International Journal of Engineering Science Invention
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org Volume 2 Issue 7 ǁ July. 2013 ǁ PP.01-08

An artificial immune system for ambiguity reduction of text using parsing

Antara Malakar[1], Himangshu Sarma[2]
[1](Computer Science and Engineering, Royal Group of Institutions, India)
[2](Computer Science and Engineering, Tezpur University, India)

ABSTRACT: *Human-Computer Interaction, often called HCI, is a field whose goal is to bring the strength of computers and communications systems to people in ways that are useful in ones working and learning. It consists of the study, planning and design of the interaction between users and computers. HCI has expanded steadily, attracting researchers from various disciplines incorporating diverse concepts. As the field is relatively new, most of the existing works have been exploratory. This paper exploits the concept of artificial immune system or AIS to solve problems of human computer interaction. AIS has become the umbrella term that covers all the exertion to develop computationally intelligent systems encouraged by the principles and processes of the vertebrate immune system. When humans communicate with the computer through text, humans might type ambiguous sentences. The user would naturally expect the machine to reflect the response stored in it. Hence, a parsing algorithm has been developed which models the text of the users in such a way that the computer can figure out the ambiguous sentences and give the desired feedback to the user. This will benefit to relieve the burden of learning special syntax by the computer.*

KEYWORDS:*ambiguity detection, artificial immune system, parsing, Human computer interaction*

I. INTRODUCTION

The Association for Computing Machinery defines human-computer interaction as "a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them." The vital challenge of HCI is the securing user satisfaction[1]. With the emergence of computers, the concept of man-machine interaction and interfacing also simultaneously came into being. The reason behind this was that the most sophisticated machines are worthless unless they can be used properly by men[2]. HCI design should consider many aspects of human behaviors and needs to be useful[3]. The complexity of the degree of the involvement of a human in interaction with a machine is sometimes invisible compared to the simplicity of the interaction method itself[4].

In this paper, several concepts from artificial immune system have been extracted and applied to solve the problems related to human computer interaction. Artificial immune system is a computational problem solving method inspired by vertebrate immune system. The immune system is capable of identifying and responding to harmful alien cells that is not recognized as self-cell[5]. Depending on the type of pathogens, and the way it enters the body, the immune system uses different response mechanisms[3][6]. One way is to destroy the infected cells and the other is to neutralize the effect. In artificial immune system, the components are artificial cells which process tasks in order to identify and forbid attacks from intrusions[7]. The artificial cells try to mould the behavior of the immune cells of human immune system.Out of the various mechanisms in the immune system that are surveyed in the AIS, negative selection is still the most discussed model. Negative selection is used to provide tolerance for self cells. It is used in detecting unknown antigens while not reacting to self cells[8]. Despite the new models being proposed the field of AIS including negative selection is young and not well explored[9].

In this paper, an interaction model has been proposed wherein the computer is able to distinguish between a self and a non-self pattern. The process of negative selection is utilized to produce an immune cell, and apply itself to self-recognition system. This holds the advantage of being able to identify many unknown antigens throughout the process.

II. ARCHITECTURE

The system will detect the query given by the user to the computer. To check the query, whether it is known or unknown, the system will retrieve the queries, which are already stored in the "question vault" of the system. The system will check if the given query is already stored in the "question vault" or not. If the query is present, it will detect the given query as "known query", otherwise it will be detected as "unknown query". In the "answer vault", every answer is stored with a unique keyword with respect to each query. For "known query", system will open the "answer vault" and will retrieve the respective answer to the environment. For

"unknown query", system will update the unknown query in the "query vault" and prompt user to suggest an answer for that particular unknown query. The suggested reply will be updated in the "answer vault" with a unique keyword generated by the system. After updating both "query vault" and "answer vault", the control is transferred to the initial stage. The process, stated above, will continue until user wants to quit from the application.

To check the presence of query string in the question vault, the query string at first parsed to remove the unwanted words[10]. Shift-reduce parsing is the first stage in the whole parsing process. Initially, the stack is empty. The first step is to shift the first element of the input string onto the stack. In the subsequent steps, a grammar consulted and matched with the input string. If it matches, the particular token is eliminated, the reduced string is obtained and the next token is shifted onto the stack. The procedure continues until the value of the stack and the reduced string is same.

III. THE PARSING SYSTEM

It consists of four main components:

1. Input string
2. Stack
3. Insignificant repository
4. Reduced string

For example:
The input string is:
"How many continents are there in this earth ?"
Stack is initially NULL
Insignificant repository is "are, in, this, ?"
Initially the reduced string will be same as the input string.

Step 1: The first token of the string "How" is entered onto the stack and compared with the insignificant repository. No matching is found. Hence, the reduced string will be same as the input string.

Step 2: Thesecond token of the string "many" is entered onto the stack and compared with the insignificant repository. No matching is found. Hence, the reduced string will be same as the input string.

Step 3: Thethird token of the string "continents" is entered onto the stack and compared with the insignificant repository. No matching is found. Hence, the reduced string will be same as the input string.

Step 4: The fourth token of the string "are" is entered onto the stack and compared with the insignificant repository. Matching is found and hence the token is eliminated. Hence, the reduced string will be –
How many continents there in this earth ?

Step 5: The fifth token of the string "there" is entered onto the stack and compared with the insignificant repository. No matching is found. Hence, the reduced string will be same as the input string.

Step 6: The sixth token of the string "in" is entered onto the stack and compared with the insignificant repository. Matching is found and hence the token is eliminated. Hence, the reduced string will be –
How many continents there this earth ?

Step 7: The seventh token of the string "this" is entered onto the stack and compared with the insignificant repository. Matching is found and hence the token is eliminated. Hence, the reduced string will be –
How many continents there earth ?

Step 8: The eighth token of the string "earth" is entered onto the stack and compared with the insignificant repository. Matching is found and hence the token is eliminated. Hence, the reduced string will be –
How many continents there earth ?

Since the value of the stack ad the reduced string is the same, hence, it is the desired reduced string.
The proposed model can be represented as follows:
Let
PS= Set of words present in Parsed String
IR= Set of words present in "Insignificant Repository"
OS= Set of words present in original string
IW= Set of insignificant words present in original string

Now it is obvious that

$IW = IR \cap OS$……(i)

When |IW| is less, it implies that there is less ambiguity in the query string and IW = Ø implies no ambiguity in the query string.

The parsed string, which is derived from the proposed model, can be expressed by following equation

$OS - IR = \{x | x \in OS \Lambda x \notin IW\}$…….(ii)

The above equation provides a set of words, which are present in the original string, but is not present in the "insignificant repository". More precisely, equation (ii) gives the desired set of words of the parsed string. Thus, equation (ii) can be rewritten as follows

$PS = OS - IR$.....(iii)

IV. ASSOCIATION BETWEEN HUMAN IMMUNE SYSTEM AND THE PROPOSED SYSTEM

TABLE I. List of proposed human immune system

Serial Number	Human Immune System	Proposed system
1.	Antibody	Internal queries.
2.	Antigen	External queries.
3.	Sensitivity level	Efficiency of the "Insignificant Repository"
4.	Memory cells	Learning mechanisms
5.	Detector	Reference keyword.
6.	Predefined amino acid structure	Predefined knowledge repository
7.	Negative selection	Triggering
8.	Epitope	Condition
9.	Paratope	Condition
10.	Memory states	Size
11.	Traffic	Message queue
12.	Via keyword	Detection
13.	Antigen reduction	Function

The antibodies of human body can identify a unique part of foreign particles called antigen. In the same way the internal queries present in the knowledge repository are used to identify foreign queries. External queries act as antigen which are the random queries given by the user. These can be detected when the internal query and the stored keyword set matches.Sensitivity depends on the efficiency of insignificant repository because as the size of the insignificant repository increases the sensitivity to recognize increases.After storing the queries and the keywords, if the updated query is asked in future, it will automatically give the response in speech. This learning process is optimized by successive training of the data sets.Reference keywords are the unique identifiers to the foreign queries.The human immune system consists of predefined amino acids. The structure of the amino acids helps to detect if it is a self or a non-self cell. The system consists of knowledge repository which is constituted of internal queries. This is used to identify foreign queries.If the foreign query that consists of epitopes matches with the keywords stored, then the query will be identified as unknown query and it will be included in the internal query with the unique keyword.Paratopes can be identified when the query given matches with the internal query in which the answer can be extracted by unique keyword.The memory states depend on the size of the knowledge repository.When the internal query and the keywords are matched, then the foreign query or the updated query is detected.The traffic in the system depends on the size of the message queue.Keyword plays a vital role in detection of queries. The queries stored can be monitored by a function in which existed keyword and the new keyword will be matched. If it is matched, it means that the foreign query is already present. If it does not match, then that particular query and the reference keyword is stored. In this way, the size of the query database can be reduced.

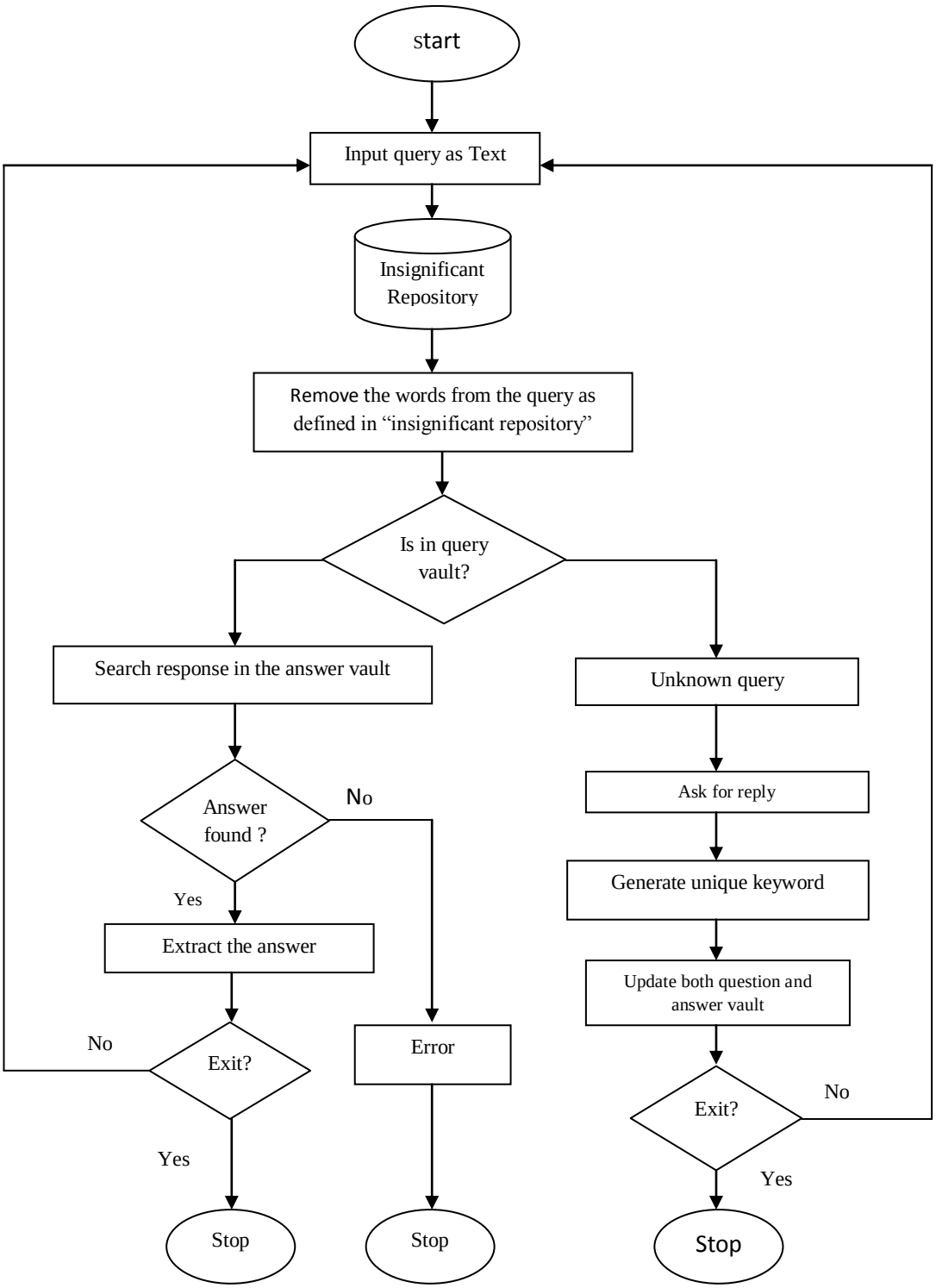
V. FLOWCHARTS

To represent some process and algorithm flowcharts are used. We have reported our flowchart in section A and in Section B we have reported our string reduced flowchart.

A. System Flowchart

Our system flowchart is shown in Fig. I

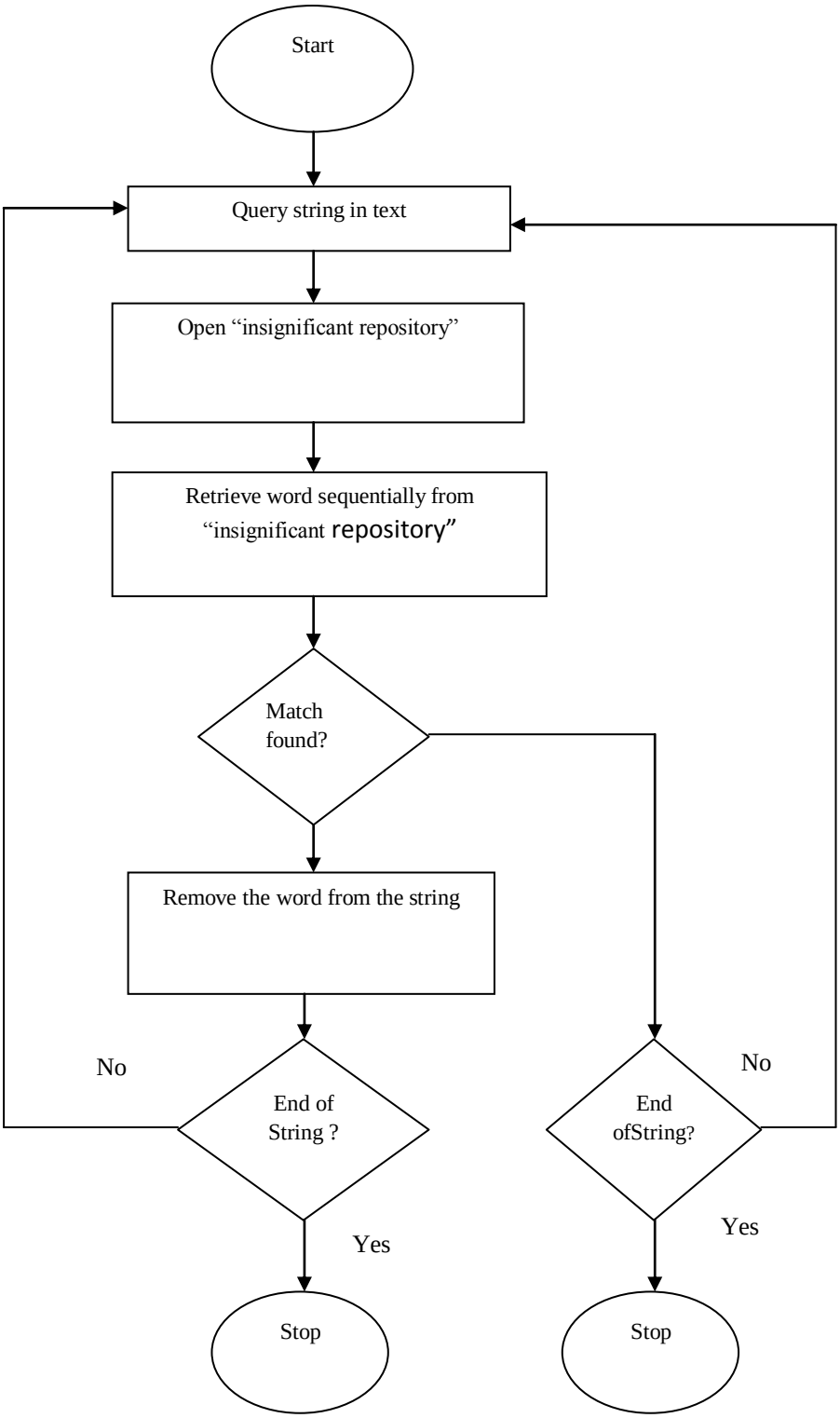
Fig I. System Flowchat



B. String Reducing Procedure Flowchart

In Fig.II we showed our string reducing procedure flowchart.

Fig II. String reducing procedure flowchart



VI. ALGORITHM

The algorithm for functions to run the model is described as follows. The algorithm of the whole system with the string reduce procedure can be written as follows [11],[12]:

START OF QUERY DETECTION SYSTEM
Step 1: Take the query as a string
Step 2: To remove the unwanted word from string as defined in "Insignificant repository" file
Step 3: Open the "Insignificant repository" file
Step 4: Take a word from the insignificant repository sequentially
Step 5: Call the "String Reducing Procedure" with three parameters
i) Query string
ii) Word from "Rule"
iii) A blank string

START OF "STRING REDUCING PROCEDURE"
Step 6: Take three parameters
i) Original string (str)
ii)Word to be replaced (orig_word)
iii) Replacing word (replace_word)

Step 7: Check whether orig_word is present in the str
Step 8: if orig_word present in str
Replace orig_word with replace_word

END OF "STRING REDUCING PROCEDURE"
Step 9: If it is not the end of "Rule" file go to step 2
Step 10: check the modified query string in the query file
Step 11: If the query string found in the query file
Step 12: Retrieve the respective answer from the answer file
Step 13: If the modified query string not found in the query file go to step 14

START OF QUERY UPDATE
Step 14: Update the query file with modified query string
Step 15: Prompt for answer for unknown query
Step 16: Update answer dictionary

- *Create a unique keyword*
- *Update given answer in answer dictionary followed by the unique keyword*

END OF QUERY UPDATE
Step 17: If user don't want to exit go to step 1
END OF QUERY DETECTION SYSTEM

VII. DESCRIPTION OF ALGORITHM

In the designed model, user interacts with the system through text. The storage structure of the system consists of three parts: "query vault", "answer vault" and "insignificant repository". The "query vault" and "answer vault" store the known query and answer respectively. The "insignificant repository" contains all insignificant words of the query given by the user. User provides the query in form of text and system takes the query as a string. The "Replace Procedure" filters the original string through parsing in following way:

The procedure retrieves the words stored in the "insignificant repository". It checks whether any word of the "insignificant repository" is present in the query string or not. If the procedure finds any insignificant word in the query string, it replaces that particular word with a blank string, i.e. it removes that particular insignificant word from the string. The procedure will check the string for every insignificant word defined in the "insignificant repository" and will produce a new modified query string.

The system will check whether the new modified query string is present in the "query vault" or not. If query string found in the "query vault", it will mark the given query as "known query" otherwise the query will be detected as "unknown query". In the answer vault, every answer is stored with a unique keyword with respect to the respective query. For "known query", system will open the answer vault and will retrieve the respective answer with the unique keyword and provide the answer in form of text to the user.

For "unknown query", system will update the unknown query in the "query vault" and prompt user to suggest an answer for that particular unknown query. The suggested reply will be updated in the "answer vault" with a unique keyword generated by the system. The main logic to generate a unique keyword is: when the system detects an unknown query, it is clear that the unknown query string is absent in both "query vault", as well as in "answer vault", i.e. it is a unique query. The unique keyword is generated from this unique query string. After updating both "query vault" and "answer vault" for unknown query, the control of the program will go back to the initial stage.

The process, stated above, will continue until user wants to quit from the system.

VIII. RESULT ANALYSIS

The efficiency of the system is dependent on the efficiency and size of the "question vault" and "Insignificant Repository". The "query vault" is learned by the user with time, i.e. with the increase of time from the time of deployment of the system, there is less number of "unknown query" and the performance of the system gets better.

On other hand "Insignificant Repository" is designed before the deployment of the system. The size of the "unknown query" is dependent on how significantly "insignificant repository" has been designed. Considering a situation where there is no word in the "Insignificant Repository", a small change in the query string cause to detect the query as "unknown query". On the contrary if a scenario is considered where every word of the English dictionary is in the "Insignificant Repository", then according to the algorithm of the system, the system will consider every query as a "known query". The reason behind this result is that the query is parsed and reduced by "Insignificant Repository" before checking with "known query" vault. If "Insignificant Repository" is built with the every word of English dictionary, the parsed and reduced query of the original query will be a blank string and it will always be treated as known query.

From the above result analysis it can be known that though the performance of the system is dependent on two parameters, i.e. "question vault" and "Insignificant Repository", the vital parameter of the system is "Insignificant Repository". The design of the "query vault" is dependent on the efficiency of "Insignificant Repository" and hence it needs to be designed efficiently.

IX. ADVANTAGES AND LIMITATION OF THE SYSTEM

In the previous work, the system was designed using Microsoft SAPI, wherein user interacts with the system through speech. The main disadvantage of HCI through speech is that: humans might speak ambiguous sentences and that too in presence of noise. So, it is required to remove the insignificant word from the given query. In the current work, parsing technique has been introduced to the existing system to remove insignificant words from the given query, in which the query passes through a filter and generates a modified query string. Thus the current system provides more accurate result than the previous one.

The limitation of the current work is that it accepts query from the user in form of text only.

X. CONCLUSION

Human-Computer Interaction is an important part of systems design. Quality of system depends on how it is represented and used by users. Therefore, enormous amount of attention has been paid to better designs of HCI. Developing Human computer interface is a challenging task that requires coordinated effort from various fields of study. In this paper, human computer interaction through text has been designed inspired by the principles of immune system. Initially, the system is trained for a particular set of queries and eventually learns the traits of foreign queries. A parsing algorithm has been developed which models the text of the users in such a way that the computer can understand the ambiguous sentences and give the desired reply to the user.

REFERENCES

Journal Papers:

[1]. Grudin, Jonathan. "Three faces of human-computer interaction." Annals of the History of Computing, IEEE 27.4 (2005): 46-62

[2]. F.Karrey, M. Alemzadeh, J.A.Saleh and M.N.Arab (2008), 'Human-Computer Interaction: Overview on State of the Art', International Journal on Smart Sensing and Intelligent Systems, Vol.1, March 2008.

[3]. Moore, Roger K. "Towards Speech-Based Human-Robot Interaction." Proc. Symposium on Language and Robotics, Aveiro, Portugal. 2007.

[4]. Singh, Chingtham Tejbanta, and Shivashankar B. Nair. "An artificial immune system for a multiagent robotics system." Proc. of the 4th World Enformatika International Conference on Automation Robotics and Autonomous Systems (ARAS 2005). 2005.

[5]. Al-Enezi, J. R., M. F. Abbod, and S. Alsharhan. "Artificial Immune Systems-models, algorithms and applications." International Journal (2010).

[6]. Forrest, Stephanie, and Steven A. Hofmeyr. "John Holland's invisible hand: An artificial immune system." Festschrift held in honor of John Holland (1999).

[7]. Sim, Kwee-Bo, and Dong-Wook Lee. "Modeling of positive selection for the development of a computer immune system and a self-recognition algorithm." system (or artificial immune system) 11 (2003): 12.

[8]. Challoo, Rajab, et al. "Navigation Control and Path Mapping of a Mobile Robot using Artificial Immune Systems." International Journal of Robotics and Automation 1.1 (2010): 01-25.

Proceedings Papers:

[9]. Chingtham, Tejbanta Singh, G. Sahoo, and M. K. Ghose. "An unmanned aerial vehicle as human-assistant robotics system." Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on. IEEE, 2010.

[10]. [de Castro, Leandro N., Fernando J. Von Zuben, and Helder Knidel, eds. Artificial Immune Systems: 6th International Conference, ICARIS 2007, Santos, Brazil, August 26-29, 2007, Proceedings. Vol. 4628. Springer, 2007.

[11]. Sharma, Rajeev, Vladimir I. Pavlovic, and Thomas S. Huang. "Toward multimodal human-computer interface." Proceedings of the IEEE 86.5 (1998): 853-869.

[12]. Hang, Xiaoshu, and Honghua Dai. "Applying both positive and negative selection to supervised learning for anomaly detection." Proceedings of the 2005 conference on Genetic and evolutionary computation. ACM, 2005.