15.

[Efremova, 2012] Efremova, N., Asakura N., Inui T.. Natural object recognition with the view-invariant neural network. In: 5th International Conference of Cognitive Science, 2012, pp.802-804.

[Gunasekaran, 2011] Gunasekaran, M., and K. S. Ramaswami. "A fusion model integrating ANFIS and artificial immune algorithm for forecasting Indian stock market." Journal of Applied Sciences 11.16 2011: 3028-3033.

[Haykin, 1994] Haykin S. Neural networks: a comprehensive foundation. New York: Macmillan, 1994.

[Kosco, 1986] Kosko B. Fuzzy cognitive maps //International journal of man-machine studies. – 1986. – Т. 24. – №. 1. – С. 65-75.

[Lefevr, 1965] Лефевр В. А. Исходные идеи логики рефлексивных игр //Материалы конфе-ренции «Проблемы исследования систем и структур». М.: Издание АН СССР. – 1965.

[Meng, 2012] Meng, Y.-B.; Zou, J.-H.; Gan, X.-S.; Zhao, L. Research on WNN aerodynamic modeling from flight data based on improved PSO algorithm. Neurocomputing 2012, 83, 212–221.

[Pousinho, 2011] Pousinho, H.M.I.; Mendes, V.M.F.; Catalão, J.P.S. A hybrid PSO-ANFIS approach for short-term wind power prediction in Portugal. Energy Convers. Manag. 2011, 52, 397–402.

[Rotshtein, 1997] Ротштейн А.П., Штовба С.Д. Нечеткая надежность алгоритмических про-цессов. - Винница: Континент-Прим, 1997. - 142 с.

[Soros, 2003] Soros G. The alchemy of finance. – John Wiley & Sons, 2003.

[Trofimov, 2010] Trofimov, A., Povidalo I. And Chernetsov S. Usage of the self-learning neural networks for the blood glucose level of patients with diabetes mellitus type 1 identification. Science and education, 2010, vol. 5. Availible at: http://technomag.edu.ru/doc/142908.html

[Wang, 2012] Wang, S.; Wu, L. An improved PSO for bankruptcy prediction. Adv. Comput. Math. Appl. 2012, 1, 1–6.

[Wang, 2015] Wang, J. S., & Ning, C. X. (2015). ANFIS Based Time Series Prediction Method of Bank Cash Flow Optimized by Adaptive Population Activity PSO Algorithm.Information, 6(3), 300-313.

[Yarushev, 2016] Yarushev S. A., Averkin A. N. Review of studies on time series forecasting based on hybrid methods, neural networks and multiple regression //Программные продукты и системы. – 2016. – №. 1 (113).

[Zadeh, 1976] Беллман Р., Заде Л. Принятие решений в расплывчатых условиях. В кн.: Во-просы анализа и процедуры принятия решений. - М.: Мир, 1976. - С. 172-215.

Authors' Information

Sergey Yarushev *– is Postgraduate Student of Dubna State University, Dubna, Russia; e mail: sergey.yarushev@icloud.com*

Major Fields of Scientific Research: time series forecasting, neural networks, fuzzy systems, big data, deep learning, big data, artificial intelligence.

Alexey Averkin *– is Senior Researcher in Dorodnicyn Computing Centre, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences;, Moscow, Russia; e-mail: , averkin2003@inbox.ru*

Major Fields of Scientific Research: data mining, big data, neural networks, time series forecasting, fuzzy systems, forecasting, deep learning, artificial intelligence.

МЕТОДЫ И СРЕДСТВА СИСТЕМ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ

С. Л. Крывый, Н. П. Дарчук, И.С. Ясенова, А.Л. Головина, А.С. Соляр

Аннотация: Рассматривается краткий обзор средств систем анализа, обработки и представления знаний, построения онтологий по естественноязыковым текстам. В частности, представлен обзор средств Дескриптивных Логик, которые активно развиваются и применяются. Приводится пример представления конкретной предметной области в виде онтографа с описанием операций на онтографах и на онтологиях.

ACM Classification Keywords: Systems of knowledge representation, Description Logics, ontograph, operations over ontologies

Введение

Одним из современных способов управления является информация. Информационное влияние – это форма управления, отличающаяся от других форм управления средством влияния, которым является содержание информации. Степень влияния информации на личность зависит от многих факторов: семантического содержания информации, способом представления и источника информации, уровня информационной культуры личности и т. п.. Степень влияния информации базируется на ее свойствах, интересующих личность, а также на ее надежности и достоверности.

Развитие информационных технологий увеличило количество источников информационных потоков, созданных для открытой публичной передачи информации при помощи разных технических инструментов. Так, наряду с традиционными СМИ (печатные СМИ, радио, телевидение) в формировании информационных потоков большую роль играют онлайновые социальные сети (ОСС). ОСС являются каналами распространения информации не только официального содержания или непредвзятого освещения фактов, но и источниками, которыми есть политики, известные личности. Таким образом, ОСС – это одно из средств общения, которое позволяет каждому участнику выразить свой субъективный комментарий или оценку, или же самому быть источником информации – лично формировать контент.

Контент (заимствовано в 1990-е годы из английского content “содержание”) – основа коммуникации в ОСМ и способ построения взаимосвязей между участниками.

Каждый день ОСМ генерируют миллионы фактов в минуту как глобально-информативного, так и субъективно-эмоционального характера. Контент является средством влияния участников ОСМ один на другого и, соответственно способом анализа окружающей действительности с отслеживанием общественно-политических настроений. Распространенную разными участниками ОСМ факты часто могут быть связаны, а значит, подтверждаться и взаимодополняться. Но возможны случаи дублирования фактов, что может свидетельствовать либо о действительном множестве реальных информационных сведений, либо о информационной атаке, основной целью которой является распространение неправдивой или обманчивой информации. Очевидно, что для определения достоверности сведений связанное множество фактов весомее чем одиночные факты.

Проблемы анализа содержания информации

В связи со сказанным приобретают актуальность системы автоматического выделения различных фактов с целью их анализа и интеграции. Наиболее распространенной формой подачи фактов в ОСМ являются естественноязыковые тексты. Они легко воспринимаются, порождаются, тиражируются и модифицируются. Соответственно возникает проблема извлечения знаний из текстов. Неформально будем считать, что знания – это системно зафиксированная в сознании человека или информационной системы совокупность фактов, которые более менее адекватно отображают реальные взаимосвязи материальных и абстрактных объектов окружающего мира [1]. Исследованиям в сфере анализа текстов естественного языка посвящено большое количество отечественных и зарубежных публикаций, среди которых нужно выделить генеративную грамматику Хомского [2; 3], предикатно-аргументные структуры Ч. Гиллмора [4], модель концептов Р. Шенка [5; 6] и т.д. В трудах Хомского реализован подход к исследованию глубинной синтаксической структуры текста и построения дерева синтактико-семантических зависимостей и выявления семантических аномалий. В трудах Ч. Гиллмора и Р. Шенка были введены понятия “концепт” и “фрейм”: структуры типа “предикат-аргумент”. В трудах И. Мельчука [7] разработана теория, ориентированная на многоуровневое превращение содержания в текст и текста в содержание. В трудах Н. Дарчук разработаны основы теоретического и экспериментального обоснования лингвистических и процедурных принципов компьютерного аннотирования текста с созданием на этой основе компьютерной грамматики (система АГАТ) для автоматического анализа текстов естественного языка [8].

Модели, в которых используются синтаксические зависимости и толково-комбинаторные словари (см. [9]) дали начало современным тезаурусу и онтологии. Так, под семантикой языковых знаков в прикладной лингвистике понимают информацию, связанную со словом, его значением, которое представлено в толковом словаре. Под содержанием понимают функцию интерпретации конечной последовательности символов, в рамках априори согласованной семантики [1].

Анализ естественноязыковых текстов требует глубинного семантического анализа, который связан с построением модели фрагмента знаний, полученных из этих текстов. На сегодня ведущей парадигмой структурирования информации по

тематике, разделам и содержанию является онтологическая парадигма, которая предусматривает создание логикоориентованной формальной концептуальной модели предметной области Δ. Одним из средств автоматизации процесса построения таких моделей в данный момент выступают Дескриптивные Логики (ДЛ) [10], которые активно развиваются и служат базой для построения систем представления знаний, построения онтологий и исследования свойств полученных фактов из разных источников информации. Создание онтологий в виде концептуальной базы знаний является фундаментальной проблемой в сфере компьютерной лингвистики и инженерии знаний. Для решения этой проблемы необходимо разработать:

– методы анализа естественноязыковых текстов с целью получения знаний;

– методы проверки знаний на совместимость и непротиворечивость;

– методы построения онтологий.

Сложность решения данной проблемы связана с целым рядом лингвистических, логических, теоретических и технических проблем. Что касается лингвистических проблем, то естественный язык содержит много метафор – переносных значений слов, которые вытесняют начальные значения, присутствие которых значительно усложняет анализ текстов. Переносные значения характерны для каждой языковой культуры и понимаются субъектами коммуникации в зависимости от контекста. Соответственно, это же должны делать информационные системы формирования онтологий. Кроме этого, существует проблема концептов – присутствие у субъекта коммуникации некоторого представления о фрагменте мира или части такого фрагмента. Такой фрагмент имеет сложную структуру, выраженную разными группами признаков, которые реализуются разными языковыми способами и средствами [11]. Концепт отображает категориальные и ценностные характеристики знаний о некотором фрагменте мира, имеющего значение для субъекта коммуникации. Решения этих вопросов натыкается на ряд трудностей, учитывающих факторы сложности естественного языка и психологии восприятия информации, заложенной в этом языке. Значительной преградой на пути автоматизации процесса анализа текстов естественного языка является то, что часть значимой информации, как правило, находится за пределами данного текста. Восстановление этой недостающей информации людьми в процессе коммуникации является процессом естественным, а в формальных системах это совсем не так.

Важной задачей в этом процессе является построение соответственной терминологии. Значимость терминологии проявляется в том, что рассуждения выражаются определенной словесной формой в виде предложений некоторого полуформального языка L, а концепты – в виде терминов [12; 13]. В идеале каждый термин должен иметь единственный точно определенный смысл (это абсолютизируется в математике), чего нельзя достичь в естественном языке в связи с неоднозначностью его семантики. Поскольку для естественного языка в целом этого достичь нельзя, то рассматривается несколько ослабленное требование: интерпретация (терминов) концептов должна быть как можно более точной. Чем точнее интерпретация концептов, тем более убедительными становятся рассуждения и результаты анализа. Это ослабленное требование дальше сужается к отдельно взятой предметной области (ПО). В некоторых ПО оно может выполняться в полном объеме, а в некоторых,

более сложных, частично, но более менее удовлетворительно. С этой целью свойства объектов предметной области выражаются в некотором формальном языке L', который служит основой построения концептуальной формальной модели ПО и средством решения задач в этой ПО. Схематически такой процесс можно представить так:

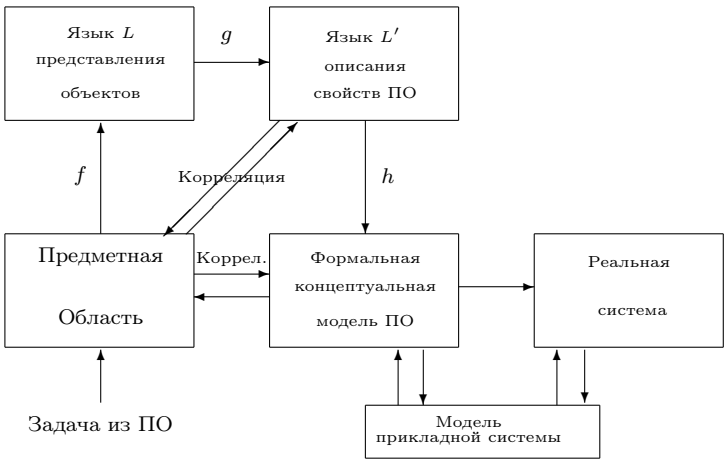


Рис 1. Общая схема

В процессе решения задачи в ПО возникают вопросы, связанные с языком представления объектов ПО, которые фигурируют в ПО.

Выбор языка представления объектов зависит от ПО и поставленной задачи. В идеале таким языком мог бы быть язык предикатов первого порядка или другой язык формальной логики с достаточными выразительными свойствами. Наличие такого языка облегчает задачу определения свойств как самого языка, так и объектов ПО. А это, в свою очередь, облегчает построение формальной модели.

После того как проведен анализ возникает необходимость проверки правильности найденных свойств. Эта проверка, названная Корреляцией, сводится к проверке истинности свойств на объектах ПО. Она может привести к пересмотру постановки задачи или к уточнению постановки задачи. Аналогичная потребность возникает при проверке построенной модели на соответствие поставленной задаче. Если модель решает задачу в рамках ограничений абстракции, то можно приступать к созданию реального изделия (если оно предусмотрено заданием) или использовать эту модель для других целей.

Названные проблемы объединяют сферы когнитивной и компьютерной лингвистики, инженерии знаний, теории поиска доказательств в разных языках математической логики, теории графов, психологии и т.д. В связи с названными проблемами в последнее время появилось много попыток решить (хотя бы частично) эти проблемы. Если средства построения онтологий более менее разработаны, то проблемы получения знаний и проверки их совместимости (непротиворечивости) являются одними из основных в этой области.

Одной из попыток построения систем представления и манипулирования знаниями, как отмечалось, являются дескриптивные логики [10], которые в данный момент активно развиваются и используются и в которых большое внимание уделяется формализации построения терминологии ПО.

Неформальное описание ДЛ

ДЛ – это совокупность формализмов для представления знаний некоторой ПО (такую ПО иногда называют “миром”). С помощью формализмов ДЛ сначала определяют концепты данной ПО, а потом, пользуясь этими концептами, специфицируют свойства объектов и индивидуумов (констант) этой ПО. Далее, опираясь на найденные свойства, проверяют некоторые гипотезы, тезисы или факты, являются ли они следствиями этих свойств.

Само название ДЛ выражает одну из характеристик этих логик, а именно, формальную логико-ориентированную семантику. Другой характерной особенностью ДЛ является возможность проводить логический вывод новых фактов из существующих, что составляет ее основной сервис. ДЛ поддерживает средства вывода не только свои собственные, но и средства других систем, с которыми она сотрудничает или в которые она встроена. Классификация концептов определяет подконцепты, надконцепты и отношения (которые называют отношениями включения) между концептами в заданной терминологии, что дает возможность структурировать терминологию в виде некоторой иерархии. Такого типа иерархия открывает полезную информацию на связанных между собой концептах, что ускоряет процесс вывода в центральном сервисе. Поскольку ДЛ является формализмом для представления знаний, то, как правило, предполагается, что система представления знаний должна всегда отвечать на вопрос пользователя в разумном интервале времени. В связи з этим возникает интерес к процедурам вывода в ДЛ и исследования их алгоритмической разрешимости. В отличии от пруверов в логике первого порядка, такие процедуры должны быть терминальными (то есть, всегда заканчивать свою работу) независимо от того, позитивным или негативным будет ответ. Следовательно, проблема поиска разрешимых процедур вывода, которые работают в разумном интервале времени и их временные характеристики, является актуальной в ДЛ.

Сложность решения проблемы вывода зависит от выразительных возможностей ДЛ. Эта сложность заключается в том, что существует некоторое противоречие между выразительными возможностями логики и сложностью вывода в ней: большие выразительные возможности логики имеют малоэффективные процедуры вывода и даже могут приводить к алгоритмической неразрешимости проблемы вывода, а слабые выразительные возможности с эффективными процедурами вывода становятся неспособными определить важные свойства концептов в данной ПО. Решение этой противоречивости является одной из самых важных проблем, которые особенно интересуют исследователей. Первые шаги и идеи на пути преодоления указанной противоречивости при разработке ДЛ были взяты из так называемых “структурно-наследственных рабочих станций” [14; 15]. Следующие идеи были заимствованы из этих работ:

а) базовыми синтаксическими блоками являются атомарные концепты (унарные предикаты), атомарные роли (бинарные базовые предикаты) и индивидуумы (константы);

б) выразительные возможности ДЛ ограничивались только использованием относительно небольшого множества конструкторов для построения более сложных концептов и ролей;

в) простейшие знания о концептах и индивидуумах можно вывести автоматически с помощью процедур выведения. В частности, отношение включения между концептами и отношения между индивидуумами играют при этом ключевую роль.

Построенная на этой основе семантика языка, которая была названа KL-One, основывалась на логике первого порядка. Это дало возможность использовать процедуры вывода в этой логике и придать точное понимание фактам, что в результате привело к формальному исследованию свойств таких языков. Первые результаты показали, что логика первого порядка достаточно выразительна, но в ней алгоритмически неразрешима проблема выполнимости отношений включения, а временная сложность не является полиномиальной [16]. Позже были найдены более менее приемлемые способы решения этих проблем. Прежде чем перейти к их обзору и задачам, которые при этом возникают, приведем основные понятия ДЛ и структуру систем представления знаний на основе ДЛ.

Языки ДЛ

01 Базовые формализмы ДЛ

Система представления знаний (СПЗ) в ДЛ включает две компоненты:

– TBox, которая содержит терминологию, то есть словарь ПО;

– ABox, которая включает утверждения о найденных индивидуумах в терминах введенной терминологии, то есть в терминах словаря ПО.

Словарь состоит из концептов, которые обозначают множества индивидуумов, и ролей, которые являются бинарными отношениями между индивидуумами. Дополнительно к атомарным концептам и ролям (именам концептов и ролей) пользователем добавляются в ДЛ более сложные описания концептов и ролей. Следовательно, TBox может использоваться для присваивания имен сложным описаниям этих объектов в ДЛ-системе. Язык, который используется для построения описаний, является характеристикой каждой ДЛ-системы и разные ДЛ-системы имеют разные языки описания. Язык описаний имеет теоретико-модельную семантику. Следовательно, утверждения в TBox и в ABox могут быть представлены некоторой формулой логики предикатов первого порядка или, в некоторых случаях, в ее незначительном расширении. ДЛ-система не только сохраняет терминологию и утверждения, но также и дополнительные сервисы, с помощью которых можно делать выводы. Типичным заданием такого типа является проверка противоречивости описаний объектов, или проверка того, является ли некоторое описание более общим, чем другое. Важной проблемой для ABox является определение, будет ли некоторое множество утверждений совместимым, то есть будет ли это множество иметь модель.

В многих использованиях систем представления знаний ДЛ встроены в более общие системы, а их компоненты взаимодействуют с другими компонентами другой системы путем использования запросов к базе знаний. Таким образом, система представления знаний с использованием ДЛ принимает вид:

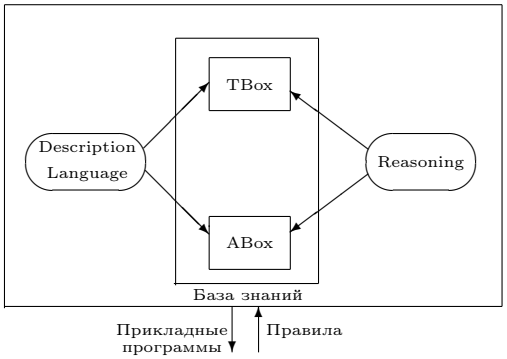


Рис 2. Архитектура системы представления знаний на основе ДЛ

02 Базовый атрибутный язык AL

Одним из основных средств ДЛ являются языки, с помощью которых представляются знания и осуществляется их обработка. Из вышесказанного следует, что изначально существует два вида объектов, которые назывались атомарными концептами и атомарными ролями. Атомарные концепты обозначаются буквами A, B, C, D, а атомарные роли символами R, S. Из атомарных концептов и ролей как элементарных описаний индуктивно строятся более сложные описания с помощью соответственных конструкторов. Поскольку конструкторы могут быть разные, то и языки описаний отличаются по типам этих конструкторов. Начальным языком описания является Атрибутный Язык (AL), основные объекты которого определяются следующим образом.

Пусть C и D – произвольные концепты. Описание базисного AL выполняется в соответствии с таким синтаксисом:

$C, D ::= A$ (атомарный концепт) | $\top$ (универсальный концепт)

| $\bot$ (пустой концепт)| $\neg A$ (отрицание атомарного концепта)

| $C \sqcap D$ (пересечение)| $\forall R.C$ (ограниченное значение)

| $\exists R.\top$ (ограниченный квантор существования).

Заметим, что в AL отрицание применяется только к атомарным концептам, а для квантора существования областью его действия является универсальный концепт.

Для демонстрации того, что можно выразить в AL, рассмотрим пример. Пусть $Person$ и $Female$ являются атомарными концептами. Тогда $Person \sqcap Female$ и $Person \sqcap \neg Female$ являются концептами в AL, которые означают, что “персона является женщиной” и что “персона не является женщиной”. Если дополнительно предположить, что $hasChild$ означает атомарную роль, то можно образовать концепты $Person \sqcap \exists hasChild.\top$ и $Person \sqcap \forall hasChild.Female$, которые описывают те персоны, которые имеют ребенка и те персоны, которые имеют детей и являются

женщинами. Используя $\perp$-концепт, можно описать тех персон, которые не имеют детей, с помощью такого концепта $Person \sqcap \forall hasChild.\perp$.

Для определения формальной семантики AL, рассмотрим интерпретацию I над непустой областью Δ^I (область интерпретации), которая является функцией, которая присваивает каждому атомарному концепту A некоторое множество $A^I \subseteq \Delta^I$ и каждой атомарной роле R бинарное отношение $R^I \subseteq \Delta^I \times \Delta^I$. Функция интерпретации расширяется на описание концептов с помощью такого индуктивного определения:

– $\top^I = \Delta^I$,

– $\perp^I = \emptyset$,

– $(\neg A)^I = \Delta^I \setminus A^I$,

– $(C \sqcap D)^I = C^I \cap D^I$,

– $(\forall R.C)^I = \{a \in \Delta^I | \forall b\ (a,b) \in R^I \rightarrow b \in C^I\}$,

– $(\exists R.\top)^I = \{a \in \Delta^I | \exists b\ (a,b) \in R^I\}$.

Два концепта C и D считаются эквивалентными и это записывается как $C \equiv D$, если $C^I = D^I$ для произвольной интерпретации I. Например, из определения семантики концептов следует, что $\forall hasChild.Female \sqcap \forall hasChild.Student$ и $\forall hasChild(Female \sqcap Student)$ эквивалентны.

03 Семейство дескриптивных языков

Более выразительные дескриптивные языки можно получить, если ввести дополнительные конструкторы в язык AL. Такими конструкторами выступают:

– Объединение концептов (обозначение $\mathcal{U}$) $C \sqcup D$ с интерпретацией

$$(C \sqcup D)^I = C^I \cup D^I,$$

– Полный квантор существования (обозначение $\mathcal{E}$) $\exists R.C$ с интерпретацией

$$(\exists R.C)^I = \{a \in \Delta^I | \exists b.(a,b) \in R^I \wedge b \in C^I\},$$

– Числовое ограничение (обозначение $\mathcal{N}$) $\geq nR$ (не меньше) и $\leq nR$ (не больше) с интерпретациями

$$(\geq nR)^I = \{a \in \Delta^I || \{b|(a,b) \in R^I\}| \geq n\},\ \ (\leq nR)^I = \{a \in \Delta^I || \{b|(a,b) \in R^I\}| \leq n\}$$

соответственно,

– Отрицание произвольного концепта (обозначение $\mathcal{C}$) $\neg C$ с интерпретацией

$$(\neg C)^I = \Delta^I \setminus C^I.$$

С помощью введенных конструкторов можно выразить, например, описание тех персон, которые имеют не больше одного ребенка, и тех женщин, которые имеют не меньше трех детей:

$$Person \sqcap (\leq 1 hasChild \sqcup (\geq 3 hasChild \sqcap \exists hasChild.Female)).$$

Расширение AL с помощью произвольного подмножества введенных конструкторов порождает соответствующие AL-языки. Каждый из этих языков обозначается аббревиатурой вида

$$AL[\mathcal{U}][\mathcal{E}][\mathcal{N}][\mathcal{C}].$$

Например, $AL\mathcal{EN}$ является AL-языком, в котором присутствуют конструкторы $\mathcal{E}$ и $\mathcal{N}$.

С семантической точки зрения не все эти языки являются разными, поскольку имеет место некоторая семантическая эквивалентность. Например, $C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$ и $\exists R.C \equiv \neg\forall R.\neg C$. Следовательно, объединение и полный квантор существования можно выразить, используя отрицание. Наоборот, комбинация объединения и полного квантора существования дает возможность выразить отрицание концептов через их эквивалентную нормальную негативную форму (форма, в которой отрицания стоят только перед атомарными концептами). Поэтому без ограничения общности считается, что объединение и полный квантор существования присутствуют в каждом AL-языке, который включает отрицание и наоборот. Отсюда следует, что все AL-языки могут быть записаны только с помощью $\mathcal{U}, \mathcal{E}, \mathcal{N}$. Несложно понять, что все восемь языков, которые можно получить таким образом, попарно не эквивалентны. Далее аббревиатуры ALC, $AL\mathcal{UE}$, $AL\mathcal{CN}$ и $AL\mathcal{UEN}$ означают один и тот же язык.

04 AL-языки как фрагменты логики предикатов

Семантические понятия AL-языков показывают, что эти языки являются фрагментами логики предикатов первого порядка. На основе того, что интерпретация I присваивает каждому атомарному концепту и роли соответствующие унарные и бинарные отношения на множестве Δ^I, то на атомарные концепты и роли можно смотреть как на соответствующие унарные и бинарные предикаты. Тогда каждый концепт C можно транслировать в формулу логики предикатов $\psi_C(x)$ с одной свободной переменной x такой, что для произвольной интерпретации I множество элементов из Δ^I, которое выполняет $\psi_C(x)$, в точности совпадает с множеством C^I. При этом атомарный концепт A транслируется в формулу $A(x)$, а конструкторы пересечения, объединения и отрицания транслируются в логическую конъюнкцию, дизъюнкцию и отрицание соответственно. Если C уже странслированное в $\psi_C(x)$ и R – атомарная роль, то кванторы существования и общности выражаются формулами

$$\psi_{\exists R.C}(y) = \exists x.R(y,x) \wedge \psi_C(x),$$
$$\psi_{\forall R.C}(y) = \forall x.R(y,x) \rightarrow \psi_C(x),$$

где y – новая переменная, а числовое ограничение значений выражаются формулами

$$\psi_{\geq nR}(x) = \exists y_1, \ldots, \exists y_n . R(x, y_1) \wedge \ldots \wedge R(x, y_n) \wedge \bigwedge_{i<j} y_i \neq y_j,$$

$$\psi_{\leq nR}(x) = \forall y_1, \ldots, \forall y_{n+1} . R(x, y_1) \wedge \ldots \wedge R(x, y_{n+1}) \wedge \bigvee_{i<j} y_i = y_j.$$

Следует заметить, что предикат равенства "=" необходим, поскольку с его помощью выражаются числовые ограничения. Но так как концепты транслируются в логику предикатов, то вводить специальный синтаксис нет необходимости.

Терминологии

Из сказанного следуют способы построения сложных описаний концептов для представления классов объектов. Рассмотрим терминологические аксиомы, которые выражают связи между концептами и ролями. Выбирая определения как специфические аксиомы и идентифицируя терминологию как множество определений, мы можем вводить атомарные концепты как аббревиатуры или имена для сложных концептов. Если определения в терминологии создают цикл, то к этой ситуации адаптируются семантики неподвижной точки с целью сделать терминологии однозначными. Будут рассмотрены также типы терминологий, для которых существуют модели семантик неподвижной точки.

05 Терминологические аксиомы

В наиболее общем виде терминологические аксиомы имеют вид

$$C \sqsubseteq D \ (R \sqsubseteq S) \text{ или } C \equiv D \ (R \equiv S),$$

где C, D – концепты (R, S – роли). Аксиомы первого типа называются включениями, а аксиомы второго типа – равенствами. Далее будут рассматриваться только аксиомы для концептов, потому что для ролей они аналогичны.

Интерпретация I выполняет включение $C \sqsubseteq D$ тогда и только тогда, когда $C^I \subseteq D^I$ и выполняет равенство $C \equiv D$ тогда и только тогда, когда $C^I = D^I$. Если $\mathcal{T}$ терминология с множеством аксиом, то интерпретация I выполняет $\mathcal{T}$ тогда и только тогда, когда I выполняет каждую аксиому терминологии $\mathcal{T}$. Если интерпретация I выполняет аксиомы $\mathcal{T}$, то говорят, что I является моделью для $\mathcal{T}$. Две аксиомы или два множества аксиом эквивалентны, если они имеют одну и ту же модель.

06 Определения

Равенство, в которой левая часть является атомарным концептом, называется определением. Определение используется для введения символических имен для сложных описаний. Например, с помощью аксиомы

$$Mother \equiv Women \sqcap \exists hasChild.Person$$

мы достаем описание правой части для имени $Mother$. Символические имена могут быть использованы как аббревиатуры в других описаниях. Если, например, имеем определение $Father$, аналогичное $Mother$, то можно определить имя $Parent$ у виде

$$Parent \equiv Mother \sqcup Father.$$

Множество определений может быть неоднозначным. Неоднозначность влияет на однозначность терминологии.

Конечное множество определений $\mathcal{T}$ называется терминологией или TBox, если нет ни одного имени, которое определяется больше одного раза, то есть когда для каждого атомарного концепта A существует не больше одной аксиомы в $\mathcal{T}$, левая часть которой – концепт A. На рис. 3 показана терминология, которая описывается семейные отношения.

Пример 1. Терминология (TBox) с концептами про семейные отношения:

$$\begin{array}{rcl}
\text{Woman} & \equiv & \text{Person} \sqcap \text{Female} \\
\text{Man} & \equiv & \text{Person} \sqcap \neg\ \text{Female} \\
\text{Mother} & \equiv & \text{Women} \sqcap \exists\ \text{hasChild.Person} \\
\text{Father} & \equiv & \text{Man} \sqcap \exists\ \text{hasChild.Person} \\
\text{Parent} & \equiv & \text{Father} \sqcup \text{Mother} \\
\text{Grandmother} & \equiv & \text{Mother} \sqcap \exists\ \text{hasChild.Parent} \\
\text{Grandfather} & \equiv & \text{Father} \sqcap \exists\ \text{hasChild.Parent} \\
\text{MotherWithManyChildren} & \equiv & \text{Mother} \sqcap \geq 3\ \text{hasChild} \\
\text{MotherWithout Daughter} & \equiv & \text{Mother} \sqcap \forall\ \text{hasChild.}\neg\ \text{Women} \\
\text{Wife} & \equiv & \text{Women} \sqcap \exists\ \text{hasHusband.Man}
\end{array}$$

Рис. 3. Терминология (TBox) для семейных отношений

Пусть $\mathcal{T}$ – некоторая терминология. Атомарные символы, которые входят в $\mathcal{T}$, делятся на два подмножества:

– символы-имена $\mathcal{N}_\mathcal{T}$, которые входят только в левые части аксиом;

– базовые символы $\mathcal{B}_\mathcal{T}$, которые входят только в правые части аксиом.

Символы-имена часто называют определенными концептами, а базовые символы – примитивными концептами. Таким образом, терминология уточняет определение имен концептов в терминах базовых концептов.

Базовой интерпретацией терминологии $\mathcal{T}$ называется интерпретация, которая интерпретирует только базовые символы. Пусть $\mathcal{J}$ – базовая интерпретация. Интерпретация $\mathcal{I}$, которая интерпретирует символы-имена, называется расширением $\mathcal{J}$, если она имеет ту же самую область интерпретации, что и $\mathcal{J}$ (то есть $\Delta^I = \Delta^J$) и согласуется с $\mathcal{J}$ для базовых символов.

Терминология $\mathcal{T}$ называется дефиниторной, если каждая базовая интерпретация имеет единственное расширение, что образует модель $\mathcal{T}$. Другими словами, если нам известна интерпретация базовых символов и $\mathcal{T}$ – дефиниторная, то понимание символов-имен полностью детерминированное (однозначное). Очевидно, что когда терминология дефиниторная, то произвольная терминология, которая ей эквивалентна, тоже дефиниторная.

Дефиниторность терминологии зависит от того, будет или нет множество определений цикличным. Например, терминология, которая состоит только из одной аксиомы

$$Human' \equiv Animal \sqcap \forall hasParent.Human', \tag{1}$$

имеет цикл, поскольку каждый $Human'$ определяется в терминах $Human'$.

Пусть A и B – атомарные концепты терминологии $\mathcal{T}$. Говорят, что концепт A непосредственно использует концепт B, если B встречается в правой части определения A. Транзитивное замыкание отношения непосредственного использования называется отношением использования.

Терминология $\mathcal{T}$ включает цикл тогда и только тогда, когда существует атомарный концепт в $\mathcal{T}$, который использует сам себя. В противном случае терминология $\mathcal{T}$ называется ациклической.

Если терминология имеет цикл, то она может иметь много расширений. Например, если рассмотреть терминологию с единственной аксиомой (1), то $Human'$ – это символ-имя, а $Animal$ и $hasParent$ являются базовыми символами. Интерпретация, которая $hasParent$ относит каждого $Animal$ к его предку, может иметь несколько возможных расширений интерпретации для $Human'$, удовлетворяющих аксиому (1). $Human'$ может интерпретироваться как множество всех зверей, как некоторый вид или какое-нибудь другое множество зверей со свойством, что каждый зверь имеет предка.

Если же $\mathcal{T}$ ациклическая, то она дефиниторная. Действительно, мы можем с помощью подстановок итеративным способом заменить в $\mathcal{T}$ все вхождения имен в правые части определений на базовые имена. Такой процесс всегда заканчивается в связи с ацикличностью $\mathcal{T}$ и в результате получаем терминологию $\mathcal{T}'$, которая называется расширением терминологии $\mathcal{T}$.

Следует отметить, что размеры расширения могут быть экспоненциальными в сравнении с размерами оригинальной терминологии [17].

Пример 2. Терминология (TBox) из примера 1 ациклическая и ее расширение получает вид:

Woman	≡	Person ⊓ Female
Man	≡	Person ⊓ ¬(Person ⊓ Female)
Mother	≡	(Person ⊓ Female) ⊓ ∃ hasChild.Person
Father	≡	(Person ⊓ ¬(Person ⊓ Female)) ⊓ ∃ hasChild.Person
Parent	≡	((Person ⊓ ¬(Person ⊓ Female)) ⊓ ∃ hasChild.Person) ⊔ ((Person ⊓ Female) ⊓ ∃ hasChild.Person)
Grandmother	≡	((Person ⊓ Female) ⊓ ∃ hasChild.Person) ⊓ ∃ hasChild.(((Person ⊓ ¬(Person ⊓ Female)) ⊓ ∃ hasChild.Person) ⊔ (Person ⊓ Female) ⊓ ∃ hasChild.Person))
MotherWithManyChildren	≡	((Person ⊓ Female) ⊓ ∃ hasChild.Person) ⊓ ≥ 3 hasChild
MotherWithout Daughter	≡	((Person ⊓ Female) ⊓ ∃ hasChild.Person) ⊓ ∀ hasChild.(¬(Person ⊓ Female))
Wife	≡	(Person ⊓ Female) ⊓ ∃ hasHusband.(Person ⊓ ¬(Person ⊓ Female))

Рис. 4. Расширение TBox для семейных отношений

Имеет место

Утверждение 1.. Пусть терминология $\mathcal{T}$ – ациклическая и $\mathcal{T}'$ ее расширение. Тогда

а) $\mathcal{T}$ и $\mathcal{T}'$ имеют одинаковое множество базовых символов;

б) $\mathcal{T}$ и $\mathcal{T}'$ эквивалентны;

в) $\mathcal{T}$ и $\mathcal{T}'$ дефиниторные.

Теорема 1.. Каждая дефиниторная ALC терминология эквивалентна ациклической терминологии.

Известно, что ALC-язык является вариантом пропозициональной модальной логики K_n [18].

07 Семантика неподвижных точек для терминологических циклов

Рассмотрение семантик неподвижной точки мотивируется тем фактом, что существуют ситуации, где циклические определения могут быть содержательными и это содержание описывается семантикой или наименьшей неподвижной точки, или наибольшей неподвижной точки.

Пример 3. Пусть необходимо описать концепт “мужчина, который имеет наследников только мужского пола” (сокращение $Momd$). В частности, такой мужчина имеет только сыновей (обозначим его Mos). Mos можно определить без циклов

$$Mos \equiv Man \sqcap \forall hasChild.Man.$$

Для $Momd$ мы хотим создать новые утверждения с помощью транзитивного замыкания роли $hasChild$. Это принимает такой вид: мужчина, который имеет наследников только мужчин, сам является таким мужчиной, а все его сыновья тоже являются мужчинами, которые имеют наследниками только мужчин. Следовательно,

$$Momd \equiv Man \sqcap \forall hasChild.Momd. \tag{2}$$

Для достижения однозначного понимания, необходимо обосновать это определение с помощью подходящей семантики неподвижной точки.

Далее будет показано, что семантика наибольшей неподвижной точки обосновывает ситуацию такого типа.

Пример 4. Пусть имеем множество объектов, которые называются $Trees$ и бинарное отношение $has\text{-}branch$ между объектами, которые порождают их поддеревья. Тогда бинарные деревья являются такими, что имеют не больше двух поддеревьев и которые, в свою очередь, тоже являются бинарными деревьями. Тогда определение $BinaryTree$ принимает вид:

$$BinaryTree \equiv Tree \sqcap \leq 2 has\text{-}branch \sqcap \forall has\text{-}branch.BinaryTree.$$

Как и в случае с $Momd$, семантика неподвижной точки дает желаемое понимание, но в данном случае – это семантика наименьшей неподвижной точки.

Определение семантик неподвижной точки. В терминологии $\mathcal{T}$ каждый символ A входит только один раз в левую часть аксиомы $A \equiv C$. Поэтому на $\mathcal{T}$ можно смотреть как на отображение, которое ассоциирует с символом-именем A его концепт $\mathcal{T}(A) = C$. В этих обозначениях интерпретация $\mathcal{I}$ являет собой модель терминологии $\mathcal{T}$ тогда и только тогда, когда $A^I = (\mathcal{T}(A))^I$.

Рассмотрим семейство этих отображений, для которых интерпретация является моделью терминологии $\mathcal{T}$ тогда и только тогда, когда она является неподвижной точкой некоторого отображения из этого семейства.

Пусть $\mathcal{T}$ – некоторая терминология и $\mathcal{J}$ – фиксированная базовая интерпретация. Пусть $Ext_{\mathcal{J}}$ означает расширение интерпретации $\mathcal{J}$ до интерпретации $\mathcal{T}_{\mathcal{J}}(\mathcal{I})$, которое определяется как

$$A^{\mathcal{T}_{\mathcal{J}}(\mathcal{I})} = (\mathcal{T}(A))^{\mathcal{J}}$$

для каждого символа-имени A.

Интерпретация $\mathcal{I}$ является неподвижной точкой $\mathcal{T}_{\mathcal{J}}$ тогда и только тогда, когда $\mathcal{I} = \mathcal{T}_{\mathcal{J}}(\mathcal{I})$, то есть тогда и только тогда, когда $A^I = A^{\mathcal{T}_{\mathcal{J}}(I)}$ для всех символов-имен. Это означает, что для каждого определения $A \equiv C$ в $\mathcal{T}$ имеет место $A^I = A^{\mathcal{T}_{\mathcal{J}}(I)} = (\mathcal{T}(A))^I = C^I$, а это означает, что $\mathcal{I}$ является моделью для $\mathcal{T}$.

Утверждение 2.. Пусть $\mathcal{T}$ – терминология, $\mathcal{I}$ – интерпретация и $\mathcal{J}$ – сужение $\mathcal{I}$ к базовым символам терминологии $\mathcal{T}$. Тогда $\mathcal{I}$ является моделью $\mathcal{T}$ тогда и только тогда, когда $\mathcal{I}$ неподвижная точка отображения $\mathcal{T}_{\mathcal{J}}$.

С этого утверждения следует, что терминология $\mathcal{T}$ дефиниторная тогда и только тогда, когда каждая базовая интерпретация $\mathcal{J}$ имеет единственное расширение, которое является неподвижной точкой $\mathcal{T}_{\mathcal{J}}$.

Пример 5. Рассмотрим пример терминологии, который показывает почему циклическая терминология не является дефиниторной. Это терминология $\mathcal{T}^{Momd}$, которая включает единственную аксиому (2). Пусть базовой интерпретацией $\mathcal{J}$ является такая интерпретация:

$\Delta^{\mathcal{J}} = \{Charles_1, Charles_2, \ldots\} \cup \{James_1, \ldots, James_{Last}\}$,

$Man^{\mathcal{J}} = \Delta^{\mathcal{J}}$,

$hasChild = \{Charles_i, Charles_{i+1} | i \geq 1\} \cup \{James_i, James_{i+1} | 1 \leq i < Last\}$.

Это означает, что династия $Charles$ живая, в то время как династия $James$ обрывается на $James_{Last}$.

Найдем неподвижные точки $\mathcal{T}_{\mathcal{J}}^{Momd}$. Отметим, что бездетный индивидуум, то есть без элементов у $hasChild$, всегда будет присутствовать в интерпретации $\forall hasChild$ независимо от того, как интерпретируется $Momd$. Следовательно, если $\mathcal{I}_1$ – неподвижная точка расширения $\mathcal{J}$, то $James_{Last}$ буде элементом в $(\forall hasChild)^{\mathcal{I}_1}$, а значит и элементом $Momd^{\mathcal{I}_1}$. Отсюда получаем, что каждый $James$ является $Momd$. Пусть $\mathcal{I}$ будет расширением $\mathcal{J}$ таким, что $Momd^{\mathcal{I}_1}$ охватывает только династию $James$. Тогда легко увидеть, что $\mathcal{I}$ является неподвижной точкой. Если к династии $James$ добавляется кто-то из династии $Charles$ как $Momd$, то все члены династии $Charles$ до и после этого члена должны принадлежать концепту $Momd$. Снова легко понять, что расширение $\mathcal{I}_2$, которое интерпретирует $Momd$, является неподвижной точкой и других неподвижных точек нет.

Для того чтобы дать дефиниторный вид циклической терминологии $\mathcal{T}$, необходимо определить отдельную неподвижную точку, если этих точек несколько. Для этого введем частичный порядок "$\preceq$" на расширениях базовой интерпретации $\mathcal{J}$:

$$\mathcal{I} \preceq \mathcal{I}' \Leftrightarrow A^{\mathcal{I}} \subseteq A^{\mathcal{I}'}$$

для каждого символа-имени A в $\mathcal{T}$. В вышеописанном примере $Momd$ является только символом-именем. Поскольку $Momd^{\mathcal{I}} \subseteq Momd^{\mathcal{I}'}$, то $\mathcal{I} \preceq \mathcal{I}'$.

Неподвижная точка $\mathcal{I}$ отображения $\mathcal{T}_\mathcal{J}$ называется наименьшей неподвижной точкой, если $\mathcal{I} \preceq \mathcal{I}'$ для каждой другой неподвижной точки $\mathcal{I}'$. Интерпретация $\mathcal{I}$ называется наименьшей неподвижной точкой терминологии $\mathcal{T}$, если $\mathcal{I}$ наименьшая неподвижная точка $\mathcal{T}_\mathcal{J}$ для некоторой базовой модели $\mathcal{J}$.

В рамках семантики наименьшей неподвижной точки допускаются только модели наименьшей неподвижной точки терминологии $\mathcal{T}$ как единственные интерпретации. Наибольшая неподвижная точка модели и семантики определяются аналогично. В примере $Momd$ интерпретация $\mathcal{I}_1$ является наименьшей неподвижной точкой, а интерпретация $\mathcal{I}_2$ – наибольшей неподвижной точкой отображения $\mathcal{T}_\mathcal{J}$.

Следует заметить, что модели неподвижной точки не всегда существуют. Например, пусть имеем терминологию с единственной аксиомой

$$A \equiv \neg A. \tag{3}$$

Если $\mathcal{I}$ – некоторая модель этой аксиомы, то $A^I = \Delta^I \setminus \Delta^I = \emptyset$, что абсурдно. Аксиома (3) не имеет никакой модели ни наименьшей ни наибольшей неподвижной точки.

Существуют случаи моделей неподвижной точки, для которых невозможно существование ни наименьшей ни наибольшей неподвижной точек. Например, рассмотрим терминологию $\mathcal{T}$ с единственной аксиомой

$$A \equiv \forall R.\neg A. \tag{4}$$

Пусть $\mathcal{J}$ – базовая интерпретация $\mathcal{T}$, где $\Delta^\mathcal{J} = \{a, b\}$ i $R^\mathcal{J} = \{(a, b), (b, a)\}$. Тогда существует два расширения $\mathcal{I}_1$ и $\mathcal{I}_2$ интерпретации $\mathcal{J}$, которые определяются с помощью $A^{\mathcal{I}_1} = \{a\}$, $A^{\mathcal{I}_2} = \{b\}$. Но эти расширения не сравниваются по отношению "$\preceq$".

Для обоснования семантик неподвижной точки терминологии напомним некоторые понятия из теории полных решеток [20].

Полной решеткой называется алгебра, носитель которой является частично упорядоченное множество с двумя операциями пересечения и объединения, которые также называются соответственно взятием наибольшей нижней и наименьшей верхней грани, в которой существует единичный элемент и каждое непустое подмножество имеет наименьшую верхнюю грань.

Множество интерпретаций $Ext_\mathcal{J}$ относительно частичного порядка $\preceq$ является полной решеткой. Действительно, для семейства интерпретаций $\mathcal{I}_i$, $i \in N$, определим $\mathcal{I}_0 = \bigsqcup_{i \in N} \mathcal{I}_i$ как поэлементное объединение $\mathcal{I}_i$-х, то есть для каждого символа-имени A будем иметь $A^{\mathcal{I}_0} = \bigcup_{i \in N} A^{\mathcal{I}_i}$. Тогда $\mathcal{I}_0$ становится наименьшей верхней гранью для $\mathcal{I}_i$-х, которая показывает, что $(Ext_\mathcal{J}, \preceq)$ является полной решеткой.

Функция $f : L \to L$ полной решетки L в себя называется монотонной, если $f(x) \preceq f(y) \Leftrightarrow x \preceq y$. Имеет место

Теорема 2 (Тарский). . Для монотонной функции на полной решетке существует непустое множество неподвижных точек и это множество тоже является полной решеткой [19].

Следовательно, для монотонной функции в полной решетке существует ее наименьшая и наибольшая неподвижные точки.

Определение 1.. Терминология $\mathcal{T}$ называется монотонной, если отображение $\mathcal{T}_{\mathcal{J}}$ монотонное для произвольной базовой интерпретации $\mathcal{J}$.

Из теоремы Тарского следует существования для монотонной терминологии наибольшей и наименьшей неподвижных точек.

Утверждение 3.. Если $\mathcal{T}$ – монотонная терминология и $\mathcal{J}$ – базовая интерпретация, то существуют расширения $\mathcal{J}$, которые являются моделями наименьшей верхней неподвижной и наибольшей нижней неподвижной точки терминологии $\mathcal{T}$ соответственно [10].

Для того чтобы найти синтаксический критерий монотонности, рассмотрим операторы AL-языка, которые использовались для концептов, а именно: $\sqcap, \sqcup, \neg, \forall, \exists$. Пусть эти операторы интерпретируются на области Δ. Им соответствуют теоретико-множественные операции пересечения, объединения, дополнения с теми самыми названиями на области Δ. Поскольку $\forall$ и $\exists$ применяются к ролям и концептам, то введем для бинарного отношения r на Δ унарную операцию A_r, которая отображает некоторое подмножество $S \subseteq \Delta$ в

$$A_r(S) = \{a \in \Delta | \forall b.\ (a, b) \in r \rightarrow b \in S\}$$

и унарную операцию E_r, которая отображает множество $S \subseteq \Delta$ в

$$E_r(S) = \{a \in \Delta | \exists b.\ (a, b) \in r \wedge b \in S\}.$$

Отсюда следует, что эти операции, за исключением дополнения, являются монотонными в том смысле, что когда они применяются к большим аргументам, то значение результата тоже будет большим. Это очевидно для пересечения и объединения, а для A_r и E_r нетрудно убедиться в том, что когда $S \subseteq S'$, то $A_r(S) \subseteq A_r(S')$ и $E_r(S) \subseteq E_r(S')$. К сожалению, подобные аналогии для отрицания не выполняются.

Определение 2.. Терминология называется $\neg$-свободной, если в ней нет ни одного знака отрицания.

Утверждение 4.. Произвольная $\neg$-свободная $ALCN$-терминология монотонна.

Расширим это достаточное условие, анализируя, какие из знаков отрицания не влияют на монотонность. Отрицание может влиять на монотонность, если в его области действия находится некоторой символ-имя. Однако отрицание может нейтрализоваться другим отрицанием, если оно входит в область действия другого отрицания. Отсюда следует такое

Определение 3.. Концепт C называется синтаксически монотонным, если каждый символ-имя в C входит в область действия парного числа знаков отрицания.

Терминология называется синтаксически монотонной, если каждый концепт в правой части равенства является синтаксически монотонным.

Утверждение 5.. Произвольная синтаксически монотонная $ALCN$-терминология монотонна.

08 Терминологии с аксиомами включения

Некоторые концепты невозможно определить в полном объеме. В таких случаях можно только формулировать необходимые условия принадлежности концепта к некоторому множеству, используя включения. Включения, левая часть которого атомарная, называется специализацией.

Например, когда инженер по знаниям считает, что определение “women” в нашем примере TBox (рис. 3) неисполнимое и он понимает, что не в состоянии полностью определить концепт “women”, то он потребует, чтобы все *woman* были *person* с помощью специализации

$$Woman \sqsubseteq Person. \tag{5}$$

Если такая специализация в терминологии допустима, то терминология теряет дефинитивность даже, если она ациклическая. Множество аксиом терминологии $\mathcal{T}$ называется обобщенной терминологией, если левые части ее аксиом являются атомарными концептами и для каждого атомарного концепта существует не более одной аксиомы, где он входит в левую часть.

Выполним преобразование обобщенной терминологии $\mathcal{T}$ в некоторую регулярную терминологию $\bar{\mathcal{T}}$, которая включает только такие определения, что $\bar{\mathcal{T}}$ эквивалентна $\mathcal{T}$ в некотором определенным смысле. В результате получим $\bar{\mathcal{T}}$ из $\mathcal{T}$ путем выбора для каждой специализации $A \sqsubseteq C$ в $\mathcal{T}$ новый базовый символ $\bar{A}$ и заменой специализации $A \sqsubseteq C$ на определение $A \equiv \bar{A} \sqcap C$. Терминология $\bar{\mathcal{T}}$ называется нормализацией $\mathcal{T}$.

Если TBox включает специализацию (5), то нормализация включает определение

$$Woman \equiv \overline{Woman} \sqcap Person.$$

Интуитивно дополнительный базовый символ $\overline{Woman}$ для равенства призван выделить женщин среди персон. Результат нормализации в TBox с определением для $Woman$ является подобным тому, как это делалось для $Female$ в TBox.

Утверждение 6.. Пусть $\mathcal{T}$ обобщенная терминология и $\bar{\mathcal{T}}$ ее нормализация. Тогда

а) каждая модель $\bar{\mathcal{T}}$ является моделью и для $\mathcal{T}$;

б) для каждой модели $\mathcal{I}$ терминологии $\mathcal{T}$ существует модель $\bar{\mathcal{I}}$ терминологии $\bar{\mathcal{T}}$, которая имеет ту же самую область интерпретации, что и $\mathcal{I}$ и согласуется с $\mathcal{I}$ на атомарных концептах и ролях $\mathcal{T}$.

Доказательство. Первый пункт выполняется в связи с тем, что модель $\bar{\mathcal{I}}$ терминологии $\bar{\mathcal{T}}$ удовлетворяет равенству $A^{\bar{\mathcal{I}}} = (\bar{A} \sqcap C)^{\bar{\mathcal{I}}} = \overline{A}^{\mathcal{I}} \cap C^{\mathcal{I}}$, а это означает что $A^{\bar{\mathcal{I}}} \subseteq C^{\bar{\mathcal{I}}}$.

Наоборот, если $\mathcal{I}$ модель $\mathcal{T}$, то расширение $\bar{\mathcal{I}}$ терминологии $\mathcal{T}$, определенное как $\overline{A}^{\bar{\mathcal{I}}} = A^{\mathcal{I}}$, является моделью $\bar{\mathcal{T}}$, поскольку $A^I \subseteq C^I$, а отсюда следует, что $A^I = \overline{A}^I \cap C^I = A^{\bar{\mathcal{I}}} \cap C^{\bar{\mathcal{I}}}$. Следовательно, $\bar{\mathcal{I}}$ выполняет $A = \bar{A} \sqcap C$. ■

Отсюда следует, что аксиомы включения не влияют на выразительность терминологии TBox. Но, с практической точки зрения, они являются удобным

средством введения термов в терминологию, которую невозможно полностью определить.

Описание миров

Другая компонента базы знаний является ABox – описание области интерпретации (мир, в котором интерпретируются концепты).

09 Утверждения об индивидуумах

ABox описывает свойства объектов области интерпретации в терминах концептов и ролей. Некоторые атомарные концепты и роли в ABox могут быть определены как имена TBox. В ABox вводятся индивидуумы путем предоставления им имен и описания свойств этих индивидуумов. Индивидуумы обозначаются $a, b, c, \ldots$. Используя концепты C и роли R, можно создать в ABox свойства таких типов:

$$C(a) \text{ i } R(b, c).$$

Первый тип называется свойством концепта, которое означает, что a принадлежит C. Второй тип называется свойством роли, которое означает, что c является наполнителем роли R для b. Например, если PETER, PAUL и MARY имена индивидуумов, то FATHER(PETER) означает, что PETER является отцом и HasChild(MARY,PETER) означает, что Peter является ребенком Mary. Следовательно, ABox $\mathcal{A}$ является конечным множеством таких утверждений. На рис. 5 показан пример ABox.

MatherWithoutDaughter(MARY) Father(PETER)
hasChild(MARY,PETER) hasChild(PETER,HARRY)
hasChild(MARY,PAUL)

Рис. 5. Описание ABox

С точки зрения пользователя, ABox выглядит как объект реляционной базы данных, который состоит только из унарных и бинарных отношений. Однако в классических базах данных отношения замкнуты в рамках данной предметной области, в то время как в ABox база знаний является открытой для данной предметной области, поскольку нормализованная база знаний является системой, в которой нет предположений о полноте ее знаний. Кроме этого, TBox связывает семантическими отношениями концепты и роли в ABox, что не имеет аналога в семантиках баз данных.

Семантика ABox определяется путем расширения на индивидуальные имена. Следовательно, интерпретация $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ отображает не только атомарные концепты и роли во множество отношений, а дополнительно отображает каждое индивидуальное имя a в некоторый концепт $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. При этом считается, что разные индивидуальные имена обозначают разные объекты. Поэтому это отображение должно соответствовать уникальным именам, то есть если a и b разные имена, то $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. Интерпретация $\mathcal{I}$ выполняет свойство концепта $C(a)$, если $a^{\mathcal{I}} \in C^{\mathcal{I}}$ и

выполняет свойство роли $R(a, b)$, если $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$. Интерпретация $\mathcal{I}$ выполняет ABox $\mathcal{A}$, если она выполняет все свойства в $\mathcal{A}$. В этом случае говорят, что $\mathcal{I}$ является моделью для ABox. Наконец, $\mathcal{I}$ выполняет свойство α или ABox $\mathcal{A}$ относительно TBox $\mathcal{T}$, если она является моделью не только для α или ABox, а и для TBox $\mathcal{T}$. Следовательно, модель $\mathcal{A}$ і $\mathcal{T}$ является интерпретацией конкретной ПО, концепты которой интерпретируются как подмножества этой области в терминах концептов и ролей из ABox.

Для индивидуальных имен в языках ДЛ допускается конструктор $\{a_1, \ldots, a_n\}$, где a_i – индивидуальное имя с интерпретацией $\{a_1, \ldots, a_n\}^{\mathcal{I}} = \{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\}$.

Вторым конструктором, который включает индивидуальные имена, является “fills”-конструктор для роли R. Семантика этого конструктора имеет вид:

$$(R : a)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} | (d, a^{\mathcal{I}}) \in R^{\mathcal{I}}\},$$

т. е. $(R : a)$ означает множество объектов, для которого a является наполнителем роли R. Оба конструкторы не добавляют ничего нового, поскольку $(R : a)$ и $\exists R.\{a\}$ эквивалентны.

Проблемы вывода в ДЛ

Системы представления знаний, которые основываются на ДЛ, могут осуществляют разного рода логические выведения. Как отмечалось выше, ДЛ с заданными ABox и TBox имеют семантики, которые делают их эквивалентными множеству аксиом логики предикатов первого порядка. Следовательно, в ДЛ можно выполнять выведения новых фактов. Например, с TBox на рис. 3 и ABox з рис. 5 можно вывести, что $Mary$ является бабушкой, поскольку этот факт явно не фигурирует в TBox. Кроме логики первого порядка в ДЛ допустимы и другие виды выведения. Их делят на выводы только для концептов, выводы в ABox, выводы в TBox и выводы в TBox и ABox вместе.

010 Вывод для концептов

В процессе моделирования конкретной ПО строится терминология, например $\mathcal{T}$, путем определения новых концептов, возможно, в терминах других концептов, которые были определены ранее. В этом процессе важным является идентификация того, будут ли нововведенные концепты совместимыми или они будут противоречить уже введенным концептам. С логической точки зрения, концепт непротиворечив, если существует некоторая интерпретация, которая удовлетворяет аксиомы $\mathcal{T}$ (т. е., является моделью $\mathcal{T}$). Концепт с таким свойством называется выполнимым относительно терминологии $\mathcal{T}$ и невыполнимым в противоположном случае.

Проверка выполнимости концептов является ключевой задачей для процесса вывода. Например, для того чтобы проверить корректность модели ПО или оптимизировать запросы, которые формулируются в виде концептов, необходимо знать, что некоторый концепт является более общим, чем другой. Эта задача называется проблемой включения (subsumption problem). Другими интересными отношениями

между концептами являются эквивалентность и попарное непересечение их подмножеств (disjointness). Эти свойства имеют такие формальные определения.

Пусть $\mathcal{T}$ – терминология (TBox). Тогда

Выполнимость: Концепт C называется выполнимым в терминологии $\mathcal{T}$, если существует модель $\mathcal{I}$ терминологии $\mathcal{T}$ такая, что $C^{\mathcal{I}} \neq \emptyset$. В этом случае говорят, что $\mathcal{I}$ является моделью C.

Включение: Концепт C включается в концепт D в терминологии $\mathcal{T}$, если $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ для каждой интерпретации $\mathcal{I}$ терминологии $\mathcal{T}$. Это обозначается как $C \subseteq_{\mathcal{T}} D$ или $\mathcal{T} \models C \sqsubseteq D$.

Эквивалентность: Концепты C и D эквивалентны в терминологии $\mathcal{T}$, если $C^{\mathcal{I}} = D^{\mathcal{I}}$ для каждой интерпретации $\mathcal{I}$ терминологии $\mathcal{T}$. Это обозначается как $C \models_{\mathcal{T}} D$ или $\mathcal{T} \models C \equiv D$.

Непересечение: Концепты C и D не пересекаются в терминологии $\mathcal{T}$, если $C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$ для каждой интерпретации $\mathcal{I}$ терминологии $\mathcal{T}$.

Пример 6. В терминологии TBox з рис. 3 $Person$ включает $Woman$, $Woman$ и $Parent$ оба включают $Mother$, а $Mother$ включает $Grandmother$, $Woman$ и Man, $Father$ и $Mother$ не пересекаются. Отношение включения следует из семантики операций $\sqcap$ и $\sqcup$.

Традиционными правилами выведения в ДЛ системах являются следующие редукции.

Утверждение 7 (Редукция к включению). Для концептов C и D справедливы такие эквивалентности в заданной терминологии $\mathcal{T}$:

1. C не выполняется тогда и только тогда, когда C включается в $\bot$;

2. C и D эквивалентны тогда и только тогда, когда C включается в D и D включается в C;

3. C и D не пересекаются тогда и только тогда, когда $C \sqcap D$ включается в $\bot$.

Дополнительными правилами выведения в ДЛ выступают правила, построенные на свойствах конструктора $\sqcap$.

Утверждение 8 (Редукция к невыполнимости). Для концептов C и D справедливы такие эквивалентности в заданной терминологии $\mathcal{T}$:

1. C включается в D тогда и только тогда, когда $C \sqcap \neg D$ не выполняется;

2. C и D эквивалентны тогда и только тогда, когда ни $C \sqcap \neg D$ ни $\neg C \sqcap D$ не выполняются;

3. C и D не пересекаются тогда и только тогда, когда $C \sqcap D$ не выполняется.

Приведенные правила имеют очевидную теоретико-множественную интерпретацию.

011 Удаление TBox

В практическом использовании желательно, чтобы TBox была пустой. Это можно сделать путем подстановок так, как это было показано в примере 2. Нетрудно понять, что когда терминология ациклическая, то это всегда можно сделать и таким образом свести проблему выведения в $\mathcal{T}$ к проблеме выведения в пустой терминологии TBox. Поскольку $\mathcal{T}$ и ее расширение $\mathcal{T}'$ эквивалентны, то произвольный концепт C и его расширение C' будут иметь одинаковую интерпретацию. То есть,

– $C \equiv_{\mathcal{T}} C'$ и

– C выполняется в $\mathcal{T}$ тогда и только тогда, когда выполняется C'.

Аналогично получаем

– $\mathcal{T} \models C \sqsubseteq D$ тогда и только тогда, когда $\models C' \sqsubseteq D'$ и

– $\mathcal{T} \models C \equiv D$ тогда и только тогда, когда $\models C' \equiv D'$ и

– C і D не пересекаются в $\mathcal{T}$ тогда и только тогда, когда C' і D' не пересекаются.

В этом случае возникает вопрос о сложности процедур вывода в $\mathcal{T}$, поскольку ее размеры могут возрастать экспоненциально. Анализ сложности процедур показывает, что они эффективнее работают в терминологии с пустой компонентой TBox [17; 21].

Отношения произвольной арности

Как следует из описания ДЛ, основными отношениями в ней являются бинарные отношения. В реальных использованиях возникает потребность в отношениях произвольной арности. Рассмотрим одно из расширений ДЛ, которое естественным образом вводит отношение произвольной арности и которое описано в работах [22; 23]. Полученный в результате такого расширения ДЛ-язык называется ДЛR.

Базовыми элементами ДЛR являются атомарные отношения и атомарные концепты, которые обозначаются P и A соответственно. Произвольное отношение арности $2 \leq n \leq n_{max}$ и произвольные концепты строятся по такому синтаксису:

$R ::= \top_n | P | (\$i/n : C) | \neg R | R_1 \sqcap R_2,$

$C ::= \top_1 | A | \neg C | C_1 \sqcap C_2 | \exists[\$i]R | \leq k[\$i]R,$

где i – i-я компонента отношения, то есть целое число $1 \leq i \leq n_{max}$, $n_{\max}$ – максимальная арность отношений P, R, R_1, R_2 и k – некоторое натуральное число.

Концепты и отношения должны быть согласованы по арности, т. е. комбинировать можно только отношения одной и той же арности.

Семантика ДЛR определяется обычным образом с помощью интерпретации $\mathcal{I} = (\Delta^{\mathcal{I}}.I)$, в которой функция интерпретации $.I$ присваивает каждому концепту C подмножество $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, и каждому n-арному отношению R – подмножество $R^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$ так, что

$\top_n^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$

$P^{\mathcal{I}} \subseteq \top_n^{\mathcal{I}}$

$(\neg R)^{\mathcal{I}} = \top_n^{\mathcal{I}} \setminus R^{\mathcal{I}}$

$(R_1 \sqcap R_2)^{\mathcal{I}} = R_1^{\mathcal{I}} \sqcap R_2^{\mathcal{I}}$

$(\$i/n : C)^{\mathcal{I}} = \{(d_1, \ldots, d_n) \in \top_n^{\mathcal{I}} | d_i \in C^{\mathcal{I}}\}$

$\top_1^{\mathcal{I}} = \Delta^{\mathcal{I}}$

$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$

$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$

$(C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \sqcap C_2^{\mathcal{I}}$

$(\exists[\$i]R)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} | \exists(d_1, \ldots, d_n) \in R^{\mathcal{I}}.d_i = d\}$

$(\leq k[\$i]R)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} || \{(d_1, \ldots, d_n) \in R^{\mathcal{I}}.d_i = d\}| \leq k\},$

где P, R, R_1, R_2 отношения арности n. Отметим, что $\top_1$ обозначает область интерпретации, в то время как $\top_n$ обозначает не декартовое произведение этой области, а только некоторое его подмножество, которое включает все n-арные отношения, что определялись на этой области. Отсюда следует, что отрицание на отношениях определяется как разница отношений.

Конструкция $(\$i/n : C)$ означает все n-ки из $\top_n$, которые являются i-ми компонентами концепта C. Эта конструкция являет собой разновидность операции селекции. Квантор существования и числовые ограничения на отношениях являются обобщениями соответствующих конструкций для ролей.

На этом обзор средств ДЛ закончим и рассмотрим свойства отношений общего типа, с помощью которых описываются ПО.

Отношения описания ПО общего типа

Отношения общего типа разделим на семантические и математические отношения и рассмотрим связь между этими типами отношений.

Семантические отношения, о которых пойдет речь далее, выработаны в лингвистике, и поэтому их часто называют лингвистическими. Основными типами таких отношений являются [8]:

– Род-вид (А есть родовым к В),

– Синонимии (А синонимичен к В),

– Часть-целое (В часть А),

– Корреляции (А противоположный к В),

– Ассоциации (А ассоциируется с В),

– Оперирования (В есть операцией для А),

– Функциональное (А выражает В),

– Абстрагирования (А ограничивает В),

– Импликации (если А, то В),

– Классификации (А относится к классу В),

– Принадлежности (А является объектом В) и другие.

Наиболее важными семантическими отношениями являются отношения Род-вид, Синонимии, Часть-целое и Корреляции в связи с тем, что они охватывают более 70 процентов имеющихся терминов в украинском, русском и других языках.

Математические отношения. Наряду с вышеприведенными семантическими отношениями, имеются и отношения общего типа, играющие фундаментальную роль в математике. Эти отношения будем называть математическими отношениями. В целях автоматизации процесса анализа естественноязыковых текстов необходима формализация этого процесса и, в частности, отношения следующих типов:

– n-арные отношения ($n \geq 3$),

– бинарные,

– подобия,

– аналогии,

– обобщения,

– специализации,

– функциональные и другие отношения более частного характера.

Наиболее важными среди этих отношений являются бинарные отношения в связи с тем, что большинство из перечисленных отношений можно выразить в терминах бинарных.

Бинарные отношения делятся на отношения

– тождества,

– рефлексивные,

– иррефлексивные,

– симметрические,

– транзитивные,

– антисимметричные,

– асимметричные,

– репродуктивные,

– эвклидовы,

– плотности и другие.

Отношением подобия в терминах бинарных отношений называется рефлексивное и симметричное бинарное отношение. С помощью этих отношений определяются бинарные отношения, которые в математике имеют фундаментальное значение. К ним относятся отношения

– эквивалентности,

– частичного порядка и его вариации,

– функциональное и его вариации.

Отношение эквивалентности – это рефлексивное, симметричное и транзитивное отношение.

Отношение частичного порядка – это рефлексивное, антисимметричное и транзитивное отношение. Его вариациями являются отношения квазипорядка – рефлексивное и транзитивное отношение; отношение строгого порядка – иррефлексивное и транзитивное отношение; линейного порядка – это отношение частичного порядка с условием тотальности (любые два элемента сравнимы); отношение полного порядка – это отношение линейного порядка с условием существования наименьшего элемента в непустых подмножествах множества, на котором это отношение определено.

Функциональное отношение – это отношение, которое ставит в соответствие некоторым элементам одного и того же множества, или разных множеств не более одного элемента.

Взаимоотношения между отношениями. Рассмотрим отношения Род-вид, Часть целое, Принадлежности. Эти семантические отношения удовлетворяют свойствам рефлексивности, транзитивности и антисимметричности, а поэтому являются математическим отношением частичного порядка. Отсюда следует, что их представление носит иерархический характер и их элементы можно представлять в виде такой структуры данных как дерево или ациклический граф.

Отношение Синонимии и Классификации (имеется в виду разбиение на непересекающиеся подмножества) в общем случае являются отношениями эквивалентности, поскольку они рефлексивны, симметричны и транзитивны (разбиение индуцирует отношение эквивалентности).

Слова “в общем случае”, относящиеся к этим отношениям, означают некоторую осторожность. Дело в том, что близкими отношениями к отношениям Синонимии и Классификации являются отношения Подобия и Аналогии. Существенная разница между этими отношениями состоит в том, что Подобие и Аналогии не являются транзитивными отношениями. Это связано с тем, что отношение подобия определяется на некотором уровне абстракции. Это проявляется в том, что в процессе поиска подобия выделяются особые свойства объектов (или элементов) и абстрагируются от других свойств, несущественных с нашей точки зрения. И два объекта считаются подобными, если они обладают одинаковыми выделенными свойствами. Отношение Подобия, в свою очередь, тесно переплетается с Аналогией, поскольку в процессе поиска подобия на первый план выходит отношение аналогии, по которому и ищутся общие свойства или устанавливаются различие между этими свойствами. Отсюда вытекает, что между одними и теми же объектами может существовать несколько аналогий и это нарушает свойство транзитивности этого отношения. Однако, часто отношение подобия является рефлексивным и симметричным в смысле одинаковости выделенных свойств.

Отношение Корреляции, Импликации и Оперирования в некотором смысле моделируются (или моделируют) каузуальные отношения, отношения следования (в смысле логического следования).

Отношения Ассоциации, Оперирования и Связывания в некотором смысле являются функциональными отношениями. Поскольку степень тождественности зависит от контекста использования отношений, семантической (смысловой) интерпретации и других обстоятельств, то полной уверенности в правильности принятых предположений не может быть. Это требует дополнительного более глубокого анализа.

Пример обработки ПО “Дискретная математика”

Принимая во внимание сказанное выше, рассмотрим пример построения терминологического тезауруса ПО “Дискретная математика” и на его основе онтографа, представляющего эту ПО.

Структура тезауруса. Тезаурус строится таким образом, что вместе с концептами в нем присутствуют семантические отношения. Семантические отношения – бинарные отношения, виды которых рассматривались выше. Тезаурус в данном случае играет роль Терминологии (аналог TBox) , а отношения, определенные на этой терминологии, можно считать ролями (в некотором смысле аналог ABox), которые присутствуют в ДЛ.

Структура тезауруса имеет вид:

Концепт	Опред-ние	X	Отнош1	Y	X	Отнош2	Y	$\cdots$
Множество	Интуит. опред.	-	-	-	-	-	-	$\cdots$
Элемент x	Объект мн-ва Y	x	R_1	Y	-	-	-	$\cdots$
A1:Акс. экст. X, Y	$X = Y$	X	R_2	Y	A1	R_3	A2	$\cdots$
A2:Акс. объемн. X, Y	$X \subseteq Y \wedge Y \subseteq X$	X	R_2	Y	A2	R_3	A1	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

,

где отношение R_1 – это отношение принадлежности, R_2 – отношение включения, R_3 – отношение синонимии и т.д.

В общем случае построенный тезаурус включает около 700 концептов, основными отношениями на которых являются отношения Синонимии, Часть-целое, Род-вид, Корреляции, Классификации, Принадлежности, Функциональное.

Имея в распоряжении терминологию и семантические отношения, начинается построение онтографа для выбранной ПО. Это построение выполняется с учетом взаимосвязей семантических отношений с математическими отношениями. На первый план здесь выступают отношения частичного порядка и эквивалентности. Отношение частичного порядка позволяет определить структуру онтографа, а отношение эквивалентности наполнение вершин этого графа.

Отношение синонимии играет важную роль в процессе интеграции онтологий, заданных своими онтографами, о чем будет сказано далее.

В результате программная система построения онтографа с учетом сказанного строит размеченный онтограф $G = (V, E)$ (см. рисунок 1 в Приложении 1 в конце статьи).

Дальнейшая работа с построенным онтографом выполняется путем реализации операций алгебры орграфов [20]. Сигнатура операций этой алгебры включает

– нульарную операцию $\Lambda = (\emptyset, \emptyset)$ – "абсолютно пустой граф";

– бинарную операцию введения вершины u в орграф $G = (V, E)$, результатом которой есть орграф $G = (V \cup \{u\}, E)$;

– бинарную операцию введения ребра e в орграф $G = (V, E)$, результатом которой есть орграф $G = (V, E \cup \{e\})$;

– бинарную операцию удаления вершины u из орграфа $G = (V, E)$, результатом которой есть орграф $G = (V \setminus \{u\}, E \setminus \bigcup\limits_{e=(u,v)} e)$;

– бинарную операцию удаления ребра e из орграфа $G = (V, E)$, результатом которой есть орграф $G = (V, E \setminus \{e\})$.

Реализация операции Λ означает заготовку места для онтографа.

Известно, что перечисленные пять операций составляют полное множество операций, с помощью которых может быть реализована любая операция на конечных орграфах [20]. Отсюда следует, что относительно этих операций и носителя, каким является множество конечных орграфов, имеем универсальную алгебру, носителем которой являются конечные орграфы над некоторым универсальным множеством U (из которого берутся отметки вершин). Обычно в качестве множества U выступает множество натуральных чисел N или его декартова степень.

Программная система, реализующая эти операции, приведена в Приложении 1 в конце статьи.

Метаоперации на онтологиях: автоматная интерпретация

Пусть имеется несколько онтологий, которые представляются в виде своих онтографов $G_i = (V_i, E_i)$, $(i = 1, 2 \ldots, k)$. Рассмотрим онтограф $G = (V.E)$, множество вершин V которого представляет множество предметных областей, которые будем называть метаконцептами) а множество ребер E – бинарное отношение между этими предметными областями. Для построения орграфа $G = (V.E)$ применяются операции на онтографах $G_i = (V_i, E_i)$, которые будем называть метаоперациями.

С каждым таким орграфом $G = (V, E)$ будем ассоциировать конечный (вообще говоря) частичный детеминированный автомат без выходов $A = (V, X = V, f, S, F)$, где V – множество состояний, которое также служит входным алфавитом данного автомата, S – подмножество начальных состояний, $F \subseteq V$ – подмножество заключительных состояний (которое, в частности, может быть пустым), а функция переходов данного автомата определяется следующим образом: $f(u, v) = v$ тогда и только тогда, когда $(u, v) \in E$ и не определено в остальных случаях.

Рассмотрим пример представления фрагмента онтологии для ПО "Комбинаторика".

Пусть задана онтология, отражающая некоторую часть этой предметной области (разделы взяты из математической энциклопедии), в виде следующего онтографа:

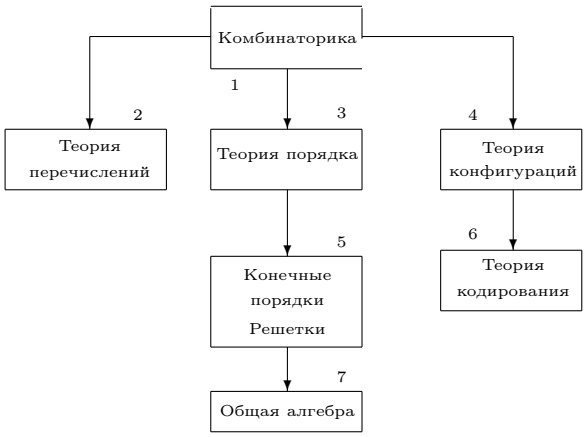


Рис. 6. Онтология C

Конечный автомат, соответствующий данной онтологии, имеет вид $A = (V = \{1,2,3,4,\ 5,6,7\}, X = \{1,2,3,4,5,6,7\}, f, \{1\}, \{7\})$, где f задана таким графом переходов:

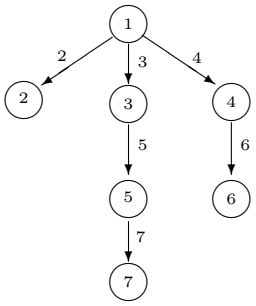


Рис. 7. Автомат A онтологии C

Это значит, что $f(1,2) = 2, f(1,3) = 3, f(1,4) = 4, f(3,5) = 5, f(5,7) = 7, f(4,6) = 6$. Остальные переходы в данном автомате неопределены.

Метаоперации на онтологиях. Представление онтологий в виде конечного автомата без выходов позволяет ввести метаоперации на онтологиях. Основными такими операциями являются:

– объединение – теоретико-множественное объединение множества состояний и множества переходов данных автоматов-аргументов;

– пересечение – теоретико-множественное пересечение множества состояний и множества переходов, пополненное транзитивным замыканием отношения достижимости на автоматах-аргументах;

– конкатенация или умножение двух автоматов – частный случай операции объединения, когда объединение выполняется только по множеству начальных состояний второго автомата;

– итерация – повторяемая конечное число раз операция умножения, применяемая в рамках одной онтологии с целью уточнения и пополнения этой онтологии (с помощью этой операции осуществляется пошаговое уточнение и пополнение онтологий);

– обращение – ориентация в противоположном направлении переходов в автомате, представляющем данную онтологию, т. е. построение функции переходов $g(v,u) = u$ тогда и только тогда, когда $f(u,v) = v$ и неопределено в остальных случаях.

Приведем примеры, иллюстрирующие перечисленные выше операции. Пусть имеем такую онтологию:

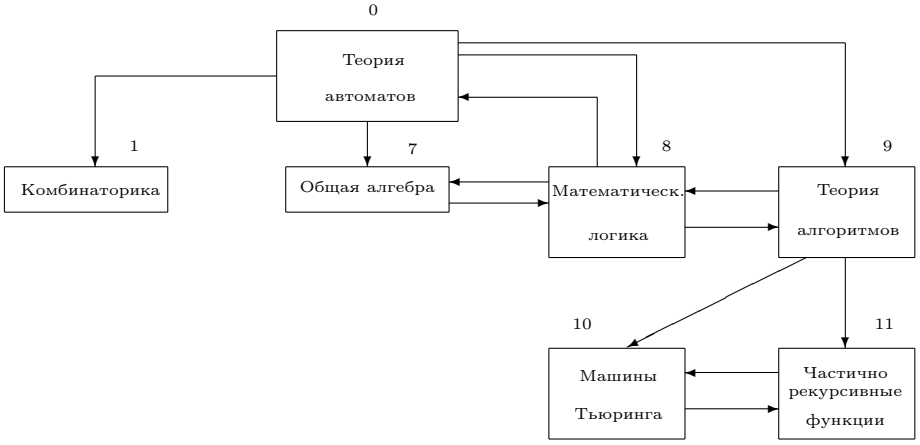


Рис. 8. Онтология O_1

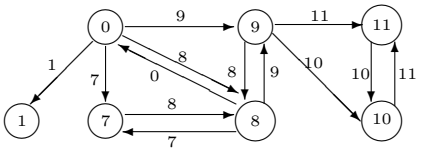


Рис. 9. Автомат A_1 онтологии O_1

Объединение онтологий C и O_1. При реализации операции объединения находятся общие вершины онтографа, по которым и выполняется объединение онтологий. Поскольку такие вершины могут иметь разные метки, то поиск ведется с учетом отношения синонимии. Если в описании вершин находятся синонимичные концепты, то эти вершины считаются одинаковыми и их можно отождествить. После отождествления вершин нужно удалить общие подграфы.

Тогда введенные выше операции дают такие результаты, если их применить к автоматам A и A_1 (автомат A показан на рис. 7):

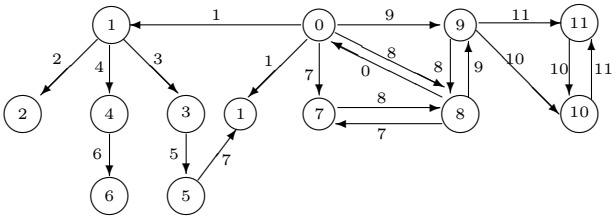


Рис.10. Объединение автоматов A и A_1

Пересечение онтологий C O_1. При выполнении операции пересечения подграфов (как и при выполнении операции объединении) учитывается отношение синонимии. Одинаковыми считаются вершины онтографа, у которых имеются синонимичные концепты. В данном примере пересечение дает такой результат:

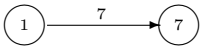


Рис. 11. Автомат $A \cap A_1$

Итерация: пополнение (уточнение) онтологии O_1. Если имеется онтология некоторой подобласти более общей онтологии, то ее пополнение выполняется путем склеивания общих вершин в онтографе. Например, пусть имеем такую онтологию:

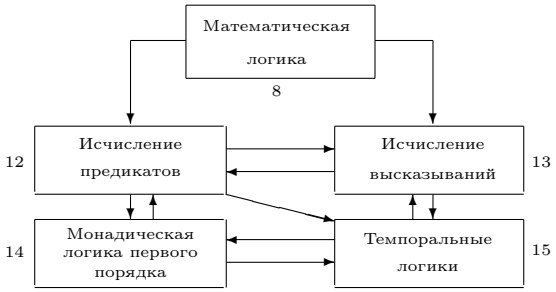


Рис. 12. Онтология O_2

Автомат A_2, соответствующий онтологии O_2, принимает вид:

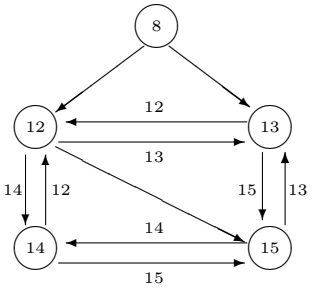


Рис. 13. Автомат A_2 для онтологии O_2

Конкатенируя автоматы A_1 и A_2 по начальному состоянию 8 автомата A_1, получаем автомат, представляющий пополненную (уточненную) онтологию $O_1 * O_2$:

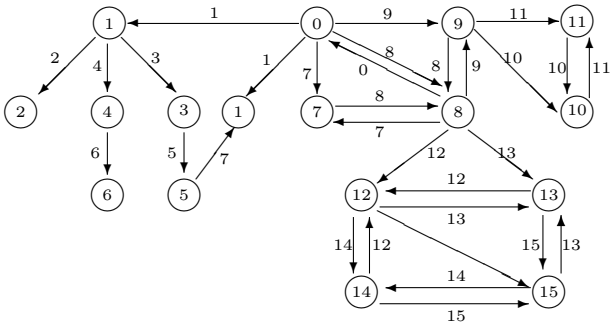


Рис. 14. Пополнение онтологии O_1

Обращение. Применяя эту операцию к A_1, получаем автомат:

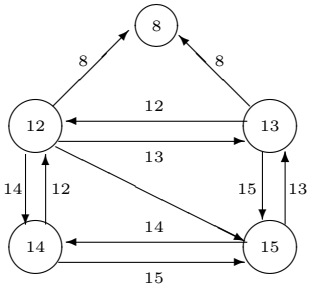


Рис. 15. Обращение автомата онтологии O_2

Приведенное множество операций (в случае надобности) можно расширять по крайней мере в двух направлениях. Одним из них является расширение операций на графах (соединения графов, изоморфного соединения, декартового произведения и т. п.).

Другим направлением является алгебра отношений. Поскольку каждая онтология является представлением некоторой совокупности отношений, то можно вводить операции реляционной алгебры.

Какое из возможных направлений будет выбрано, зависит от практических потребностей использования онтологий. Представленные операции над онтологиями оказываются полезными при анализе, синтезе и манипулировании онтологиями и онтологическими объектами.

Выводы

Подводя итог сказанному, выделим некоторые проблемы, возникающие при построении и работе с онтологиями как концептуальными моделями ПО.

Первая проблема (и возможно основная при работе с онтологиями) связана с автоматизацией процесса извлечения знаний из естественноязыковых текстов. Одним из возможных направлений в решении этой проблемы является некоторая предварительная обработка текста. Такая обработка должна связывать семантические и математические отношения, например с помощью аннотирования текста или другого способа, облегчающего дальнейшую обработку.

Проблемы реализации вышеприведенных операций на онтологиях, связаны с тем, что корректное выполнение этих операций требует создания некоторого общего глоссария предметных областей и понятий, с помощью которого можно было бы однозначно идентифицировать соответствующие объекты. По видимому, эта проблема является не только проблемой на пути реализации введенных операций, но и в некотором смысле общей проблемой на пути построения онтологий и работы с онтологиями. Здесь приходит на помощь отношение синонимии (которое предполагается отношением эквивалентности). Вершины онтографов могут иметь разные названия, но если эти вершины содержат синонимичные объекты, то они склеиваются в одну вершину при реализации таких операций как объединение и пересечение.

Следующая проблема, возникающая при реализации операций, связана с имеющейся иерархией областей и понятий. Дело в том, что в различных онтологиях одни и те же понятия и объекты могут находиться на разных уровнях иерархии и это необходимо учитывать при применении операций. В предлагаемом подходе эта проблема решается с помощью построения транзитивного замыкания отношения достижимости на состояниях автоматов, представляющих данные онтологии. Однако, нет уверенности в том, что этого замыкания достаточно для полного решения проблемы. Здесь необходимы эксперименты с реальными онтологиями и их представлениями.

И еще одна проблема связана с полнотой знаний, имеющихся в представленных онтологиях. Эта проблема является основной в процессе создания программного

и технического обеспечения систем. Здесь же эта проблема состоит в построении в некотором смысле полной онтологической картины той или иной предметной области.

Приложение 1

Инструкция

Программная система строит онтограф некоторой предметной области и выполняет операции на этом онтографе, которые позволяют искать, добавлять и удалять концепты, а также устанавливать связи между концептами. Система имеет интерфейс, который позволяет выводить на экран онтограф.

Если нужно добавить новый концепт, то необходимо нажать на вкладку Concepts

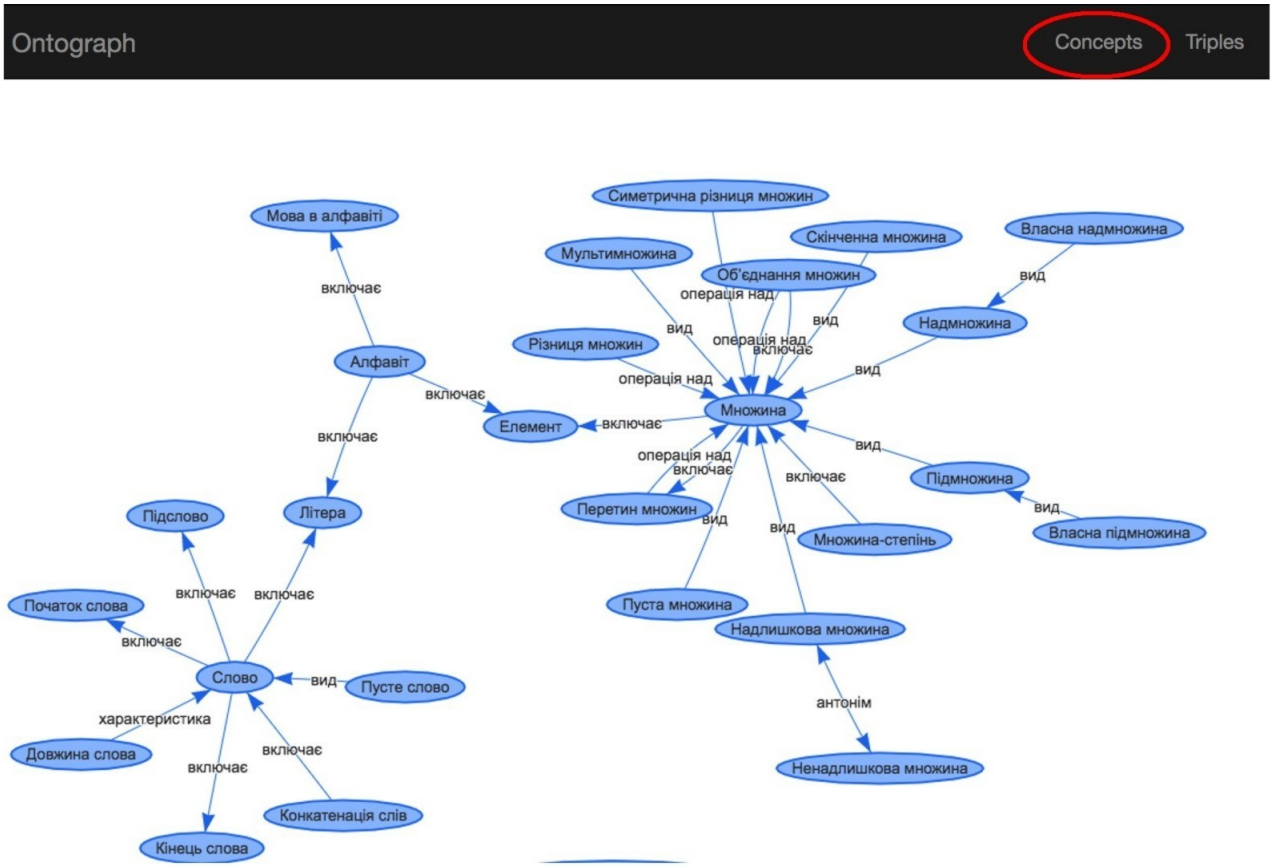


Рис. 1: Пример перехода на вкладку с концептами

После чего появляется окошко:

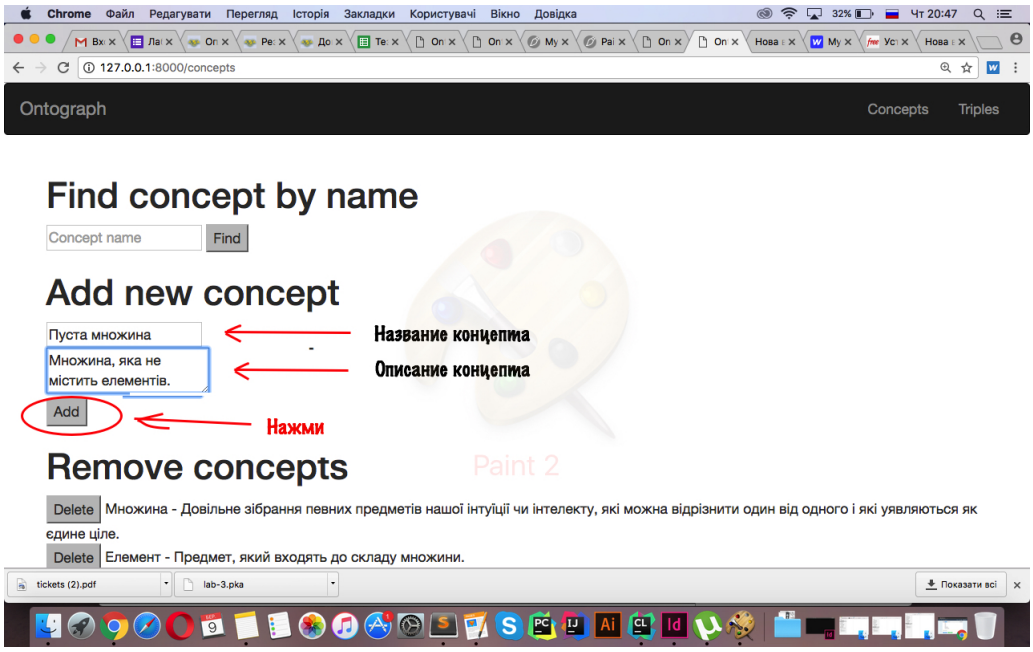


Рис. 2: Пример добавления концепта

Для того что б найти концепт нужно ввести его название в поле и нажать на кнопку <Find>:

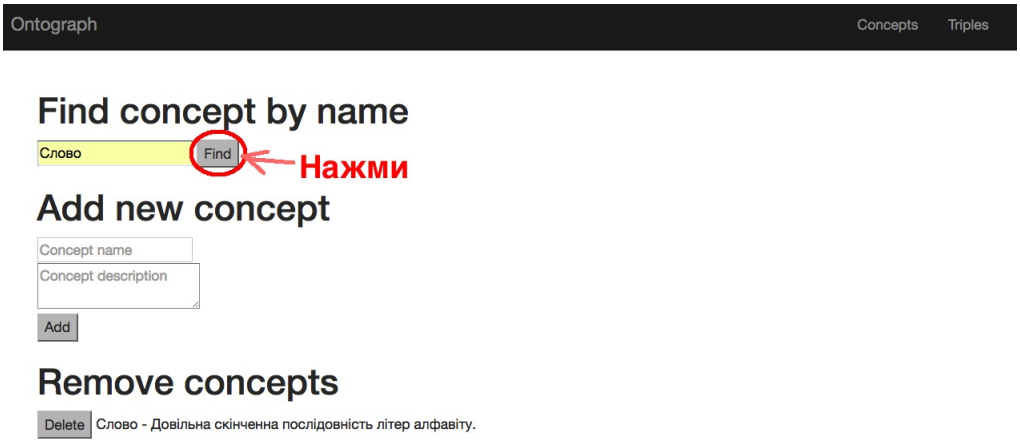


Рис. 3: Пример поиска концепта

Для удаления концепта нужно нажать клавишу <Delete> возле концепта который нужно удалить:

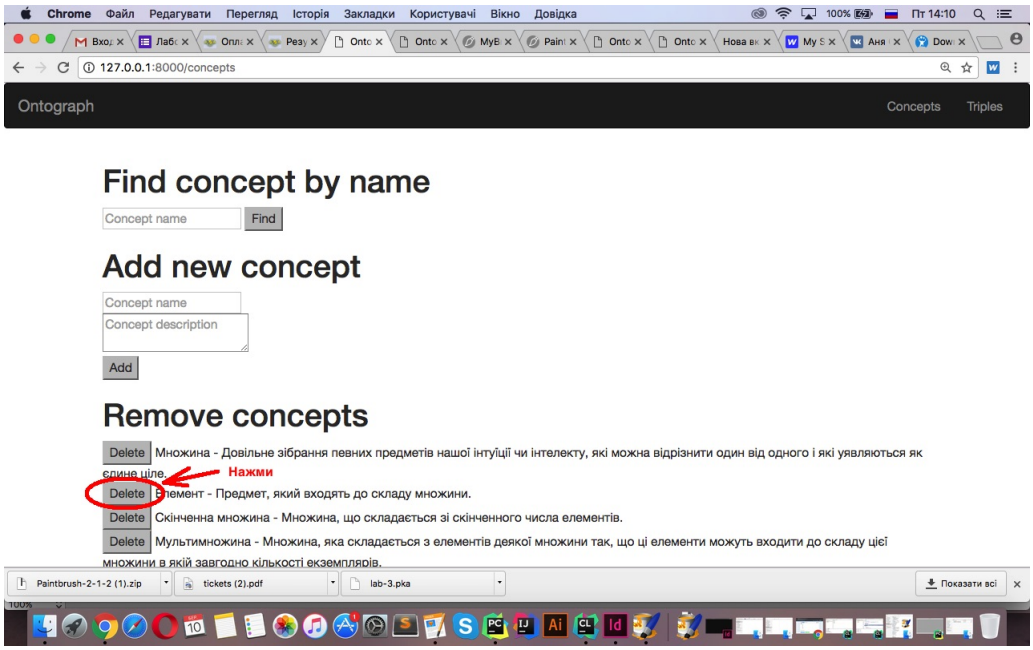


Рис. 4: Пример удаления концепта

Когда нужно установить связь между двумя концептами, то для этого выбираем вкладку <Triples>

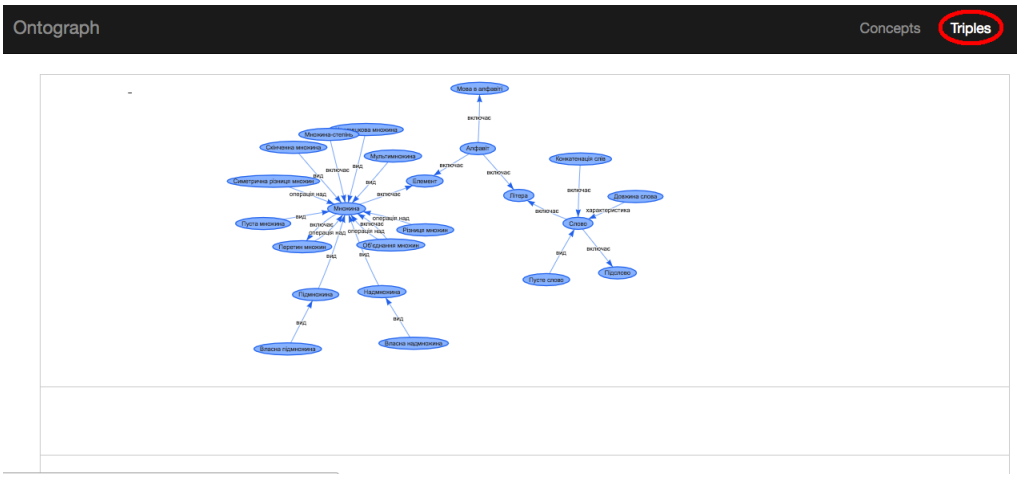


Рис. 5: Пример перехода на вкладку со связями

Затем выбирается первый концепт из списка, вводится имя связи, выбирается второй концепт и нажимается клавиша Add.

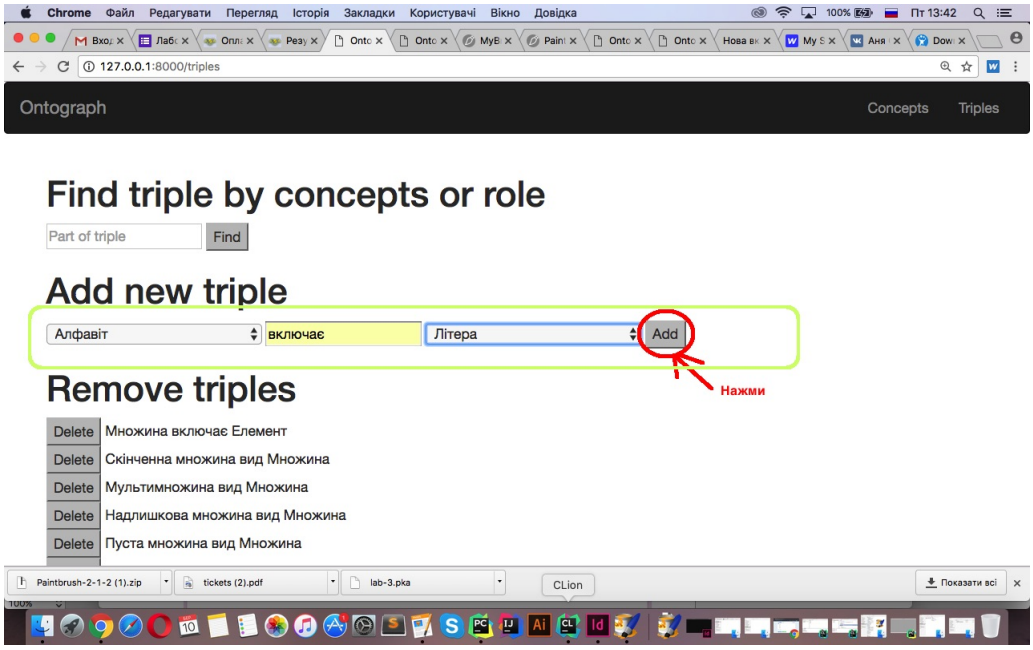


Рис. 6: Пример добавления связи

Имеется возможность найти все концепты по связи. Для этого нужно ввести название связи и нажать клавишу <Find>:

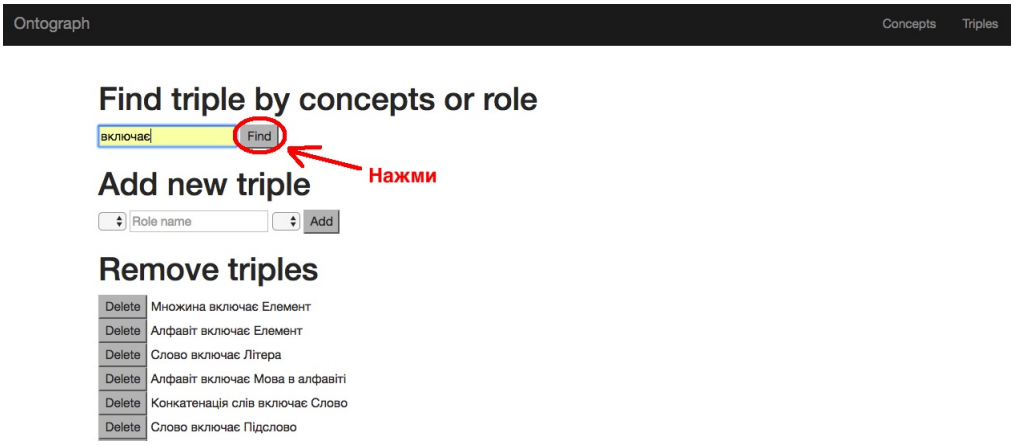


Рис. 7: Пример поиска связи

Для удаления связи нужно нажать клавишу <Delete> возле связи, которую нужно удалить.

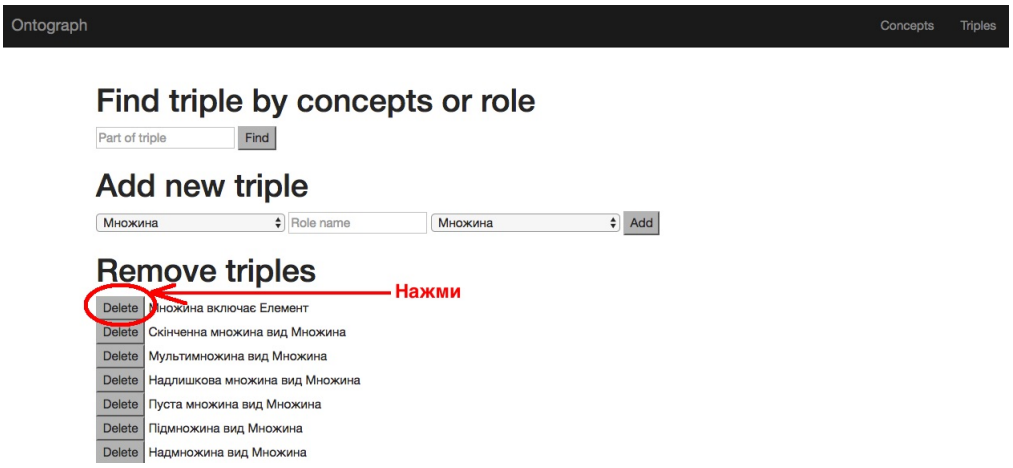


Рис. 8: Пример удаления связи

Для просмотра определения концепта нужно дважды нажать на него левой кнопкой мыши и определение появится под онтографом:

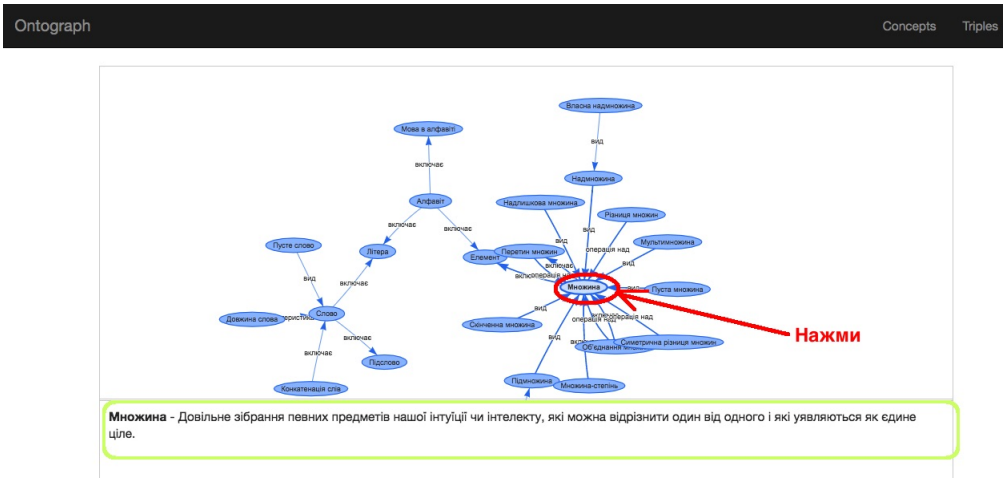


Рис. 9: Пример показа определения

Для просмотра синонимов концепта нужно один раз нажать на него левой кнопкой мыши и синонимы появятся возле данного концепта:

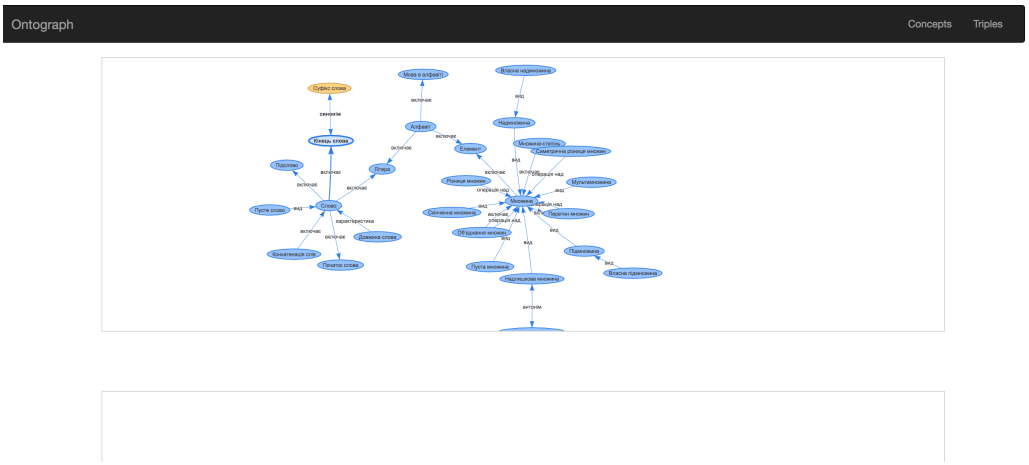


Рис. 10: Пример показа синонимов

Список литературы

[1] А.В. Палагин, С.Л. Крывый, Н.Г. Петренко Знание-ориентированные информационные системы с обработкой естественноязыковых объектов: основы методологии и архитектурно-структурная организация. – УСиМ. – 2009. – № 3. – С. 42 – 55.

[2] Chomsky N. Studies on Semantics in Generative Grammar. The Hague: Mouton, 1972.– Limited preview available in English. – Vista previa limitada disponible en castellano.

[3] Chomsky N. Topics in the Theory of Generative Grammar. Paris: Mouton, 1972. – Limited preview available in English. – Vista previa limitada disponible en castellano.

[4] Fillmore C. J. Frame Semantics . – Linguistics in the morning calm: Selected papers from the SICOL. Seoulю – 1982. – PP. 111 – 137.

[5] Шенк Р. Обработка концептуальной информации. – М.: Энергия, 1980. – 361с.

[6] Шенк Р., Бирнбаум Л., Мей Дж. Новое в зарубежной лингвистике. Компьютерная лингвистика. – М.: Прогресс, 1989. – Вып. 14.

[7] Мельчук И. А. Опыт теории лингвистических моделей “Смысл <=> Текст”. М. – 1974 (2-е изд.). – 1999.

[8] Дарчук Н. П. Комп'ютерне анотування українського тексту: результати і перспективи. К: Освіта України. – 2013. – 544 с.

[9] Леонтьева Н. Н. К теории автоматического понимания естественного языка. М.: Изд.-во Моск. ун-та. ч.2. Семантические словари: состав, структура, методика создания. – 2001. – 40 с.

[10] Baader F., Calvanese D., McGuinness D.L. and other. The Description Logic Handbook. Cambridge: University Press. – 2007. – 601 p.

[11] Пименова М.В. Введение в когнитивную лингвистику. Кемеровою – 2004.

[12] Рубашкин В. Ш. Представление и анализ смысла в интеллектуальных информационных системах. – М.: Наукаю – 1989. – 192 с.

[13] Кулик Б. А. Логика естественных рассуждений. – С.-Петербург: Невский диалект. – 2001. – 127 с.

[14] Brachman R.J. What's in a concept: structural foundations for semantic networks. – In Journal of Man-Machine Studies. – 1977. – 9(2). – PP.127 – 152.

[15] Brachman R.J. Structured inheritance networks. – Research in Natural Language Understanding. – Quarterly Progress Report. – N. 1 – BBN. – Report No. 3742.

[16] Schmidt-Schauβ M. Subsumption in KL-ONE is undesidable. – Proc. of the 1st Int. Conf. on Princieples of Knowledge Representation and Reasoningю – 1989 (KR'89). – PP. 421 – 431.

[17] Nebel B. Termonological reasoning is inherently intractable. – Artificial Inteligence. – 1990. – N. 43 – PP. 235 – 249.

[18] Schild K. A correspondence theory for terminological logics. – Preliminary report. In Proc. of the 12h Int. Joint Conf. on Artificial Intelligence (IJCA'91). – 1991. PP. 466 – 471.

[19] Tarski A. A lattise-theoretical fixpoint theorem and its applications. – Pacific Journal of Mathematics. – 1955. – N. 5 – PP. 285 – 309.

[20] Сергієнко І.В., Кривий С.Л., Провотар О.І. Алгебраїчні аспекти інформаційних технологій. – К., Наукова думка. – 2011. – 399 с.

[21] Lutz C. Complexity of termonological reasoning revisited. – In Proc. of the 6th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR'99). Lecture Notes in Artificial Intelligence. - v.1705 – 1999. – PP. 181 – 200.

[22] Calvanese D., Giacomo G., Lancerini M. Conjunctive query containment in Description Logics with n-ary relations. – In Proc. of the 1997 Description Logic Workshop (DL'97). – 1997. – PP. 5 – 9.

[23] Calvanese D., Giacomo G., Lancerini M. On the decidability of query containment under constraints. – In Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98). – 1998. – PP. 149 – 158.

Сведения об авторах:

Крывый Сергей Лукьянович, доктор физ.-мат. наук, профессор, профессор кафедры информационных систем факультета компьютерных наук и кибернетики Киевского национального университета имени Тараса Шевченка.

Дарчук Наталия Петровна, доктор филологических наук, профессор, профессор кафедры украинского языка и прикладной лингвистики Института филологии Киевского национального университета имени Тараса Шевченка.

Ясенова Ирина Сергеевна, кандидат технических наук, доцент кафедры компьютерных наук факультета информационных технологий Киевского национального университета биоресурсов и природопользования.

Головина Александра Леонидовна, студентка 4-го курса факультета компьютерных наук и кибернетики Киевского национального университета имени Тараса Шевченка.

Соляр Анна Сергеевна, студентка 4-го курса факультета компьютерных наук и кибернетики Киевского национального университета имени Тараса Шевченка.

METHODS AND TOOLS OF KNOWLEDGE REPRESENTATION SYSTEMS

S.L.Kryvyy, N.P.Darchuk, I.S.Yasenova, A.L.Golovina, A.S.Solyar

Abstract: Short review of systems analisys, tools and knowledge representations, building of ontologies by using natural languages text processing are considred. In spacial case presented review on Descriptions Logics, which is now active development. An example representation of concrete objective area by using ontograph with set of operations over ontologies are described.

ACM Classification Keywords: Systems of knowledge representation, Description Logics, ontograph, operations over ontologies

TABLE OF CONTENTS