Using Genetic Algorithms for Building Metrics of Collaborative Systems

Cristian CIUREA
Economic Informatics Department,
Academy of Economic Studies, Bucharest, Romania
cristian.ciurea@ie.ase.ro

The paper objective is to reveal the importance of genetic algorithms in building robust metrics of collaborative systems. The main types of collaborative systems in economy are presented and some characteristics of genetic algorithms are described. A genetic algorithm was implemented in order to determine the local maximum and minimum points of the relative complexity function associated to a collaborative banking system. The intelligent collaborative systems based on genetic algorithms, representing the new generation of collaborative systems, are analyzed and the implementation of auto-adaptive interfaces in a banking application is described.
Keywords: *Collaborative Systems, Genetic Algorithms, Metrics, Banking, Auto-Adaptive Interfaces*

1 Collaborative systems in economy

The knowledge-based society requires increasing at a very high level the human-computer interactions in order to solve certain problems of the citizens. The implementation of collaborative informatics systems in different activity fields facilitates the resolution of many citizens' problems.
The field of application classifies collaborative systems in followings categories:

- collaborative educational systems;
- collaborative banking systems;
- collaborative planning systems;
- collaborative tagging systems;
- collaborative writing systems;
- collaborative medical systems.

Collaborative educational systems are applied in the educational field and have the objective to evaluate and increase the performance of the learning process. The virtual campus represents the most encountered type of collaborative educational system.
In a *collaborative banking system*, the customer interaction with the bank is done in several ways, some of them requiring minimal effort from the customer. On the banks websites there are applications and simulators used to calculate loans or deposits interest rates. The application for the calculation of a loan interest rate will display to the user the monthly interest rate and the value of the monthly fees. These calculations are carried out also in the bank branches, but this involves that the customer goes to the bank.
In a collaborative banking system, the customer enters on the online application, realizes an exchange of information, and knows which loan he can take, without interacting with the human factor. Based on some coordinates that the customer provides, the bank automatically builds his credit file, querying different databases: from his job, from the tax authorities, from the police. By applying an own decision algorithm, the bank will respond, automatically, in real time, to the customer if he can take a loan or not.
Collaborative planning systems present the most appropriate way to tackle certain kind of planning problems, especially those where a centralized solving is unfeasible. The main goal is to efficiently obtain a good collective plan [1].
Collaborative tagging systems provide a new means of organizing and sharing resources. A collaborative tagging system allows arbitrary users to assign tags freely to any documents available on the web [2].
The major benefits of *collaborative writing systems* include reducing task completion time, reducing errors, getting different

viewpoints and skills, and obtaining an accurate text. A collaborative writing system is modeled as follows: it considers *n* sites with each site owning a copy of shared data. When a site performs an update, it generates a corresponding operation. [3]

Collaborative medical systems are those in which modern communication technologies allow doctors from around the world to work on the same patient. In a chirurgical operation each person from the group of doctors has distinct roles [4].

Each type of these systems must be analyzed and evaluated from both points of view: qualitative and quantitative. In order to accomplish these assessments, metrics are implemented for each system, to measure different quality characteristics. Using genetic algorithms for building and testing metrics of collaborative systems ensure the high accuracy of these assessments.

The paper is structured in five sections, in the second one being presented some characteristics of genetic algorithms. Section 3 describe the implementation of a genetic algorithm in a banking application to determine the local maximum and minimum points of the relative complexity function associated to a collaborative banking system. In section 4 are presented the intelligent collaborative systems based on genetic algorithms and the implementation of auto-adaptive interfaces in a banking application. Section 5 ends with conclusions and future work.

2 Key elements of genetic algorithms

A genetic algorithm is an informatics model simulating the biological evolutionary model in order to solve problems of optimization or search. It includes a set of individual elements, represented in the form of binary strings, representing the population, and a set of biological operators defined on the population. With the help of operators, genetic algorithms manipulate strings, evaluated according to an objective function, searching for better solutions [5].

Genetic algorithms are a class of models inspired by the evolutionary theory. These algorithms encode the possible solutions of some specific problems in a data structure of chromosome type and apply recombination operators to these structures in order to preserve useful information [6].

Genetic algorithms are part of the heuristics algorithms, they being applied successfully to problems that do not allow polynomial time algorithms. They are represented by intelligent applications that are able to solve certain problems by using a concept of species evolution. Genetic algorithms are inspired from the nature, specifically from the way in which the species are improved from genetic recombination.

The idea of genetic algorithms is to represent the solutions of a problem in the form of chromosomes and to work, at each step, with a fixed number of chromosomes, forming a population. Thus, is attempting to improve the population of chromosomes within the time available, within the meaning of closeness as much as the optimal solution.

Genetic algorithms offer a model that simulates biological evolution and natural selection. Due to the advantage of parallel processing, genetic algorithms are used in informatics, as well in solving transport problems and optimization problems.

By using genetic algorithms, the complexity of neural network structure reduces and required fewer neurons to solve a problem.

In [7] is considered that genetic algorithms represent a method inspired from biology that is used to optimize non-linear functions with multiple local minimum or maximum points. The principle underlying this class of optimization algorithms is taken from nature and aims the natural development.

Genetic algorithms search for solutions of a problem mimicking specific mechanisms of natural evolution. In order to find the solution, a population is build, each individual representing possible solutions of the problem to be solved. On the population are applied changes inspired by the natural evolution, such as selection, crossing and mutation. In the process of evolution, individuals are obtained that are increasingly better adapted to the environment.

3 Building and validating metrics of collaborative systems with the help of genetic algorithms

The technology for building metrics of collaborative systems assumes that collaborative systems are influenced by the factors if_1, if_2, ..., if_n that have associated a number of variables. A graph of influences is realized and a factorial analysis of variables is performed. A correlation between the influence factors is realized. It follows a list of dependent variables and a list of independent variables and analytical expressions are built. The performance criteria are specified and results a technology that allows building the optimal metric in relation with the influencing factors and performance criteria.

The capacity to integrate indicators in metrics of collaborative systems is given by the ratio between the number of processing made to ensure the integration of indicators and the number of metrics obtained:

$$CIM = \frac{NPI}{NMO},$$

where:

CIM – the capacity to integrate indicators in metrics;

NPI – the number of processing made to ensure the integration of indicators;

NMO – the number of metrics obtained.

The integration is achieved by adding indicators and maintaining the metric homogeneity, or by restructuring indicators and reducing complexity.

The integration process leads to indicators that are compensatory. When defining the indicators, an analysis of properties must be performed and should be taken into account the extra information that is desired. Thus, by using indicators with simple structure, the same quality of information is obtained, but with a little effort.

The integration of indicators obtained by neural networks and genetic algorithms in metrics of collaborative systems help to automate the current operations performed in a collaborative system, but also to provide strategic, tactical and operational information required in the decision-making process.

The implementation of a genetic algorithm starts with a population of chromosomes, randomly chosen. These structures are evaluated and reproductive facilities are allocated so that those chromosomes, which represent the better solution to the target problem to have more chances to reproduce themselves than those chromosomes which are less good solutions. Defining a good solution is made compared with the current population.

Genetic algorithms are very useful in the implementation and validation of metrics, especially for those complex indicators and functions for which is difficult to find a solution. Such complex metrics are encountered in the collaborative systems field, where the number of components and links between them is big.

The relative complexity of a collaborative banking system, *CR*, is determined according to the relationship:

$$CR = \frac{\sum_{i=1}^{n}(f_i * \log_2 f_i)}{(\sum_{i=1}^{n} f_i) * \log_2(\sum_{i=1}^{n} f_i)},$$

where:

f_i – the weight i associated to a quality characteristic of the collaborative banking system, with the property that $f_1 + f_2 + \ldots + f_n = 1$ and $f_i \in R$.

In order to determine the local minimum and maximum values of the *CR(x)* function, a genetic algorithm has been implemented within Collaborative Multicash Servicedesk – CMS application.

The goal of the Collaborative Multicash Servicedesk application is to store and process the customers' requests regarding the functionality of the electronic payments service, solved by the Multicash helpdesk analysts within a commercial bank in Romania.

The source code has been written in the C# programming language, being implemented the classes *GeneticAlgorithm*, *Genome* and *GenomeComparer*, as follows [8]:

```
public delegate double
GAFunction(double[] values);

public class GA
{
static private GAFunction getFitness;
public GAFunction FitnessFunction
{
};
}

GA ga = new GA(0.8,0.05,100,2000,2);

ga.FitnessFunction = new
GAFunction(theActualFunction);

public sealed class GenomeComparer:
IComparer
{
public GenomeComparer()
{
}

public int Compare(object a, object b)
{
if (!(a is Genome) || !(b is Genome))
throw new ArgumentException("The object
is not of type Genome");
if (((Genome) a).Fitness > ((Genome)
b).Fitness)
return 1;
else if (((Genome) a).Fitness ==
((Genome) b).Fitness)
return 0;
else
return -1;
}
}
```

The *Genome* class was built as a simple container and a matrix whose elements lie in the range 0-1 gives the basic structure of the container. The algorithm will use these values, and the user will expand them to the scale of needs. Because mutations occur on the genome, in the *Genome* class there is the *Mutate()* method, implemented as follows:

```
public void Mutate()
{
for (int pos = 0; pos < length; pos++)
{
if (random.NextDouble() < mutationRate)
genes[pos] = (genes[pos] +
random.NextDouble()) / 2.0;
}
}
```

The *CrossOver()* method needs access to private data of the genome, so it is a class member function in the *Genome* class, this method exits being two child objects of the *Genome* class.

```
public void CrossOver(ref Genome g2, out
Genome child1, out Genome child2)
{
int pos = (int)(random.NextDouble() *
(double)length);
child1 = new Genome(length, false);
child2 = new Genome(length, false);
for (int i = 0; i < length; i++)
{
if (i < pos)
{
child1.m_genes[i] = genes[i];
child2.m_genes[i] = g2.genes[i];
}
else
{
child1.genes[i] = g2.genes[i];
child2.genes[i] = genes[i];
}
}
}
```

The genetic algorithm implemented requires the execution of the following steps:

- creating a new population of chromosomes;
- selecting the best two individuals from the population and cross them for obtaining children;
- replacing the old population with a new one;
- resumption of the previous steps until it reaches the optimal solution of the problem.

Figure 1 shows how to calculate the extreme points of local maximum and minimum of *CR(x)* function using the genetic algorithm implemented within the CMS application:

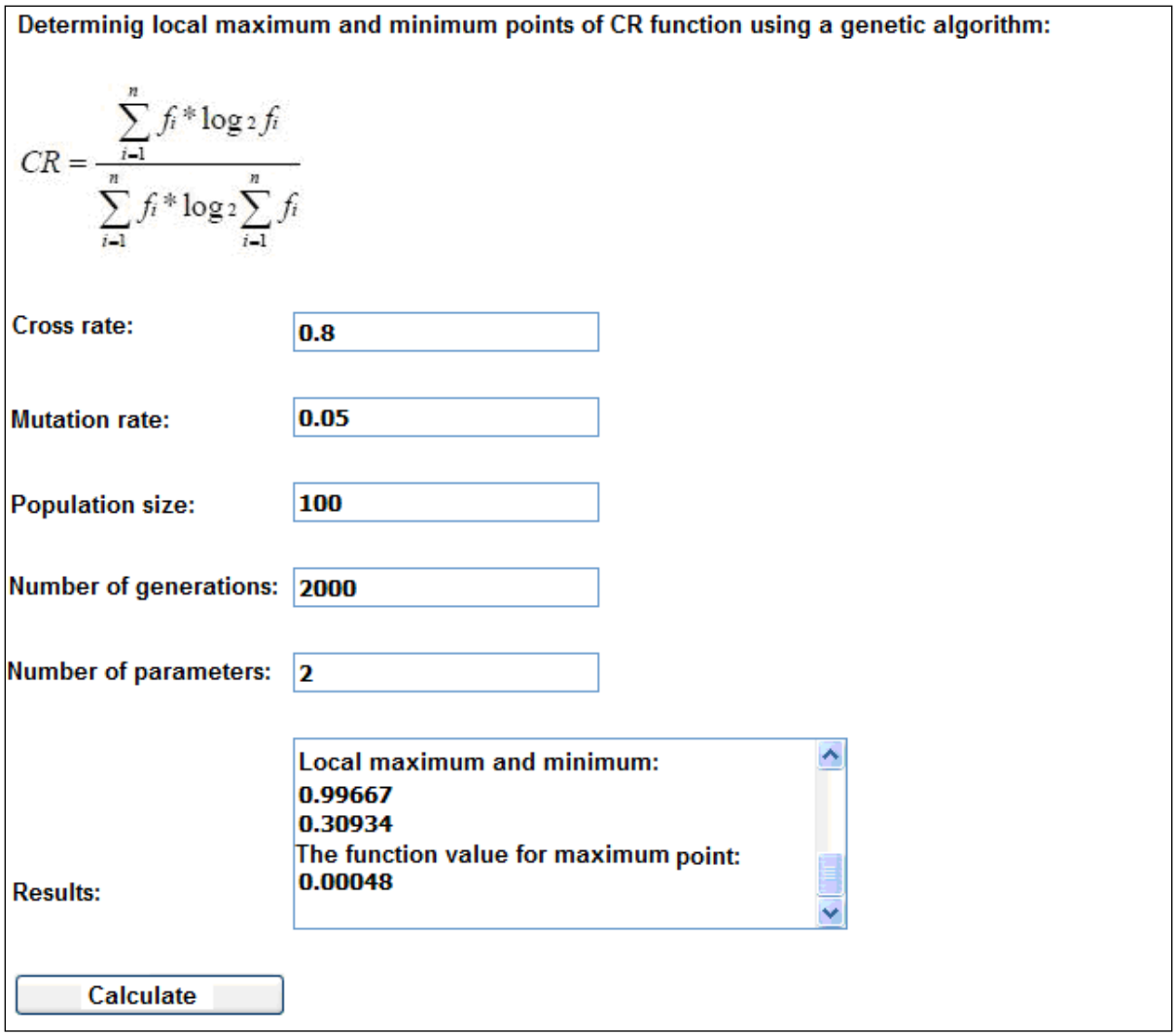

Fig. 1. Determining the local maximum and minimum points of CR(x) function

There are created different genomes with elements in the range 0-1, but the values are improved at each generation, so that the final values obtained to be much closer to the maximum and minimum points of the function.

In Table 1 are presented the values obtained in three successive generations of the genetic algorithm.

Table 1. The values obtained with the genetic algorithm

Generation	Maximum point	Minimum point	Function value in the maximum point
1	0.94760	0.68387	-0.00785
2	0.98459	0.55019	-0.00225
3	0.99667	0.30934	-0.00048

Figure 2 shows the values of maximum and minimum points in the three generations of the algorithm. As seen from the graphic, the maximum point tends near to 1 and the minimum point decrease near to 0.36.

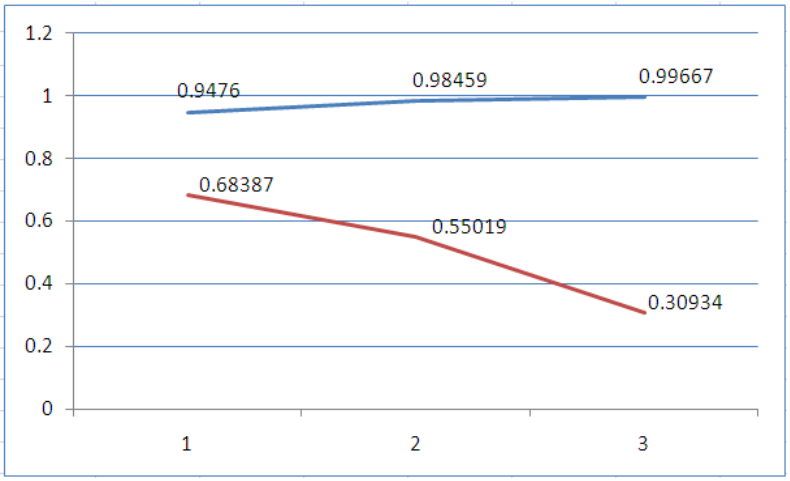

Fig. 2. The values of maximum and minimum points

The accurate values of the maximum and minimum point, obtained with mathematical methods, are 1 for the maximum and 0.36 for the minimum point.

The local minimum point value, i.e. *0.30934*, obtained with the help of the genetic algorithm, shows that the relative complexity of the collaborative banking system is minimal for $f_i = \frac{1}{e}, i = 1..n$, where $f_i \in R$.

Table 2 presents the differences between the accurate values and the genetic algorithm values for the local extreme points of the CR(x) function.

Table 2. The differences between values

Generation	Maximum point			Minimum point		
	GA	Math	Difference	GA	Math	Difference
1	0.94760	1	-0.05240	0.68387	0.36	0.32387
2	0.98459	1	-0.01541	0.55019	0.36	0.19019
3	0.99667	1	-0.00333	0.30934	0.36	-0.50660

The objective of the genetic algorithm implemented is to detect increasingly better chromosomes, until reaching a value of the report between the assessment associated to a solution and the average evaluation of all the solutions about that is known to be optimal, or until the genetic algorithm cannot make a difference.

4 Intelligent collaborative systems based on genetic algorithms

The evolution of collaborative systems and the genetic algorithms' impact on the nature of realized processing are the starting points in developing a new category of intelligent systems with collaborative character.

The specific mechanism of these intelligent systems is inspired by biological systems in that it encourages the candidate solutions able to solve a problem and penalizes solutions without success. In this way are obtained, after several generations, very good solutions for complex optimization problems with a large number of parameters.

Due to their inherently parallel structure, the intelligent collaborative systems based on genetic algorithms have proven efficient in solving the problems of search and identification of structures and specific relationships within bulky databases and knowledge bases.

Because they can learn complex relationships and structures inside incomplete sets of information and knowledge, the intelligent collaborative systems can adapt to changes in the environments in which they operate, and can be used as tools for discovery new knowledge.

A special category of intelligent collaborative systems is represented by the collaborative

informatics systems implemented inside certain banks.

Another implementation of genetic algorithms in the CMS application is represented by the automatic assignment of customer requests to the helpdesk analysts [8]. Each analyst is specialized in better solving certain problems, but this does not means that an analyst cannot solve any problem assigned. Every analyst registered in the CMS application has associated a vocabulary of keywords, which is updated periodically. When a customer creates a new request in the application, the field with problem description is analyzed in order to extract the most significant terms of the problem claimed. These terms are compared with the keywords of each analyst in order to determine the most appropriate analyst to solve the problem. When the algorithm finds the best solution, the request is automatically assigned to the appropriate analyst.

Figure 3 give an example of automatic request assignment of customer problem to the helpdesk analyst.

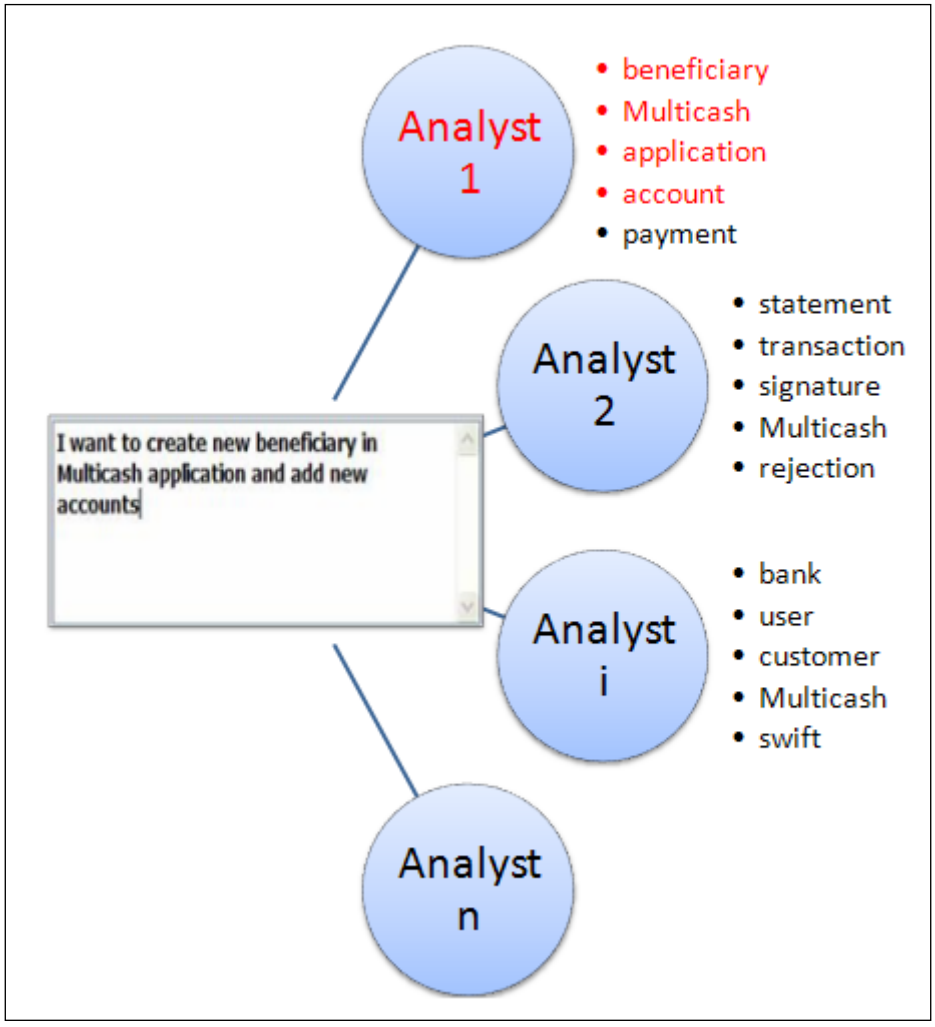

Fig. 3. The automatic assignment of customer requests

As seen in Figure 3, the most appropriate analyst to solve the problem claimed by the customer is the Analyst 1, because its vocabulary of keywords contains many common terms. If the algorithm finds two or more analysts that are eligible to be selected, the winner analyst will be the one that has the most common words in its vocabulary.

The algorithm for automatic requests assignment has the followings steps:

Step 1. The customer enters on the CMS application and registers the new

request by selecting the category and filling the description field.

Step 2. The process of keywords extraction is performed and text measurements are realized.

Step 3. It is calculated the percent of keywords matching for each analyst vocabulary; if there are no common words, the algorithm ends and a notification is sent to all the analysts in order to assign manually the request.

Step 4. The most appropriate analyst is selected based on common words identified; the selection process is performed based on frequency of common words.

Step 5. The request is assigned to the analyst that will be notified by email.

Step 6. The analyst solves the request and if he offered a good solution to the problem, its vocabulary of keywords is updated with the other significant words from the problem description.

In Figure 4 is presented the logical scheme of the algorithm for automatic requests assignment in the CMS application.

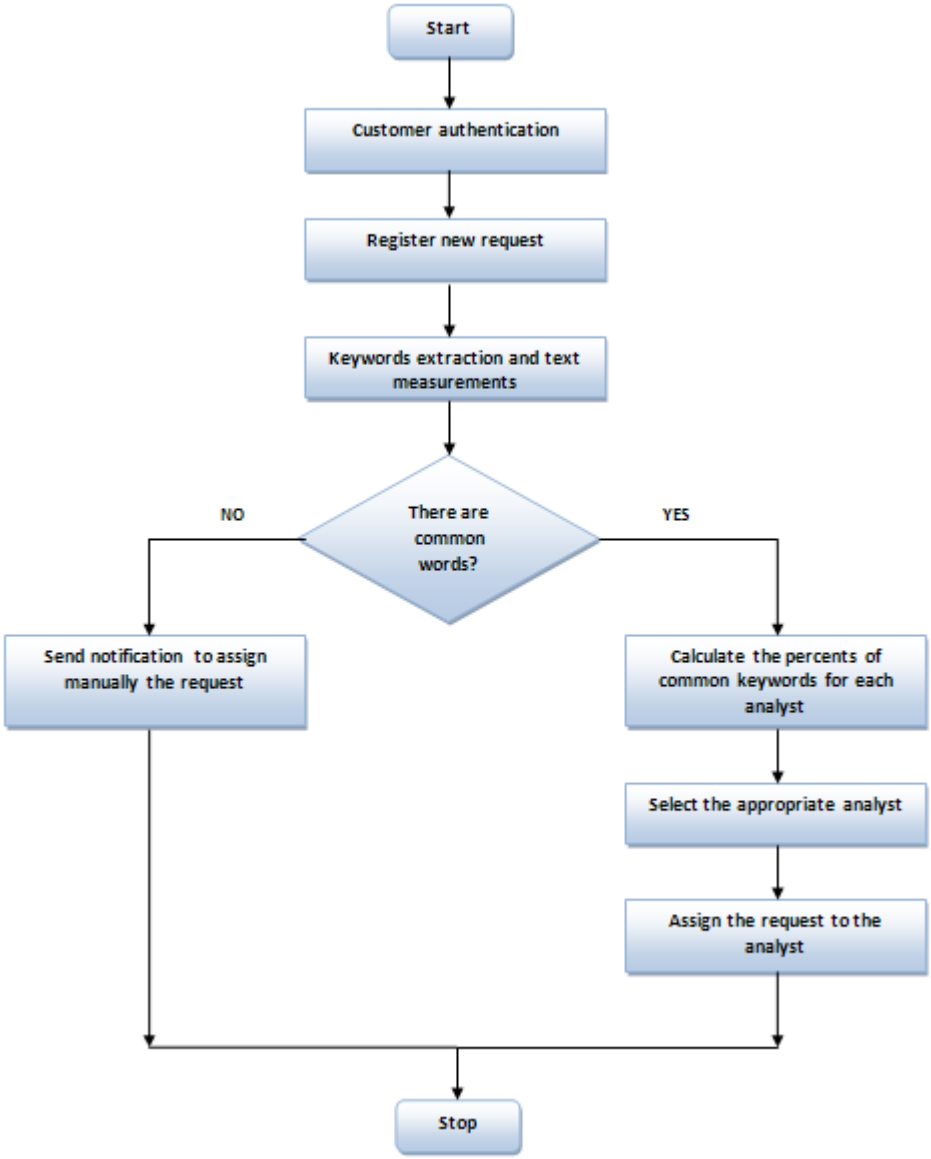

Fig. 4. The logical scheme of the automatic requests assignment algorithm

The decision point of the algorithm is given by the existence of common keywords in the problem description and the vocabularies of each analyst. If common words are

identified, then the automatic requests assignment process is performed. But, if there are no common words, then the algorithm ends and a notification is sent to all the analysts in order to assign manually the request.

The informatics applications used inside banks have increasingly become more oriented towards the citizen, respectively towards the employee and customer. The quality of Internet banking services offered by a bank or other is influenced by the interfaces of electronic payments applications. Customers prefer applications with easy interfaces, intuitive, easy to use, that do not require re-entry of data or causing confusion about what needs to be inserted in a given field. Applications that contain the implementation of auto-adaptive interfaces, which change depending on the category to which the user belongs, the data he enter or select, is the next generation of citizen-oriented informatics applications.

The auto-adaptive interfaces contribute to increasing the collaborative character of informatics applications and to better meet the needs of users. By implementing applications with interfaces that change automatically depending on certain inputs, a high level of accuracy and completeness of data entered into the databases of the applications is ensured.

In the Collaborative Multicash Servicedesk – CMS application the inquiries of bank customers are introduced, relating to the problems they have in using the Multicash electronic payments service. The CMS application is used effectively within the Raiffeisen Bank, in its database being introduced over two thousand requests per month.

In the module for recording phone requests by Multicash helpdesk analysts, after logging into the application, the analyst see the page for the registration of requests into the database.

The standard interface requires that the helpdesk analyst to complete or to select the following fields:

- the customer name, based on suggestions from a predefined list of customers using the Multicash service;
- the name of the person that call the helpdesk;
- the request category, represented by a dropdown list with predefined categories and codes associated with them;
- the request description, field in which are inserted the details of the problem;
- the solving manner, by selecting the appropriate option.

Figure 5 shows the standard interface for inserting a request into the CMS application. The customer name is automatically suggested, based on auto-complete facility, from the existing customers table of the database application, in the case in which the entered characters allow its retrieval.

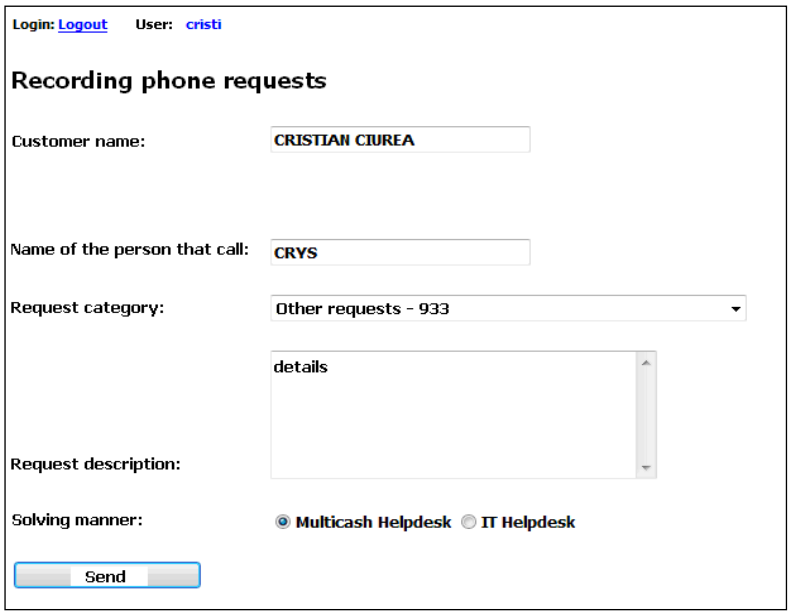

Fig. 5. The standard interface of CMS application

The standard interface for recording phone requests require multiple processing of code sequences, based on users behavior. This processing is realized on the server, but some code sequences are executed on the client side, like in a grid computing architecture.

If the request that is inserted into the application relates to an existing customer in the database, certain fields such as customer Segment and customer ID are automatically filled.

In the case in which the request relates to a new customer, the application will require the introduction of customer ID and customer Segment, for completing them in the requests table and for inserting the data regarding the new customer in the auto-complete table, in order to be retrieved at a later search.

Figure 6 shows the auto-adaptive interface for entering the request in the associated table, but also for populate the auto-complete table containing the users list of the electronic payments service.

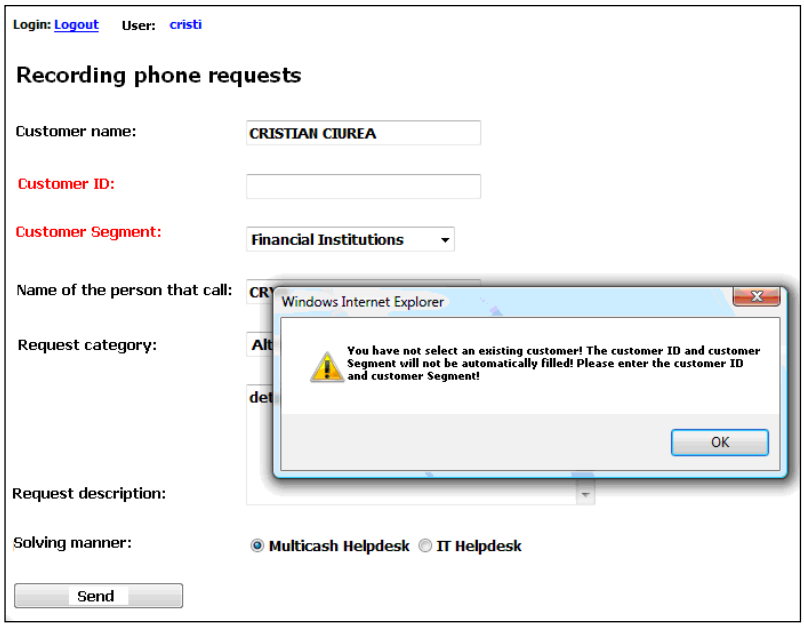

Fig. 6. The auto-adaptive interface of CMS application

Because the entered customer is not into the database, the application automatically activates the field customer ID and customer Segment to assume this information.
The implementation of auto-adaptive interfaces is possible with the help of genetic algorithms and the grid computing technology. The processing power of multiple computers is used to make available the citizen-oriented informatics applications with such interfaces.
The intelligent interfaces require running applications which include them on computers with powerful processors in order to execute code sequences and generate controls depending on the context. For the same application, the server will deliver to connected users different web pages, at the same time moment and according to users' requirements. Two users from the same category will access, at a time moment, the same page of the application, but with different content depending on which information entered each of them.
Intelligent interfaces are a necessity in the banking informatics applications, because they offer the possibility of automatic configuration of the application, based on customer preferences. Intelligent interfaces are based on intelligent agents that contribute to the generation of context and working environment for each user.

5 Conclusions and future work

Genetic algorithms use principles from genetics. More fundamental principles of genetics are borrowed and used to build efficient algorithms that are robust and which require minimal information about the problem to solve. Possible solutions of a problem are encoded into a data structure of chromosome type and recombination operators are applied on this structure.
In designing an intelligent collaborative system using genetic algorithms should be taken into account the context in which it will be used and the views of the social group that will use it.
The power of genetic algorithms is the ease with which they are implemented and that often give good results, although they not always find the global optimum. In the process of building and validating metrics of collaborative systems, the genetic algorithms proved to be efficient.
The future work of the paper is given by the implementation of the algorithm for automatic requests assignment in the CMS application, in order to extract data regarding the efficiency of the allocation process and the workload of each analyst.

Acknowledgements

This article is a result of the project POSDRU/6/1.5/S/11 „Doctoral Program and PhD Students in the education research and innovation triangle". This project is co funded by European Social Fund through The Sectorial Operational Programme for Human Resources Development 2007-2013, coordinated by The Bucharest Academy of Economic Studies, project no. 7832, Doctoral Program and PhD Students in the education research and innovation triangle, DOC-ECI.

References

[1] O. Sapena, E. Onaindia, A. Garrido and M. Arangu, "A distributed CSP approach for collaborative planning systems," *Eng. Appl. Artif. Intell.* Vol. 21, No. 5, August 2008, pp. 698-709.
[2] C. Au Yeung, M. G. Noll, N. Gibbins, C. Meinel and N. Shadbolt, "On Measuring Expertise in Collaborative Tagging Systems," *Proceedings of the WebSci'09: Society On-Line*, 18-20 March 2009, Athens, Greece.
[3] H. Skaf-Molli, C. L. Ignat, C. Rahhal and P. Molli, "New Work Modes for Collaborative Writing," *International Conference on Enterprise Information Systems and Web Technologies- EISWT 2007*, Orlando, Florida, 2007.
[4] P. Pocatilu and C. Ciurea, "Collaborative Systems Testing," *Journal of Applied Quantitative Methods*, Vol. 4, No. 3, 2009.
[5] Genetic Algorithm, Available at: http://ro.wikipedia.org/wiki/Algoritm_genetic

[6] D. Whitley, *Artificial Intelligence: Genetic Algorithms and Evolutionary Computing*, Van Nostrand's Scientific Encyclopedia, Wiley, 2005.

[7] C. Boja, *Optimizarea aplicaţiilor informatice, Teză de doctorat*, Academia de Studii Economice, Bucureşti, 2008.

[8] http://www.codeproject.com/KB/recipes/btl_ga.aspx

[9] Padej Phomasakha Na Sakolnakorn and Phayung Meesad, "Decision Tree-Based Model for Automatic Assignment of IT Service Desk Outsourcing in Banking Business," *Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, IEEE Computer Society, 2008, pp. 387–392.

Cristian CIUREA has a background in computer science and is interested in collaborative systems related issues. He has graduated the Faculty of Economic Cybernetics, Statistics and Informatics from the Bucharest Academy of Economic Studies in 2007 and the Informatics Project Management Master in 2010. He is currently conducting doctoral research in Economic Informatics at the Academy of Economic Studies. Other fields of interest include software metrics, data structures, object oriented programming in C++ and windows applications programming in C#.