ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2006, Vol. 35, No. 3 A

APPLICATION OF GENETIC ALGORITHMS TO INCREASE AN OVERALL PERFORMANCE OF NEURAL NETWORKS IN THE DOMAIN OF DATABASE STRUCTURES SYNTHESIS

Eduard Babkin[1], Margarita Petrova[2]

[1]*State University – Higher School of Economics (Nizhny Novgorod branch), Russia, Nizhny Novgorod, Bol. Pecherskaya, 25, 6030155*

[2]*Datavision CIS Nizhny Novgorod Branch, Russia, Nizhny Novgorod, Gagarina prospect, 25-b, 603006*

Abstract. In the given paper a method is described to augment an artificial neural network with corresponding genetic optimization algorithm inside the unified algorithmic framework. Application of this approach is demonstrated for the synthesis of optimum logical structure of a distributed database that has a great impact on an effective design of modern distributed information infrastructure. A neural network algorithm gives local-optimum solutions of that extreme task, while the function of the genetic algorithm within the framework of this problem is the selection among the received local-optimum solutions a global-optimum, satisfying to the given constraints of the problem.

1. Introduction

The problems of the optimum solutions search in various areas of science and engineering render decisive influence on scientific researches. The common point for these problems is that their mathematical model can be formulated as an optimization problem asking to search of such values of controlled variables which provide extreme value for one of the most important technical and economic characteristics of the object, provided other characteristics satisfy to the given set of the technical requirements. The main difficulties of the numerical solution of an extreme task are related to its dimension and the form of the objective function, which generally can be nonlinear, explosive, non differential and multiextreme.

One of the approaches allowing to overcome specified difficulties is the evolution-genetic approach [4] which allows to build ***genetic algorithms*** – algorithms of search of the optimum solutions, on the basis of modeling biological mechanisms of population genetics.

It is a matter of fact that distributed data processing and especially *distributed databases* (DDB) become one of the cornerstones of modern information infrastructure. Their adaptive optimal logical and physical design in presence of dynamical changes of network topology and of operation conditions became an important application domain where researchers face numerous optimization problems [6, 8, 9]. In particular, the synthesis of optimum DDB logical structure is one of the central problems of DDB design. Solution of this problem gives an optimal composition of multiple data elements into several logical record types with subsequent allocation of the record types to certain network hosts. In general, mathematical formulation of this problem includes multiple criteria, moreover the problem itself belongs to a class of nonlinear problems of integer programming and is NP-complete. That's why the practice of DDB design recommends using heuristic methods of designing the optimal logical structures of DDB. *The artificial neural networks* (ANN) approach [1, 5] falls within this category.

In previous work [2], authors developed an original ANN approach based on a Hopfield network that produces a set of locally optimal solutions for the problem of optimum DDB logic structure synthesis. Each of such solutions corresponds to certain values of factors-parameters of an energy function of the neural network. As there are no regular ways of definition of these factor values, it was decided to apply a paradigm of genetic algorithms for evolutionary search of such factor values which is corresponding to the global-optimum solution of the problem.

2. Statement of the Problem

We deal with the simplified task of optimum DDB logic sructure syntesis in presence of single optimized

(minimized) criterion – a total time needed for consecutive processing of a set of DDB users' queries.

In accordance with formalized description [7], all properties of the problem constituents, including initial DDB structure, a set of queries, DDB users, hosts and topology of the computer network (CN), average initial time characteristics, and other significant variables, are shown in Table 1.

Table 1. Formalization of the problem

The name	The designation
Characteristics of a DDB structure	
The set of data elements	$\mathbf{D^G} = \left\{ d_i^g / i = \overline{1, I} \right\}$
The vector of elements lengths	$\boldsymbol{\rho} = \left\{ \rho_i \right\}$
The vector of data element's instantiation numbers	$\boldsymbol{\pi} = \left\{ \pi_i \right\}$
The matrix of a semantic contiguity of data elements	$\mathbf{A^G} = \left\| a_{ii'}^g \right\|$, where $a_{ii'}^g = 1$ if there is a semantic connection between the i-th and i'-th elements, $a_{ii'}^g = 0$ – otherwise
Characteristics of user's queries	
The set of user's queries	$\mathbf{Q} = \left\{ q_p / p = \overline{1, P_0} \right\}$
The matrix of data elements usage during processing of queries	$\mathbf{W^Q} = \left\| w_{pi}^Q \right\|$, where $w_{pi}^Q = 1$ if query p uses (during processing time) the i-th element, $w_{pi}^Q = 0$ – otherwise
The matrix of frequencies of queries used by the users	$\mathbf{\Lambda^Q} = \left\| \xi_{kp}^Q \right\|$, where ξ_{kp}^Q is the frequency of the usage of the p-th query by the user k
Characteristics of the users	
The set of the users	$\mathbf{U} = \left\{ u_k / k = \overline{1, K_0} \right\}$
The matrix of an attachment of the users to hosts in the computing network	$\boldsymbol{\nu} = \left\| \nu_{kr} \right\|$, where $\nu_{kr} = 1$ if the k-th user is attached to host r of computing network, $\nu_{kr} = 0$ – otherwise
The matrix of queries usage by the DDB users	$\mathbf{\Phi^Q} = \left\| \varphi_{kp}^Q \right\|$, where $\varphi_{kp}^Q = 1$ if user k uses query p, $\varphi_{kp}^Q = 0$ – otherwise
The matrix of an attachment of the queries to client hosts	$\mathbf{\Delta^Q} = \left\| \delta_{pr}^Q \right\|$, where $\delta_{pr}^Q = 1$ if $\sum_{k=1}^{k_0} \nu_{kr} \varphi_{kp}^Q \geq 1$; $\delta_{pr}^Q = 0$ if $\sum_{k=1}^{k_0} \nu_{kr} \varphi_{kp}^Q = 0$
Characteristics of the set of computing network's hosts	
The characteristics of the set of computing network's hosts	$\mathbf{N} = \left\{ n_r / r = \overline{1, R_0} \right\}$
The vector of memory volumes on servers of computing network, accessible to the user	$\boldsymbol{\eta}^{\mathbf{EMD}} = \left\{ \eta_r^{EMD} \right\}$, where η_r^{EMD} is the value of accessible external memory on server at host r in the computing network
Average initial time characteristics	
The average time of assembly of the data block at formation of queries' data array	t^{ass}
The average time of one query formation	t^{dis}
The average time of a route choice, establishment of logic (virtual) connections between the client-host ($r1$) and server-host ($r2$)	t_{r1r2}^{ser}
The average time of transfer of one data block (logical record) of query or transaction from the client-host ($r1$) on the server site ($r2$) on the shortest way	t_{r1r2}^{trf}
The average time of access to LDB files and search in them of required logic records	t^{srh}
The average time of processing of one logical record on the server-host ($r2$), dependent of productivity of the server	t_{r2}

For proper formalization of the synthesis task, we define the following variables:

$x_{it} = 1$ if the i-th data element is included into the t-th logical record type; $x_{it} = 0$, otherwise.

$y_{tr} = 1$ if the t-th logical record type is allocated to the server of the r-th host in the computing network; $y_{tr} = 0$, otherwise.

$z^t_{pr2}=1$ if $\sum_{i=1}^{I} y_{tr2} w^Q_{pi} x_{it} \ge 1$; $z^t_{pr2}=0$ if $\sum_{i=1}^{I} y_{tr2} w^Q_{pi} x_{it} = 0$. Variable z^t_{pr2} defines types of logical records used by the p-th query on the server of the $r2$-th host in the computing network.

$z_{pr2}=1$ if $\sum_{t=1}^{T}\sum_{i=1}^{I} y_{tr2} w^Q_{pi} x_{it} \ge 1$; $z_{pr2}=0$ if $\sum_{t=1}^{T}\sum_{i=1}^{I} y_{tr2} w^Q_{pi} x_{it} = 0$. Variable z_{pr2} defines a set of LDB server-hosts to which the p-th query addresses. T is a number of logical records types synthesizing in the solution process.

In the given mathematical framework, the current problem of DDB synthesis might be formulated as follows.

The objective function of the problem:

$$\min_{\{x_{it},y_{tr}\}} \sum_{k=1}^{K_0}\sum_{p=1}^{P_0} \xi^Q_{kp}\cdot\varphi^Q_{kp}\cdot\{\sum_{r1=1}^{R_0} v_{kr1}\cdot[\sum_{r2=1}^{R_0} z_{pr2}\cdot(t^{dis}+t^{ser}_{r1r2}+t^{trf}_{r1r2}\cdot(1+\sum_{t=1}^{t_0} z^t_{pr2}))+t^{ass}]+\sum_{r2=1}^{R_0}\sum_{t=1}^{t_0} z^t_{pr2}\cdot(t^{srh}_{r2}+t_{r2})\} \quad (1)$$

Constraints of the problem:

1) on the number of elements in the logical record type $\sum_{i=1}^{I} x_{it} \le F_t, \forall t=\overline{1,T}$, where F_t − maximum number of elements in the record t;

2) on single elements inclusion in the logical record type $\sum_{t=1}^{T} x_{it}=1, \forall i=\overline{1,I}$;

3) on the length of the formed logical record type $\sum_{i=1}^{I} x_{it} y_{tr} \rho_i \psi_0 \le \theta_{tr}$, where θ_{tr} − as much as possible allowable length of the record t determined by characteristics of the server r;

4) on the total number of synthesized logical records types placed on the server r: $\sum_{t=1}^{T} y_{tr} \le h_r$, where h_r − maximum number of logical records types supported by local database management system of the server-host r;

5) on the volume of accessible external memory of servers for a storage of local databases $\sum_{t=1}^{T}\sum_{i=1}^{I} \psi_0 \rho_i \pi_i x_{it} y_{tr} \le \eta_r^{EMD}$;

6) on the required level of information safety of the system $x_{it}x_{i't}=0$ for given d_i and $d_{i'}$;

7) on the total processing time of operational queries on servers $\sum_{r2=1}^{R_0}\sum_{t=1}^{T} z^t_{pr2}\cdot(t^{srh}_{r2}+t_{r2}) \le T_p$ for given $\mathbf{Q_p} \in \mathbf{Q}$, where T_p − allowable processing time of p operative search.

3. Optimization Approach

In our approach, the solution of the problem (1) is divided into two stages. In the first stage, composition (synthesis) of data elements into some types of logical records is performed. In the second stage, synthesized types of records are no redundantly allocated in computer network.

For each stage, a separate neural network model is constructed. Elaborate representation of the original mathematical statement of our problem in terms of neural network can be found in [2]. For each stage of the problem solution Hopfield neural networks are used.

A paradigm of neural networks. Let we have a matrix of a semantic contiguity of data elements. Let the dimension of the matrix is specify as $n \times n$, where n is a quantity of data elements in DDBS structure. It is necessary to find distribution of data elements on logical records according to the matrix of a semantic contiguity, i.e. to determine splitting set of capacity n on some number ($k \le n$) of subsets. In order to find the solution the ANN is used. This ANN holds neurons in a mode with the large steepness of the characteristic (i.e. the transfer function of a network is "almost threshold":

$$F(x)=\frac{1}{1+e^{-\lambda\cdot NET}}, \text{ where } \lambda \to \infty,$$

where NET — weighed sum of inputs).

Each record is submitted by a row (line) of n neurons. The data elements, which correspond to the neurons with outputs equal to 1, are included in the given logical record. If, for example, $n=5$ (five data elements is present), result of neural network work can be submitted as shown in the Table 2.

Table 2. Example of the result obtained with help of neural network

Record number / Number of data element	1	2	3	4	5
1	1	0	0	0	0
2	0	1	0	0	0
3	0	1	0	0	0
4	0	0	1	1	0
5	0	0	1	1	0

As each data element should be included only into the one synthesized logical record, in each row of the

resulting matrix only one unit should be available. Under such condition constraint on a single data elements inclusion in the record is satisfied automatically:

$$\sum_{t=1}^{T} x_{it} = 1, \forall i = \overline{1, I},$$

where T is a capacity of logical records set, I is a capacity of data elements set, $x_{it} = 1$, if i-th data element is included in t-th logical record, and $x_{it} = 0$, otherwise.

To solve the problem it is offered to use a Hopfield neural network, which has a block diagram shown in Figure 1.

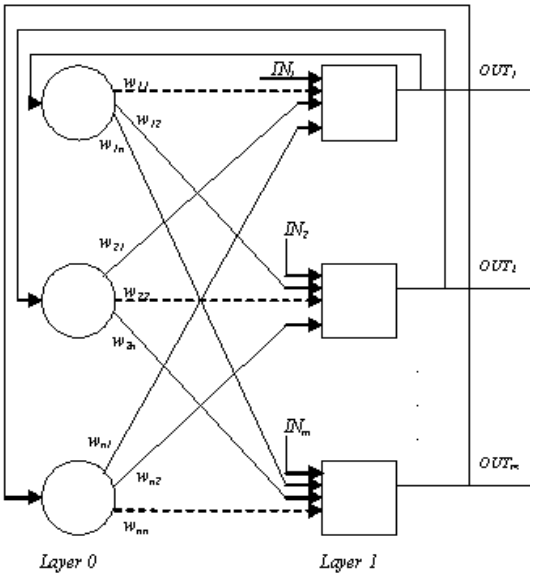

Figure 1. Structure of Hopfield neural network. The dashed lines designate zero weights

The zero layer do not performs any computing function, and only distributes outputs of the network back to its inputs. The output of each neuron is connected to inputs of all other neurons. Each neuron of the first layer calculates the weighed sum of its inputs and returns a signal *NET*, which is transformed with the help of nonlinear function *F* to a signal *OUT*. The submission of entrance vectors is carried out through separate neurons inputs.

If F is a thresholds function with a threshold Th_j, the output of j-th neuron is calculated as

$$NET_j = \underbrace{\sum_{i \neq j} w_{ij} \cdot OUT_i}_{\text{the weighed sum of outputs from other neurons}} + IN_j$$

$$OUT_j = 1 \text{ if } NET_j > Th_j$$

$$OUT_j = 0 \text{ if } NET_j < Th_j$$

$$OUT_j \text{ does not change, if } NET_j = Th_j$$

For our problem: $m = n^2$, where n is a quantity of data elements.

Let each neuron is supplied with two indexes, which corresponds to the number of data element and the number of logical record. For example, $OUT_{xi} = 1$ shows, that the data element x will be included to the i-th logical record. All outputs of the network (OUT_{xi}) have a binary nature, i.e. accept values from the set $\{0,1\}$.

The energy function for the computing network of our problem should satisfy to two constraints:

1. should be small only for those solutions, which have only one unit in each row;

2. should render preference to the solutions with such distribution of data elements into the logical records, which logically follows from a matrix of a semantic contiguity of data elements (i.e. based on the fact that it is more preferable to include into the same logical record, rather than in different, those data elements, which are semantic connected according to the given matrix).

The first requirement is satisfied by introduction of the following energy, consisting of two terms:

$$E = \frac{A_1}{2} \cdot \sum_x \sum_i \sum_{j \neq i} OUT_{xi} \cdot OUT_{xj} + \frac{C_1}{2} \cdot \left[\left(\sum_x \sum_i OUT_{xi} \right) - n \right]^2,$$

where A_1, C_1 are some constants. Therefore the following constraints are satisfied.

1. The threefold sum is equal to zero only if each row (the data element) contains no more than one unit.

2. The second term of the energy function is equal to zero only if the resulting matrix contains exactly n units.

The second requirement is satisfied with the help of addition of the following member to the energy function:

$$\frac{B_1}{2} \cdot \sum_i \sum_x \sum_{y \neq x} OUT_{xi} \cdot \left(OUT_{yi} - a^g_{xy} \right)^2,$$

where $A^g = \left\{ a^g_{xy} \right\}$, $x, y = \overline{1, n}$ is the matrix of a semantic contiguity of data elements, B_1 is some constant.

For enough large values A_1, C_1 low-energy states will represent the allowable fragmentations of data elements set on logical records, and the large values of B_1 guarantee, that the most preferable fragmentation will be found.

So, the energy function for our task looks like

$$E = \frac{A_1}{2}\cdot\sum_x\sum_i\sum_{j\neq i} OUT_{xi}\cdot OUT_{xj} +$$
$$+\frac{B_1}{2}\cdot\sum_i\sum_x\sum_{y\neq x} OUT_{xi}\cdot\left(OUT_{yi} - a_{xy}^g\right)^2 +$$
$$+\frac{C_1}{2}\cdot\left[\left(\sum_x\sum_i OUT_{xi}\right) - n\right]^2 =$$
$$= -\frac{1}{2}\cdot\sum_i\sum_j\sum_x\sum_y [-A_1\cdot\delta_{xy}\cdot\left(1-\delta_{ij}\right) + \quad (2)$$
$$+B_1\cdot\delta_{ij}\cdot\left(1-\delta_{xy}\right)\cdot\left(2\cdot a_{xy}^g - 1\right) - C_1]\cdot OUT_{xi}\cdot OUT_{yj} +$$
$$+\sum_i\sum_x[\frac{B_1}{2}\cdot\sum_{y\neq x}\left(a_{xy}^g\right)^2]\cdot OUT_{xi} -$$
$$-\sum_i\sum_x[C_1\cdot n]\cdot OUT_{xi} + \frac{C_1}{2}\cdot n^2,$$

where δ_{ij} is the Cronecker delta.

For the complete description of Hopfield neural network for the first stage of the problem solution it is necessary to expand the energy function (2) taking into account the constraints 1) and 6). The constraint on the required level of information safety of the system: $x_{it}x_{i't} = 0$ for given d_i and $d_{i'}$, means that data elements i and i' are incompatible, i.e. they can not be included into the one logical record. Let this constraint is given by the matrix $\mathbf{incomp\_gr} = \left\|incomp\_gr_{ij}\right\|, (i, j = \overline{1,n})$, where

$$incomp\_gr_{xy} = \begin{cases} 1 \text{ if data element } x \text{ is compatible} \\ \quad \text{with data element } y \\ 0, \text{ otherwise or if } x = y \end{cases}.$$

Each of the constraints considered above is included into the energy function of the network as a separate term with weight coefficient. And as a result of constraints translation into the terms of neural network the following energy function of network for the first stage of the problem's solution is provided:

1 stage

$$E = \frac{A_1}{2}\cdot\sum_x\sum_i\sum_{j\neq i} OUT_{xi}\cdot OUT_{xj} + \frac{B_1}{2}\cdot\sum_i\sum_x\sum_{y\neq x} OUT_{xi}\cdot(OUT_{yi} -$$
$$-a_{xy}^g)^2 + \frac{C_1}{2}\cdot\left[\left(\sum_x\sum_i OUT_{xi}\right) - n\right]^2 + \frac{D_1}{2}\cdot\sum_i\left(\frac{1}{F_i}\cdot\sum_x OUT_{xi}\right) +$$
$$+\frac{E_1}{2}\cdot\sum_i\sum_x\sum_y OUT_{xi}\cdot OUT_{yi}\cdot\left[incomp\_gr_{xy} + incomp\_gr_{yx}\right] =$$
$$= -\frac{1}{2}\cdot\sum_i\sum_j\sum_x\sum_y [-A_1\cdot\delta_{xy}\cdot\left(1-\delta_{ij}\right) + B_1\cdot\delta_{ij}\cdot\left(1-\delta_{xy}\right)\cdot\left(2\cdot a_{xy}^g - 1\right) - \quad (3)$$
$$-C_1 - E_1\cdot\delta_{ij}\cdot(incomp\_gr_{xy} + incomp\_gr_{yx})]\cdot OUT_{xi}\cdot OUT_{yj} +$$
$$+\sum_i\sum_x[\frac{B_1}{2}\cdot\sum_{y\neq x}\left(a_{xy}^g\right)^2 + \frac{D_1}{2\cdot F_i}]\cdot OUT_{xi} - \sum_i\sum_x[C_1\cdot n]\cdot OUT_{xi} + \frac{C_1}{2}\cdot n^2$$

The result of the first stage of the problem solution is the matrix $\mathbf{X} = \left\|x_{ij}\right\|, (i = \overline{1,n};\ j = \overline{1,T})$ of distribution of data elements into logical records, where

$$x_{it} = \begin{cases} 1 \text{ if data element } i \text{ is included} \\ \quad \text{into the logical record } t \\ 0, \text{ otherwise} \end{cases}.$$

At the second stage irredundant allocation of the synthesized types of logical records in the computer network is performed taking into account the constraints 3), 4), 5) and 7).

Each host of the computer network is submitted by a row of T neurons. Each of these neurons corresponds to the logical record type. The logical records types, which correspond to the neurons with outputs equal to 1, are placed on the given host of the computer network. For the representation of the second stage of the problem in the terms of neural network it is required $T\times R_0$ neurons, where T is a number of logical records types synthesized in the solution process, R_0 is a number of hosts in the computer network.

Each of the constraints is included into the energy function of the network as a separate term with weight coefficient. And as a result of constraints translation into the terms of neural network the following energy function of network for the second stage of the problem's solution is provided:

2 stage

$$E = \frac{A_2}{2}\cdot\sum_{t=1}^{T}\sum_{r_1=1}^{R_0}\sum_{r_2=1}^{R_0} OUT_{tr_1}\cdot OUT_{tr_2} + \frac{B_2}{2}\cdot\left[\left(\sum_{t=1}^{T}\sum_{r=1}^{R_0} OUT_{tr}\right) - T\right]^2 +$$
$$+\frac{C_2}{2}\cdot\sum_{t=1}^{T}\sum_{r=1}^{R_0}\left(\frac{OUT_{tr}\cdot\psi_0}{\theta_{tr}}\cdot\sum_{i=1}^{n}\left(x_{it}\cdot\rho_i\right)\right) + \frac{D_2}{2}\cdot\sum_{r=1}^{R_0}(\frac{1}{h_r}\cdot\sum_{t=1}^{T} OUT_{tr}) +$$
$$+\frac{E_2}{2}\cdot\sum_{r=1}^{R_0}\left(\frac{\psi_0}{\eta_r^{EMD}}\cdot\sum_{t=1}^{T}\sum_{i=1}^{n}\left(\rho_i\cdot\pi_i\cdot x_{it}\cdot OUT_{tr}\right)\right) +$$
$$+\frac{F_2}{2}\cdot\sum_{p=1}^{P_0}\left(\frac{1}{T_p}\cdot\sum_{r=1}^{R_0}\sum_{t=1}^{T} OUT_{tr}\cdot SN_{pt}\cdot\left(t_r^{srh} + t_r\right)\right) =$$
$$= -\frac{1}{2}\cdot\sum_{r_1=1}^{R_0}\sum_{r_2=1}^{R_0}\sum_{t_1=1}^{T}\sum_{t_2=1}^{T} [-A_2\cdot\delta_{t_1t_2}\cdot\left(1-\delta_{r_1r_2}\right) - \quad (4)$$
$$-B_2]\cdot OUT_{t_1r_1}\cdot OUT_{t_2r_2} + \sum_{r_1=1}^{R_0}\sum_{t_1=1}^{T}[\frac{C_2\cdot\psi_0}{2\cdot\theta_{t_1r_1}}\cdot\sum_{i=1}^{n}\left(x_{it_1}\cdot\rho_i\right) +$$
$$+\frac{D_2}{2\cdot h_{r_1}} + \frac{E_2\cdot\psi_0}{2\cdot\eta_{r_1}^{EMD}}\cdot\sum_{i=1}^{n}\left(\rho_i\cdot\pi_i\cdot x_{it_1}\right) +$$
$$+\frac{F_2\cdot\left(t_{r_1}^n + t_{r_1}\right)}{2}\cdot\sum_{p=1}^{P_0}\left(\frac{SN_{pt_1}}{T_p}\right)]\cdot OUT_{t_1r_1} -$$
$$-\sum_{r_1=1}^{R_0}\sum_{t_1=1}^{T}[B_2\cdot T]\cdot OUT_{t_1r_1} + \frac{B_2}{2}\cdot T^2,$$

where SN_{pt} is a normed sum, i.e.

$$SN_{pt} = \begin{cases} 1 \text{ if } \sum_{i=1}^{I} w_{pi}^{Q} \cdot x_{it} \geq 1 \\ 0 \text{ if } \sum_{i=1}^{I} w_{pi}^{Q} \cdot x_{it} = 0 \end{cases}.$$

The result of the second stage of the problem solution is the matrix $\mathbf{Y} = \|y_{tr}\|, (t = \overline{1,T}; r = \overline{1,R_0})$ of irredundant allocation of logical records types in the computer network, where

$$y_{tr} = \begin{cases} 1 \text{ if logical record type } i \text{ is placed} \\ \quad \text{on the host } r \\ 0, \text{ otherwise} \end{cases}.$$

The developed neural network algorithm accepts as input data both intrinsic domain parameters (characteristics of initial DDB structure, characteristics of computer network hosts, characteristics of users queries, characteristics of users and constraints of a problem) and two vectors of weight coefficients $\overrightarrow{\mathbf{C_1}} = \left(A_1,B_1,C_1,D_1,E_1\right)$ and $\overrightarrow{\mathbf{C_2}} = \left(A_2,B_2,C_2,D_2,E_2,F_2\right)$. These coefficients play a role of artificial parameters in the energy functions of neural networks (2), (3) in the first and second stages, respectively. Proper selection of weight coefficients is vitally important for our problem because, given ***concrete vectors*** $\overrightarrow{\mathbf{C_1}}$ and $\overrightarrow{\mathbf{C_2}}$, the neural network algorithm produces a ***quasi-optimum*** problem solution for the used objective function and constraints, and the obtained quasi-optimum solution does not obviously coincide with the globally optimum solution of the problem.

Unfortunately, there is no a regular way to define elements of the vectors. It is only known that the constraint $A_1,B_1,C_1,D_1,E_1,A_2,B_2,C_2,D_2,E_2,F_2 > 0$ should be satisfied. Thus, we have the following situation: the neural algorithm is capable to produce a set of pairs "result-matrices" (a matrix of data elements composition on logical records types and a matrix of their allocation on hosts in computer network). Each such pair corresponds to "instantiations" of vectors $\overrightarrow{\mathbf{C_1}}$, $\overrightarrow{\mathbf{C_2}}$ (i.e. a set of concrete values of elements of these vectors). It is necessary to find out such "instantiations" of vectors $\overrightarrow{\mathbf{C_1}}$ and $\overrightarrow{\mathbf{C_2}}$ (to look for the values of their elements) at which the neural algorithm would produce the global optimum solution according to the objective function and constraints of the problem. We denote such "instantiations" of vectors $\overrightarrow{\mathbf{C_1}}$ and $\overrightarrow{\mathbf{C_2}}$ by $\overrightarrow{\mathbf{C_{1\,optimum}}}$ and $\overrightarrow{\mathbf{C_{2\,optimum}}}$. Genetic algorithms are offered to find $\overrightarrow{\mathbf{C_{1\,optimum}}}$ and $\overrightarrow{\mathbf{C_{2\,optimum}}}$ vectors.

A paradigm of genetic algorithms. Let's consider the general principles of genetic optimization algorithms that were applied in our case. In accordance with [4], given a complex objective function of several variables, such on algorithm should solve an optimization problem, i.e. to find such variable's values at which the value of the function is maximum (minimum).

This problem is solved on the basis of biological evolutionary approaches. Let's consider each variant (the set of variable's values) as an ***individual*** (a point in the area of the feasible solutions), and a value of suitability function for this variant − as ***suitability*** of the given individual. Then during evolution suitability of individuals will grow, so, better (more and more optimum) variants will occur. If we stop evolution at some moment and chose the best variant, it is rather possible to obtain a good solution of the problem.

The ***genetic algorithm (GA)*** can be represented as a sequence of managing actions and operations, simulating evolutionary processes on the basis of analogues of genetic inheritance mechanisms and natural selection.

Thus, the biological terminology in the simplified kind is kept. An ***individual*** is a variant of the problems' solution or a point in the set of the feasible solutions. A ***gene*** is a carrier of one hereditary attribute. A ***chromosome*** is an association of genes, i.e. all hereditary attributes. A ***locus*** is a serial number of a gene in the chromosome. An ***allele*** is some defined value of an individuals' gene. A ***genotype*** is a certain value of the chromosome, its filling. A ***phenotype*** is a degree individuals' suitability to an environment, it influences on individuals' survival and its posterity. Thus, a phenotype is a value of suitability.

The genetic algorithms represent a heuristic approach rather than uniform exact algorithms. Any concrete problem requires substantial accommodation of the general scheme of GA described by the sequence of steps given in [4].

Step 0. Form of an initial population. N is the number of individuals in a population.

Step 1. Form of new generation. K is the number of children. This parameter should be defined outside of the algorithm. At the same time the inequality K > N should be satisfied, otherwise some reproduction steps will be impossible.

1.1. The choice of a parental pair.

1.2. The choice of a crossing system.

1.3. The reproduction process.

Steps 1.1. − 1.3. are repeated so long as the size of new generation will not grow to K. At the next iteration in ***1.1.***, children, obtained on the previous iteration, can act as the parents.

Step 2. Mutation.

Step 3. Form of reproduction group, i.e. a set of individuals, which can included into the next generation.

Step 4. Natural selection: only N individuals from the reproduction group will survive.

Step 5. Checking of stop conditions for the population evolution process. If the stop conditions are not satisfied, then generations are changed and all calculations for a population of the next generation repeat from ***Step 1***.

In Figure 2, one of variants of the genetic algorithm structure is shown.

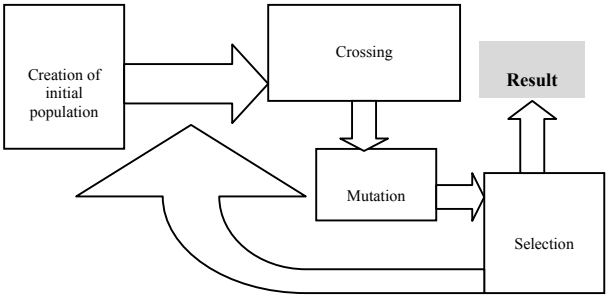

Figure 2. Structure of a genetic algorithm

The search of an optimum solution in genetic algorithms is carried out by a direct manipulation with a set of the several feasible solutions forming a population, each of which is coded in a binary code. Since our neural algorithm produces various solutions for the problem of DDB logic structure synthesis at various pairs of "instantiations" of vectors $\overrightarrow{\mathbf{C_1}}$ and $\overrightarrow{\mathbf{C_2}}$, the genetic algorithm can be used as the intellectual tool of movement to the optimum solution of the problem.

Genetic algorithm as a framework for the neural network algorithm. Let's develop a model of application of genetic algorithms to the given problem.

The individual in our case is a point $\overrightarrow{\mathbf{C_i}}, i \in \{1, 2\}$ in the k-dimensional space (if $i = 1$, then $k = 5$, if $i = 2$, then $k = 6$, i.e. the dimension depends on the number of constraints for the first and second stages of the problem's solution). The number of genes is equal to $k = 6$ for each individual. The chromosome is a consecutive association of all genes.

To construct a genotype, the following symbolical model was developed: each gene is represented by a string of 32 bits. Such a representation turns out with the help of two operations: 1) translation of value of a gene in a decimal notation in binary (at addition of the senior categories of zero, if it is necessary), 2) translation of a gene in a binary notation into a Gray code.

Thus, the genotype of each individual is represented as a $32 \cdot k$-bit sequence, where the first, the second, etc. element of the sequence is Gray code of the decimal value of the first, the second, etc. element of a vector $\overrightarrow{\mathbf{C_i}}$ respectively.

From genetic algorithm's point of view the neural algorithm is used as a "black box" carrying out mapping of individual $A_i - B_i - C_i - \ldots$ to the corresponding matrix $\mathbf{X} = \left\| x_{ij} \right\|, (i = \overline{1,n};\ j = \overline{1,T})$ or $\mathbf{Y} = \left\| y_{ij} \right\|, (i = \overline{1,T};\ j = \overline{1,R_0})$ (it depends of the number of a stage of the problem's solution), on which suitability of $A_i - B_i - C_i - \ldots$ individual is defined: $A_i - B_i - C_i - \ldots \rightarrow \{\mathbf{X}, \mathbf{Y}\}$.

Suitability function of the genetic algorithm for the 1st stage of the problem's solution. To define the individual's suitability, the following suitability function was developed.

$$F_{suit.} = 1/(1 + \sum_x \sum_i \sum_{j \neq i} OUT_{xi} \cdot OUT_{xj} + \sum_i \sum_x \sum_{y \neq x} OUT_{xi} \cdot \left(OUT_{yi} - a_{xy}^{g}\right)^2 + \\ + \left(\left(\sum_x \sum_i OUT_{xi}\right) - n\right)^2 + \sum_i \left\lfloor \frac{1}{F_i} \cdot \sum_x OUT_{xi} \right\rfloor + \sum_i \sum_x \sum_y OUT_{xi} \cdot OUT_{yi} \cdot \\ \cdot \left(incomp\_gr_{xy} + incomp\_gr_{yx}\right)) \qquad \textbf{(5)}$$

Here square brackets mean a taking of the integer part of the division (i.e. the floor function). In a denominator all terms, except the first and the third one, respond for satisfying of problem's constraints and are considered first of all. If the sum of these four terms is equal to zero, it means that all constraints are satisfied. The third term shows as far as well the given solution takes into account semantic contiguity of data elements. The less its value the better semantic contiguity is taken into account. In compare with the constraints of a problem the complete account of semantic contiguity is not necessity, it is desirable. Generally, conditions of semantic contiguity and constraints of a problem can be incompatible. The priority is always given back to satisfy of problem's constraints. Thus, at the first stage, the genetic algorithm allows to select from a set of the solutions satisfying to constraints, such solutions which take into account a semantic contiguity of data elements in the best way.

Suitability function of the genetic algorithm for the 2nd stage of the problem's solution. To define the individual's suitability, the following suitability function was designed:

$$F_{suit.} = 1/(1 + \sum_{t=1}^{T} \sum_{r_1=1}^{R_0} \sum_{r_2=1}^{R_0} OUT_{tr_1} \cdot OUT_{tr_2} + \left(\left(\sum_{t=1}^{T} \sum_{r=1}^{R_0} OUT_{tr}\right) - T\right)^2 + \\ + \sum_{t=1}^{T} \sum_{r=1}^{R_0} \left\lfloor \frac{OUT_{tr} \cdot \psi_0}{\theta_{tr}} \cdot \sum_{i=1}^{n} \left(x_{it} \cdot \rho_i\right) \right\rfloor + \sum_{r=1}^{R_0} \left\lfloor \frac{1}{h_r} \cdot \sum_{t=1}^{T} OUT_{tr} \right\rfloor + \\ + \sum_{r=1}^{R_0} \left\lfloor \frac{\psi_0}{\eta_r^{EMD}} \cdot \sum_{t=1}^{T} \sum_{i=1}^{n} \left(\rho_i \cdot \pi_i \cdot x_{it} \cdot OUT_{tr}\right) \right\rfloor + \\ + \sum_{p=1}^{P_0} [\frac{1}{T_p} \cdot \sum_{r=1}^{R_0} \sum_{t=1}^{T} OUT_{tr} \cdot SN_{pt} \cdot \left(t_r^{srh} + t_r\right)] + \\ + \sum_{k=1}^{K_0} \sum_{p=1}^{P_0} \xi_{kp}^{Q} \cdot \varphi_{kp}^{Q} \cdot \{ \sum_{r_1=1}^{R_0} v_{kr_1} \cdot (\sum_{r_2=1}^{R_0} z_{pr_2} \cdot (t^{dis} + t_{r_1 r_2}^{ser} + t_{r_1 r_2}^{trf} \cdot \\ \cdot (1 + \sum_{t=1}^{T} z_{pr_2}^{t})) + t^{ass}) + \sum_{r_2=1}^{R_0} \sum_{t=1}^{T} z_{pr_2}^{t} \cdot (t_{r_2}^{srh} + t_{r_2}) \}) \qquad \textbf{(6)}$$

Here square brackets mean a taking of the integer part of division. In a denominator, the first six terms after unit respond for satisfying the constraints of a

problem and are considered first of all. If the sum of these six terms is equal to zero, it means that all constraints were satisfied. In this case, the given individual-solution is placed in a newly formed population and the value of the objective function of our problem (1), which is last term in the denominator, is calculated. If the sum of the first, following after unity, six terms is not equal to zero, then the given individual is excluded from consideration and, hence, does not get in a new population. In such case, among all solutions satisfying the constraints (a part of suitability function, checking satisfaction of constraints, is equal to zero), solutions, which correspond to a minimum of problem objective function, are selected. Taking into account the equation (6) it becomes obvious that the smaller value of the objective function of the problem the closer the suitability function of the genetic algorithm to unity value. Based on the logic of the problem, one can see that $F_{suit.} \in (0;1]$, and the best value of individual's suitability is $\max\{F_{suit.}\} = 1$. The suitability value, calculated in this way, "defines the further destiny" of the individual representing $A_i - B_i - C_i - \ldots$.

The solution of the problem with the help of genetic algorithms gives us the point $\vec{C}_{ioptimum}$ for each of two stages. Thus application of such vector as input data for neural algorithm will ensure finding of the optimum result-matrix pair both for the first stage and for the second stage of the considered DDB design problem.

In the given work, the idea of genetic algorithm's application as a framework for central neural algorithm, both for first and for the second stages of the problem's solution is offered. The function of this framework is the selection among the solutions produced by the neural algorithm (generally local optimum) and appropriate to various values of coefficients, global-optimum solutions at the given constraints.

The block diagram of the developed algorithm for the optimum DDB logic structure synthesis task is shown in Fig. 3. A neural algorithm is a nucleus of each stage. Neural network algorithm performs the following functions: constructing of a Hopfield neural network with definition of weights, biases and neurons' thresholds, and getting the locally optimal solutions for various values of weight coefficients in networks' energy functions with help of the created neural network. The values of weight coefficients $A_i, B_i, C_i, \ldots$, i.e. elements of $\vec{C_i}$ vectors, are generated by the genetic algorithm − a "shell" of the central neural network algorithm. As the input data, the NN−GA-algorithm takes the characteristics of initial DDB structure, characteristic of hosts in calculation network, characteristic of the users' queries, characteristic of users and constraints of the problem. The output data consist of the solution selected by the genetic algorithm (i.e. the matrix of data elements combined into logical records types and the matrix of allocation of logical records types on the hosts in the network) and graphic information allowing to control the process of problem solution. The diagrams of populations' structure and suitability of individuals from the next population, histograms of individuals' distribution on levels of their suitability are the parts of the output graphical information.

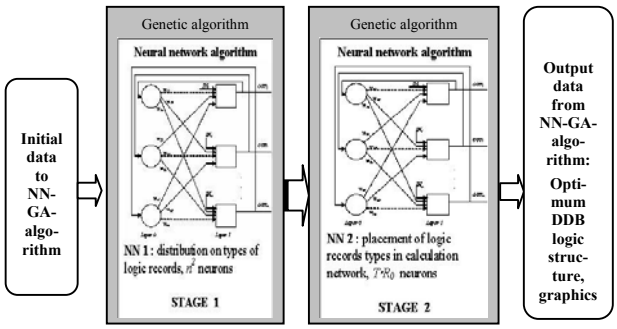

Figure 3. The block diagram of the implemented NN-GA-algorithm

Inside the described NN-GA algorithm, it is possible to allocate the following roles. The NN-part responds for satisfying constraints of a problem and for the taking into account of semantic contiguity of data elements in the best way. As generally conditions of a semantic contiguity and the constraints of the problem can be incompatible, the priority is given to the fulfillment of the constraints. The GA-part responds for minimization of problem's objective function, i.e. provides an optimality of the solution.

4. Algorithm testbed software inplementation

During implementation of the NN-GA algorithm an Object-Oriented library of C++ classes was constructed, allowing to simulate dynamics of the Hopfield neural networks and to use these networks for the solution of the problem. An object-oriented model of genetic algorithms used as a framework for neural network's classes was also developed.

The result of the problem's solution is the following:

1) the vector $\vec{C_1}$ and the matrix of data elements combination on logical records types (stage 1);

2) the vector $\vec{C_2}$ and the matrix of allocation of logical records types on hosts in a computer network (stage 2).

Features of genetic algorithm's implementation. In the current implementation of the genetic algorithm, the following mechanisms are used.

1) Gray code for representation of a symbolical model of a genotype. This code is constructed of binary digits in such a manner that the neighbor numbers in it differ always only in one digit.

2) Formation of an initial population on a feno-type.

3) A way of a pair choice − punmixing. Punmixing is a casual crossing. The only one constraint imposed on the choice of parental pair − this parental pair should not be repeated.

4) Genetic operations − simple crossover and dot mutation.

5) A general way of parental group formation, i.e. both children and parents are included into the reproduction group.

6) Strict natural selection.

The algorithm stops if during several populations the champion on suitability level does not vary. The limit on the number of such populations is submitted to the algorithm as a parameter. The second criterion of algorithm's stopping is as follows: if suitability of at least one individual of the current population is equal to1. In this case, the output is the vector $\vec{C}_i$ corresponding to individual whose suitability is equal to 1.

5. Results of experiments and their analysis

During testing, the optimization problem for various configurations of constraints was solved. The optimality of the solution was judge by value of the objective function of the problem on the obtained solution.

The large number of experiments with cardinal number (potency, power) of data elements set ranging from 3 up to 40 have allowed to synthesize optimum types of logical structures of the distributed database for various constraints.

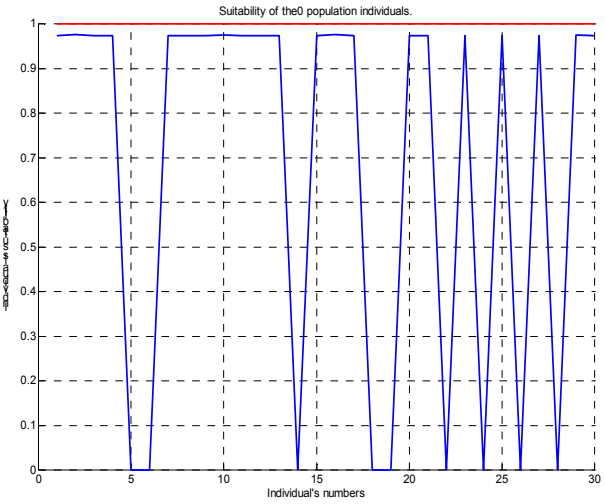

Figure 4. Suitability of the individuals in the initial population

To show abilities of genetic algorithms in selection of the optimum solutions, we present in Figures 4 up 6 the diagrams of suitability values change on the initial and first two populations during the solution of the second stage of synthesis of the optimum DDB logic structure with the number of elements equal to 20 (Figures 4-6). The horizontal lines on suitability level equal to 1 in the figures show the best value of individuals' suitability, i.e. $\max\{F_{suit.}\} = 1$. The graphics below these lines mean the suitability of individuals from initial (Figure 4), first (Figure 5) and second (Figure 6) populations.

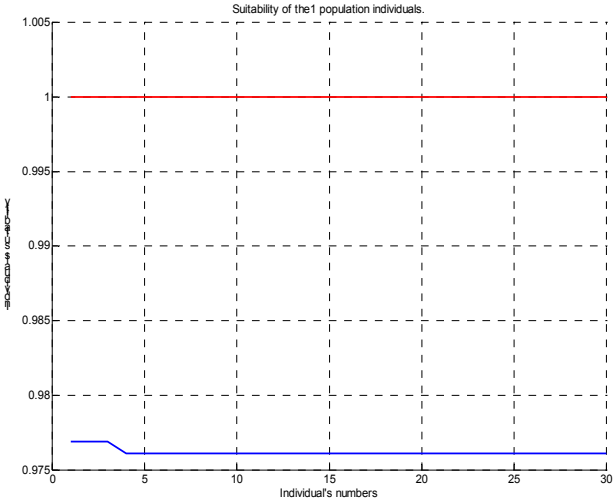

Figure 5. Suitability of the individuals in the first population

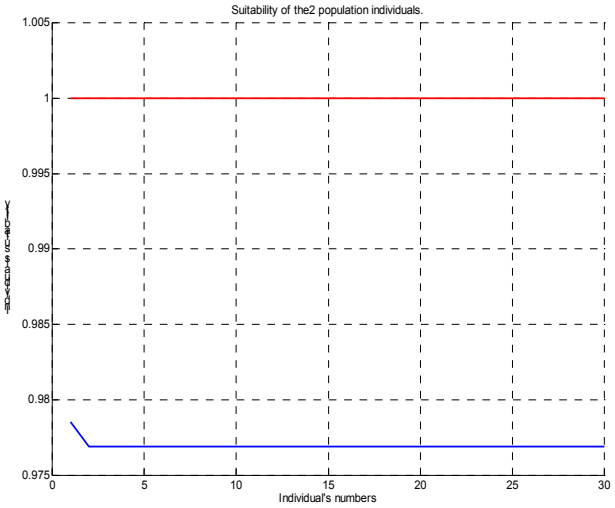

Figure 6. Suitability of the individuals in the second population

The quality of solutions obtained using the neural algorithm was compared with the quality of solutions produced by a branch-and-bound method proposed in [7] to solve the problem with similar constraints; for each of the solutions, percent of a deviation was calculated by the following formula

$$P_{deviation} = \left(\frac{\text{the value of objective function (NN-GA-algorithm)}}{\text{the value of objective function (branch-and-bound method)}} - 1 \right) \cdot 100\% \quad (7)$$

The average percent of a deviations received as a result of testing on tasks of various size is $P_{deviation_{average}} = -6.44\%$.

In [5, 8, 10] the Hopfield networks approaches for the solution of NP-complete optimization problems, in particular for the solution of the traveling salesman problem, are described. The main problem of the approach described in these papers − uncertainty in a choice of coefficients of energy function of a network. Besides that, it is mentioned that regular methods of definition of these coefficient's values do not exist. In work [3], the useful recommendations for allocation of areas of acceptable coefficient's values are given. In such a situation, the use of genetic algorithms is a preferable alternative. Such an approach, though being heuristic, allows to move in a correct direction during search of the most suitable values of coefficients.

The algorithm, proposed in the current paper, remains efficient even after removal of the genetic framework from the neural networks. However, with "included" genetic framework the quality of solutions is considerably higher, since in the case of significant changes of the input characteristics even such robust systems as neural network can give results far from the optimum. It is possible to improve results by correctly picking up coefficients of the energy function. The rules of such selection are offered by genetic algorithms. These rules do not carry casual character and work in such a way that it is possible to approach better the value of the energy function of a problem towards the optimum.

Authors are grateful for all reviewers for valuable comments and to the State University - Higher School of Economics for financial support of this work in the framework of the Scientific Fund grant # 06-04-0005.

References

[1] **M.A. Arbib (ed.).** Handbook of brain theory and neural networks. 2*ed*. *MIT*, 2003.

[2] **E. Babkin, M. Karpunina.** Towards application of neural networks for optimal synthesis of distributed database systems. *Proc. of the* 12th *IEEE Intrl. Conference on Electronics, Circuits, and Systems, satellite workshop Modeling, Computation and Services, Gammarth, Tunisia*, 2005, ISBN: 9973-61-100-4, 486-490.

[3] **G. Feng, C. Douligeris.** Using Hopfield Networks to Solve Traveling Salesman Problems Based on Stable State Analysis Technique. *IEEE-INNS-ENNS International Joint Conference on Neural Networks* (IJCNN'00), *Vol*.6, 2000.

[4] **J.H. Holland.** Adaptation in Natural and Artificial Systems. *University of Michigan Press, Ann Arbor*, (1975).

[5] **J.J. Hopfield, D.W. Tank.** Neural computation of decisions in optimization problems. *Biological Cybernetics*52, 1985, 141-152.

[6] **D. Kossmann.** The State of the Art in Distributed Query Processing. *ACM Computing Surveys, Vol*. 32, No.4, *December* 2000.

[7] **V.V. Kulba, S.S. Kovalevskiy, S.A. Kosyachenko, V.O. Sirotyuck.** Theoretical backgrounds of designing optimum structures of the distributed databases . 1999, 660.

[8] **S.R. Madden, M. J.Franklin, J.M. Hellerstein, W. Hong.** TinyDB: An Acquisitional Query Processing System for Sensor Networks. *ACM Transactions on Database Systems, Vol*.30, *No*.1, *March* 2005.

[9] **Z. Mao, C. Douligeris.** A Distributed Database Architecture for Global Roaming in Next-Generation Mobile Networks. *IEEE/ACM Transactions on Networking, Vol*.12, *No*.1, *February* 2004.

[10] **D.Kossmann.** The State of the Art in Distributed Query Processing. *ACM Computing Surveys, Vol*.32, *No*.4, *December* 2000.

[11] **C.Peterson, B. Söderberg.** Artificial neural networks and combinatorial optimization problems in Local Search in Combinatorial Optimization. *Eds. E.H.L. Aarts and J.K. Lenstra, New York- John Wiley & Sons*, 1997.

Received August 2006.