Building of an Information Retrieval System Based on Genetic Algorithms

Badr Hssina(✉), Soukaina Lamkhantar, Mohammed Erritali, Abdelkrim Merbouha, and Youness Madani

Faculty of Sciences and Technics, Sultan Moulay Slimane University, Beni Mellal, Morocco
badr.hssina7@gmail.com

Abstract. In an information retrieval system (IRS) the query plays a very important role, so the user of an IRS must write his query well to have the expected result.

In this paper, we have developed a new genetic algorithm-based query optimization method on relevance feedback for information retrieval. By using this technique, we have designed a fitness function respecting the order in which the relevant documents are retrieved, the terms of the relevant documents, and the terms of the irrelevant documents.

Based on three benchmark test collections Cranfield, Medline and CACM, experiments have been carried out to compare our method with three well-known query optimization methods on relevance feedback. The experiments show that our method can achieve better results.

1 Introduction

Nowadays, we see an incessant development of information technologies. These technologies produce large volumes of information, which can exist in the form of different languages, making the retrieval of a specific information very difficult. To remedy this problem, the information retrieval domain provides us with the techniques and the tools necessary to easily find the looked-for information, called relevant information. These tools are called Information Retrieval Systems [16].

In an IRS, each document is represented by an intermediate representation called indexation, and to find the documents that are relevant to a user's information need, the user expresses his need by a query, and the choice of this query is a very important step in the search for relevant documents.

The first problem in an Information Retrieval System is represented in the formulation of the first request of the user. This explains the importance placed on current query optimization techniques, which allow the user to obtain his information needs. One of the most effective techniques is the relevance feedback. It uses the judgment provided by the user during the first search for information by the system to modify the second query. In fact, the application of the artificial intelligence techniques on the information science knew great progress, notably in information retrieval which is one of the principal lines of the research in

© Springer International Publishing AG 2017
S. Bouzefrane et al. (Eds.): MSPN 2017, LNCS 10566, pp. 195–206, 2017.
DOI: 10.1007/978-3-319-67807-8_15

the artificial intelligence field. Among the evolutionary algorithms in the world of artificial intelligence that gives powerful results in the field of information retrieval, we cite the genetic algorithms.

The use of the Genetic Algorithms (GA) in the Information Retrieval System has grown greatly in recent years because it gives good results in the search for information that interests us from a large volume of information. GA is used in the different steps to perform an IRS, either in the phase of reformulation of the query, the indexing phase or the search phase.

In this paper, we will present a new genetic algorithm-based query optimization method based on relevance feedback for Information Retrieval System.

The reminder of the paper is organized as follows: In Sect. 2, we present some previous work of GA in an Information Retrieval System and some related work with ours. A detailed description of our work is presented and detailed in Sect. 3. In Sects. 4 and 5, we give some experimental results. Finally, we give a conclusion and some future works in Sect. 6.

2 Genetic Algorithm in Information Retrieval System

2.1 Information Retrieval System

An Information Retrieval System is defined as a system allowing to find the relevant documents to a users query written in a free language, from a voluminous documents database.

The search for information tries to solve the following problem: "Given a very large collection of objects (mostly documents), find those that respond to a need for information expressed by a user (request)". In the Information Retrieval System, we find a request and we want to find the objects (documents) that are relevant to it. The way to evaluate a document if it is relevant or not is to calculate the similarity between the request and that document.

After the calculation of the similarity, it is important to index all the documents and also the request, that is to make them in a presentation to facilitate its use. In our case, we use the vector representation [1], where each element of the vector represents the weight (frequency) of each term or concept in the document or query.

2.2 Genetic Algorithm

Genetic algorithms are stochastic optimization algorithms based on the mechanisms of natural selection and genetics [2]. Their operation is extremely simple. We leave with a population of potential solutions(chromosomes), initially selected arbitrarily. We evaluate their relative performance(fitness)and on the basis of these performances, a new population of potential solutions is created using simple evolutionary operators: selection, crossing and mutation. This cycle is repeated until a satisfactory solution is illustrated in Fig. 1.

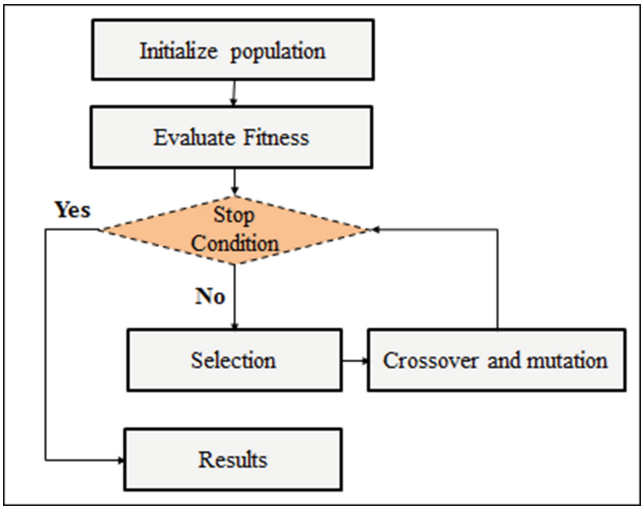

Fig. 1. General scheme of a genetic algorithm

There has been an increasing interest in the application of GA tools to IR in the last few years. Concretely, the machine learning paradigm, whose aim is the design of a system able to automatically acquire knowledge by itself, seems to be interesting in this topic.

The first thing in a genetic algorithm is the definition of the initial population (selection operator or evaluation) on which we will apply the treatment. In our case, we use the similarity calculation which plays an important role of fitness function as it enables us to decide whether an individual is going to be selected or not. There are lots of methods to make the selection such as the biased lottery, the elitist method or the selection by tournaments.

After applying the selection operator to the initial population, the second step is reproduction with the application of the crossing or crossover operation and the mutation operation.

In the literature, we find much of the work that apply genetic algorithms in the search for information, as in [3] the authors use in their Information Retrieval System the genetic algorithm to find the relevant documents for a user's query, using the vector representation to present the documents of the search base and the query. They have made comparisons with precision measurements and recall of the system using different fitness functions like cosine, Dice and Jaccard.

Vajitoru [4] also uses the Genetic Algorithms in the research of information. He proposed a new operation of crossing to improve the research with the genetic algorithm. For that, he made a comparison between his proposal and a classic GA, and the results shows the effectiveness of its proposal.

Sathya and Simon [5] use the genetic algorithms to improve an information retrieval system and make it effective for obtaining more pages relevant to the users query and optimize the search time.

In [6] the researchers present a new fitness function for approximate information retrieval which is very fast and very flexible than cosine similarity.

Fan et al. propose an algorithm for indexing function learning based on GA, whose aim is to obtain an indexing function for the key term weighting of a documentary collection to improve the IR process [7].

2.3 Genetic Algorithms in Query Optimization

In the literature, we find many works that use genetic algorithms for query optimization to improve the efficiency of Information Retrieval Systems. As in [8] the authors have utilised genetic algorithms for database query optimization for a large query. The Researchers of [9] use Genetic algorithms in Information retrieval in the area of optimizing a boolean query. They use Information Retrieval effectiveness measures, precision and recall as a fitness function. The goal of this work is to retrieve most relevant documents with less number of the non-relevant document. The authors conclude that the quality of initial population was important to have the best results of the genetic programming process, and the less quality of initial population caused worse results. To get good results, they choose parents depending on the recall fitness values than the precision fitness values.

The work of *Anubha Jain et al.* [10] reviews relevance of genetic algorithms to improve upon the user queries in the field of Information Retrieval. The results of the studies categorically prove the applicability of genetic optimization algorithms in improving the Information Retrieval process. The paper presents diverse proposals on the relevance of genetic algorithm in search query optimization which are promising and still developing areas of research.

As pointed out by Leroy et al. [11], the query optimization methods based on relevance feedback or genetic algorithms using dynamic query contexts could help users search the internet. From the study of Salton and Buckley [12], we know that, in a method, the calculation of traditional relevance feedback query optimization expression is simple, but the determination of its parameters is difficult. According to the study of Lopez-Pujalte et al. [13], the order information of the relevant documents is very useful to search an optimized solution for a genetic algorithm-based relevance feedback method.

In a genetic algorithm based query optimization method, the key work to consider is how to use the relevance feedback information to design its genetic operators and fitness function.

3 Genetic Algorithm Based Query Optimization Method

3.1 Document Vectorization and Relevance Feedback

We produce a dictionary $D = (t_1, t_2, ..., t_n)$, each document in the collection is described as an n-dimensional weight vector $w = (w_1, w_2, ..., w_n)$, where each weight w_i is calculated by the TF*IDF formula, and each query in the collection is also described as a weight vector $q = (u_1, u_2, ..., u_n)$, is calculated by the TF-method formula.

–
$$TF = \frac{f(t_i, d_j)}{N} \tag{1}$$

$f(t_i, d_j)$ is the number of occurrences of the term t_i in the document d_j and N is the total number of terms in the document d_j.

–
$$IDF = \frac{log(f(t_i, d_j))}{M} \tag{2}$$

$f(t_i, d_j)$ is the number of occurrences of the term t_i in the document d_j and M is the total number of documents in the corpus.

For each query, the top-15 documents retrieved based on the cosine similarity (ranking the values in a descending order) will be input to our GA as relevance feedback.

$$Sim_{cos}(X, Y) = \frac{\sum_{i=1}^{n} x.y}{\sqrt{(\sum_{i=1}^{n} x^2)} \cdot \sqrt{(\sum_{i=1}^{n} y^2)}} \tag{3}$$

3.2 Chromosomes and Population

A chromosome is represented as a weight vector $w = (w_1, w_2, ..., w_n)$, where w_i is a real number and denotes the weight of the keyword t_i for i = 1,2,. . .n .

Our GA receives an initial population P consisting of $\mid R_{rel} \mid +2$ chromosomes, including the original query vector q, the $\mid R_{rel} \mid$ relevant document vectors in R_{per} and the average-weight vector $q_{avg} = (avg_1, avg_2, ..., avg_n)$.

3.3 Fitness Function

In our GA, the definition of our fitness function consists of two parts: x and y. The x is relative to both the order of appearance of the relevant documents in feedback and the terms of relevant documents in feedback. The y is relative to the terms of the irrelevant documents in feedback.

For any chromosome $w = (w_1, w_2, ..., w_n)$ in the current population P, its fitness value is calculated by the formula:

$$F(w) = x + y \tag{4}$$

The formula of x is:

$$\sum_{d_i \in R_{rel}} \mid Horng(w) - cosine(w, d_i) \mid \tag{5}$$

The Horng and Yeh fitness function is defined as:

$$Horng(w) = \frac{1}{\mid R \mid} \sum_{i=1}^{|R|} \left[r(d_i) \sum_{j=1}^{|R|} \frac{1}{j} \right] \tag{6}$$

Here, $\mid R \mid$ is the number of the documents in set R. $d_1, d_2, ..., d_{|R|}$ are the documents in R sorted by descending order of their cosine similarity values with the chromosome w. Function $r(d_i)$ gives the relevance of d_i, being unity if d_i belongs to R_{rel} and zero if d_i belongs to R_{irrel}.

The formula of y is:

$$y = \sum_{d_i \in R_{irrel}} Cosine(w, d_i) \tag{7}$$

$\sum$ counts for every document $d_i in R_{irrel}$, its cosine similarity with the chromosome w.

3.4 Genetic Operators

The formal definitions of the three genetic operators used in our GA can be described as follows:

Two-Point Crossover: Firstly, two integers i and j in (1, 2, . . . , n) will be produced randomly, and we select two parents w and v, which are randomly selected using the fitness proportional selection from current population P. Suppose $1 \leq i \leq j \leq n$, and the two parents are:

$w = (w_1, w_2, , w_{(i-1)}, \mid w_i, ..., w_j, \mid w_{(j+1)}, ..., w_n), v = (v_1, v_2, , v_{(i-1)}, \mid v_i, ..., v_j, \mid v_{(j+1)}, ..., v_n)$

then, two offspring w and v will be generated as below:

$w^{'} = (w_1, w_2, , w_{(i-1)}, \mid v_i, ..., v_j, \mid w_{(j+1)}, ..., w_n), v^{'} = (v_1, v_2, , v_{(i-1)}, \mid w_i, ..., w_j, \mid v_{(j+1)}, ..., v_n)$

Weight-Adjusting Mutation: This genetic operator is used to tune the weights of keywords (genes) in a chromosome. It can generate an offspring from a parent w, which is randomly selected using the fitness proportional selection from the current population P.

Firstly, an integer i in (1, 2, ..., n) will be produced randomly, and then a real number $w_i^{'}$ between MIN_i *and* MAX_i will be produced randomly. Finally, from the parent:

$$w = (w_i, w_2, ..., w_{(i-1)}, w_i, w_{i+1}, ..., w_n)$$

an offspring $w^{'}$ will be generated as below:

$$w^{'} = (w_i, w_2, ..., w_{(i-1)}, w_i^{'}, w_{i+1}, ..., w_n)$$

Overturn Mutation: Firstly, an integer i in (1, 2, . . . , n) will be produced randomly, and then from the parent:

$$w = (w_i, w_2, ..., w_{(i-1)}, w_i, w_{i+1}, ..., w_n)$$

an offspring $w^{'}$ will be generated as below by executing a reversal operation between zero and non zero:

$$w^{'} = (w_i, w_2, ..., w_{(i-1)}, w_i^{'}, w_{i+1}, ..., w_n)$$

where $w_i^{'}$ will be $(MAX_i + MIN_i)/2$ if $w_i = 0$, otherwise it will be 0.

3.5 Next Generation

After the offspring have been produced by operating our three genetic operators given above with configurable probabilities, our fitness function is used to determine the chromosomes of the next generation.

Firstly, the offspring is added into current population P. Secondly, the fitness values of all chromosomes in P are calculated. Lastly, the $\mid R_{rel} \mid +2$ chromosomes with the smaller fitness values (i.e. better chromosomes) in P will be brought to the next generation.

3.6 Termination Criteria and Solution

The iterative procedure of our GA will be stopped by one of the following termination criteria:

- From a generation, its fitness value does not change for the rest of the iterations.
- From a generation, its fitness value changes but very weakly for the rest of the iterations.
- A threshold of the number of iterations is reached.

If one of these criteria is met then the value of the fitness function of the generation is defined as the best fitness value of all the current generations.

After stopping the iteration procedure, the chromosome with the lowest fitness value (the best chromosome) in the latest generation P will be selected as the optimized query produced by our genetic algorithm.

4 Experimental Results

4.1 Test Collections

Our experiments were carried out based on three benchmark test collections:

- Cranfield: contains 1400 documents on different aspects of aeronautical engineering.
- Medline: contains 1033 documents on medicine.
- CACM: contains 3204 documents on computing.

4.2 Experiments Preparation

Dictionaries: In our experiments, for the efficiency of converting the documents and queries in a test collection into the weight vectors in VSM, a dictionary of keywords was used for each test collection. The dictionary was formed with the following procedure:

1. Extract all the words from all documents in each collection.
2. Remove stop-words using the list of stop-words generated according to the frequency dictionary of Kucera and Francis [14].
3. Stem the rest of the words using the Porter Stemmer [16], which is the most commonly used stemmer in English.
4. Delete all irrelevant words to reduce the size of the weight vector, such as the words that appear before the text for each document.

As a result, the dictionary for Cranfield collection contains 3824 keywords; the dictionary for Medline collection contains 6985 keywords and the dictionary for CACM collection contains 719 keywords.

Description of Documents and Queries: In each collection, when using its dictionary to generate the keyword vector of each document, we need first to use the Porter Stemmer to stem the document, then to extract keywords from the document according to the dictionary, and last to calculate keyword weights with the TF*IDF-method.

In addition, for each request for a given collection, its term vector is treated in the same way as the documents, but the weight of the terms is calculated by the TF method.

4.3 Selection of Relevance Feedback

In our experiments, for each query in a collection, the first 15 documents (a = 15) extracted and sorted in descending order of the cosine similarity values with the query will be examined to determine their relevance. The first 15 documents will be used for the relevancy judgment, which will be used to optimize the query and includes the four query optimization methods that we will compare with our experiments.

4.4 Selection of Queries

For each collection, we have selected only the queries that result at least three relevant documents by the first 15 documents found, and do not extract at least five documents. Our experiments were carried out on these queries.

5 Explanation of Our Experiments

Based on the descriptions of the Dec-hi method, the Fitness9 and the Fitness10 in Lopez-Pujalte et al.'s experiments [12], we have realized the Ides traditional Dec-hi method [4], the Horng and Yehs GA-based method [15] and the Lopez-Pujalte et al.'s GA-based method [12]. Below, we use Dec-hi(Ide), Fitness9 (Horng) and Fitness10 (Pujalte) to represent respectively the three query optimization methods. As done in Lopez-Pujalte et al.'s experiments, in our experiments both Fitness9 and Fitness10 use the one-point crossover and the random mutation genetic operators.

5.1 Control Parameters

All the control parameters used in our genetic algorithm have been determined experimentally. The crossover probability c1 is 0.4. The mutation probabilities m1 and m2 are 0.3 and 0.3, respectively. The limit on the number of iterations is 2000, namely threshold $\beta = 2000$.

For Fitness 9 and Fitness10, the probabilities of the one-point crossover and random mutation genetic operators are 0.6 and 0.4, respectively. Their limits on the number of iterations are 2000 and 200, respectively, because from our experiments we have found that the fitness value of Fitness10 only varies in the first few iterations.

5.2 Evaluation and Experimental Plan

As done in Lopez-Pujalte et al.'s experiments, we evaluate the results of retrieval by the classical measures of recall and precision. The precision is calculated by interpolation at fixed recall intervals. We calculate the average precision for three recall values (0.25, 0.5, and 0.75, representing low, medium, and high recall, respectively) so as to be able to compare the different methods.

The experimental plan follows the following steps:

- Each query is compared with all documents belonging to a given collection, using the Cosine similarity measure. Therefore, a similarity list of each query with the other documents in the collection is obtained.
- This list is ranked in descending order of degree of similarity.
- The standardized document vectors corresponding to the first 15 documents in the list, with their degrees of similarity to the standardized query vector, will be the inputs of our genetic algorithm.
- The program produces a hidden file containing for each request all the documents that are not to be considered in the evaluation process, i.e. the first 15 documents used in the modification of the requests. This method is called the residual collection method, used by Salton and Buckley [16].

5.3 Experimental Results and Comparison

Comparison Between Our Genetic Algorithm and the Other Genetic Algorithms: Based on CACM, Medline and Cranfield collections, we have conducted three experiments to compare our method with the two other methods: Fitness9 (Horng) and Fitness10 (Pujalte). Our experiment results on three collections that are shown, respectively, in Figs. 2, 3 and 4.

From these figures, we can see that, by making comparison with the original query, fitness9 function, fitness10 function and our GA have increased the average accuracy, respectively, from 133.71 and 159.43 to 177.05 for the CACM collection, from 33.84 and 74.46 to 76.55 for the Medline collection, and from 66.95 and 118.41 to 120.97 for the Cranfield collection.

We can also see that, compared with the original query, Ide Dec-hi method and our GA the average accuracy have raised respectively from 150.61 to 177.05 on the CACM collection and from 105.47 to 120.97 on the Cranfield collection.

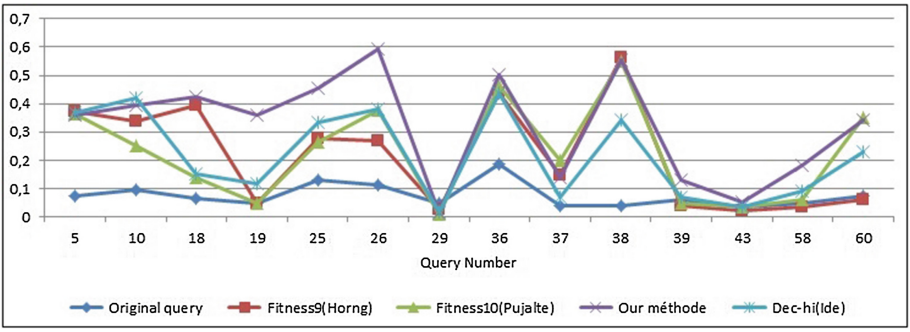

Fig. 2. Comparison of our GA, to other GAs and the Ide Dec-hi method (on CACM)

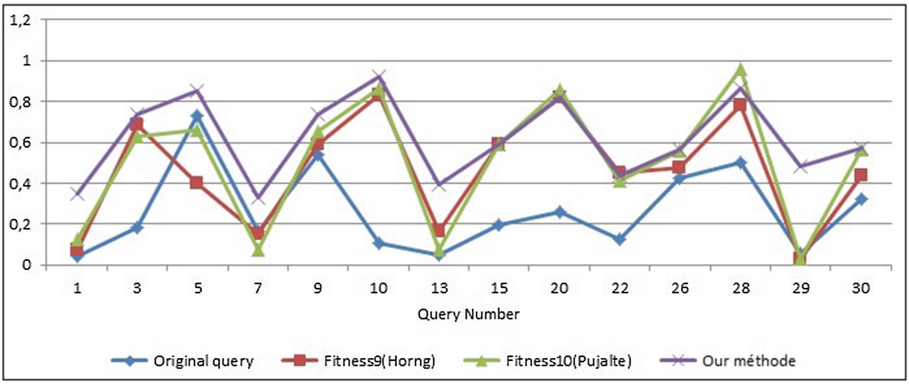

Fig. 3. Comparison of our genetic algorithm and other genetic algorithms (on Medline)

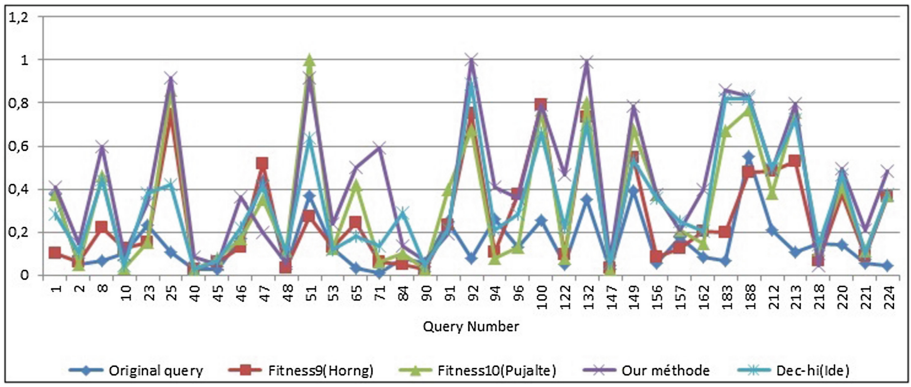

Fig. 4. Comparison of our GA, to other GAs and the Ide Dec-hi method (Cranfield)

6 Conclusion and Perspectives

In this work, we have presented our genetic algorithm for optimizing a query in order to improve the results of an Information Retrieval System by searching the optimal query that gives us the best results.

Based on three benchmark test collections: Cranfield, Medline and CACM, we have conducted three experiments to compare our GA-based method with three other well-known query optimization methods on relevance feedback: Horng method (Fitness 9), Lopez-Pujalte method (Fitness 10) and the traditional Ide Dec-hi method. The results of our experiments indicate that: First, based on the Cranfield, Medline and CACM collections, our GA-based method can get better results than both the Horng and Yehs GA and the Lopez-Pujalte et al.'s GA. Second, based on the Cranfield and CACM collections, our GA can also achieve better results than the traditional Ide Dec-hi method.

As Perspectives, We aim to consolidate the proposed approach by evaluating it on other larger collections such as the well-known collection called TREC, then work on languages other than English to prove the effectiveness of our method.

An other perspective of our work is to apply our method in an e-learning system for finding an optimal profile for a learner.

References

1. Salton, G.: Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison- Wesley Publishing Co., Inc., New York (1989)
2. Alam, F., Saadi, H.S., Alam, M.S.: A novel comparative study between dual population genetic algorithm and artificial bee colony algorithm for function optimization. In: 19th International Conference on Computer and Information Technology (ICCIT) (2016)

3. Klabbankoh, B., Pinngern, O.: Applied genetic algorithms in information retrieval. IJCIM **7**(3) (December 1999)
4. Vrajitoru, D.: Crossover improvement for the genetic algorithm in information retrieval. Inform. Process. Manag. Int. J. **34**(4), 405–415 (1998)
5. Simon, P., Siva Sathya, S.: Genetic algorithm for information retrieval. In: lAMA 2009 (2009)
6. Radwan, A.A.A., Latef, B.A.A., Ali, A.M.A., et al.: Using genetic algorithm to improve information retrieval systems. World Acad. Sci. Eng. Technol. **17**, 1021–1027 (2008)
7. Fan, W., Gordon, M.D., Pathak, P.: AIJPersonalization of search engine services for effective retrieval and knowledge management, In: Proceedings of 2000 International Conference on Information Systems (ICIS), Brisbane, Australia (2000)
8. Butey, P.K., Meshram, S., Sonolikar, R.L.: Query optimization by genetic algorithm. J. Inform. Technol. Eng. **3**(1), 44–51 (2012) ISSN: 2229–7421
9. Owais, S.S.J., Kromer, P., Snasel, V.: Query optimization by genetic algorithms. In: Dateso 2005, pp. 125–137 (2005). ISBN 80-01-03204-3
10. Jain, A., Chande, S.V., Tiwari, P.: Relevance of genetic algorithm strategies in query optimization in information retrieval. Int. J. Comput. Sci. Inform. Technol. **5**(4), 5921–5927 (2014)
11. Leroy, G., Lally, A.M., Chen, H.: The use of dynamic contexts to improve casual Internet searching. ACM Trans. Inform. Syst. **21**(3), 229–253 (2003)
12. Lopez-Pujalte, C., Guerrero-Bote, V.P., Moya-Anegon, F.D.: A test of genetic algorithms in relevance feedback. Inform. Process. Manage. **38**, 793–805 (2002)
13. Cordon, O., Herrera-Viedma, E., Lopez Pujalte, C., Luque, M., Zarco, C.: A review on the application of evolutionary computation to information retrieval. Int. J. Approximate Reasoning **34**, 241–264 (2003)
14. Kucera, H., Francis, W.N.: Computational Analysis of Present-day American English. Brown University Press, Providence (1967)
15. Horng, J.-T., Yeh, C.-C.: Applying genetic algorithms to query optimization in document retrieval. Inform. Process. Manage. **36**, 737–759 (2000)
16. Buckley, C., Salton, G., Allan, J., Singhal, A.: Automatic query expansion using SMART: TREC 3. In: Proceedings of the Third Text Retrieval Conference, Gaithersburg, Maryland pp. 69–80 (1994)