Bibliography

1. R. Agrawal and J. C. Shafer, "Parallel Mining of Association Rules", *IEEE Transactions On Knowledge And Data Engineering*, pages 962–969, 1996.
2. R. Agrawal and R. Srikant, "Fast algorithms for mining association rules", In *Proc. of the Int. Conf. on Very Large Databases*, SanDiago, Chile, pages 487–499, 1994.
3. J. Chattratichat, "Large Scale Data Mining: Challenges and Responses," *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, 1997, pp 143-146.
4. W. Li, J. Han and J. Pei, "CMAR: Efficient classification based on multiple class-association rules. In *Proc. of the Int. Conf. on Data Mining*, pages 369–376, 2001.
5. B. Liu, W. Hsu and Y. Ma, "Integrating classification and association rule mining", In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.
6. D. B. Skillicorn, "Strategies for parallel data mining", *IEEE Concurrency*, vol. 7, No. 4,1999.
7. X. Yin and J. Han, "CPAR: Classification based on Predictive Association Rules", In *Proc. of the Int. Conf. on Data Mining*, SDM. SIAM, 2003.
8. F. Thabtah, P. Cowling and Y. Peng, "MCAR: Multi-class Classification based on Association Rule Approach. In *Proceeding of third IEEE International Conference on Computer Systems and Applications*, Cairo, Egypt, pages 1-7, 2005.
9. F. Thabtah, P. Cowling and Y. Peng, "MMAC: A New Multi-class, Multi-label Associative Classification Approach", In *Proceeding of fourth IEEE International Conference on Data Mining* (ICDM '04), Brighton, UK, pages 217-224, Nov. 2004.

Authors' Information

Thakur S. Ghanshyam, Dr. R.C.Jain *– Department of Computer Application; Samrat Ashok Technological Institute; Vidisha(M.P.), INDIA; e-mail: ghanshyamthakur@gmail.com*

ENHANCING INFORMATION RETRIEVAL BY USING EVOLUTION STRATEGIES

Abdelmgeid Amin Aly

Abstract: *Similar to Genetic algorithm, Evolution strategy is a process of continuous reproduction, trial and selection. Each new generation is an improvement on the one that went before. This paper presents two different proposals based on the vector space model (VSM) as a traditional model in information Retrieval (TIR). The first uses evolution strategy (ES). The second uses the document centroid (DC) in query expansion technique. Then the results are compared; it was noticed that ES technique is more efficient than the other methods.*

1. Introduction

Since the 1940s the problem of Information Retrieval (IR) has attracted increasing attention, especially because of the dramatically growing availability of documents. IR is the process of determining relevant documents from a collection of documents, based on a query presented by the user.

There are many IR systems based on Boolean, vector, and probabilistic models. All of them use their model to describe documents, queries, and algorithms to compute relevance between user's query and documents.

Information Retrieval (IR) proposes solutions for searching, in a given set of objects, for those replying to a given description. IR tries to make a suitable use of these databases, allowing the users to access to the information which is really relevant in an appropriate time interval [1]. Unfortunately, commercial IR Systems (IRSs), usually based on the Boolean IR model [2], have provided unsatisfactory results. Vector space, probabilistic and fuzzy models, which have been developed to extend the Boolean model [3], as well as the application of knowledge-based techniques, have solved some of these problems, but there are still some lacks [4]. In the last few years, an increasing interest on the application of artificial intelligence (AI)-based techniques to IR has been shown with the aim of solving some of those lacks.

One of the AI areas with a considerable growth in the last decades is evolutionary computation (EC) [5], based on the use of models of evolutionary process for the design and implementation of computer-based problem solving systems. The different models which have been proposed within this philosophy are named in a generic way as evolutionary algorithms (EAs) [5].

In this paper, we introduce two different proposals using our IR model. One is using Evolution Strategies (ES) and the second using the Document Centroid (DC) in Query Expansion (QE) technique. Then the results are compared with our Traditional IR (TIR) model. To this end, we used the ES that presented the best performance, running it with two different fitness functions in the vector space model which is the most commonly used model in this type of application. We applied it to three well-known test collections (CISI, CACM and NPL). This allows us to generalize our earlier results and conclusion.

2. Antecedents

2.1. Evolutionary algorithms

EC [5] uses computational models of evolutionary processes as key elements in the design and implementation of computer-based problem solving systems. There is a variety of evolutionary computational models that have been proposed and studied, which are referred as EAs [5]. There have been four well defined EAs which have served as the basis for much of the activity in the field: Genetic Algorithms (GAs) [6], Evolution Strategies (ES) [7], Genetic Programming (GP) [8] and Evolutionary Programming (EP) [9].

An EA maintains a population of trial solutions, imposes random changes to these solutions, and incorporates selection to determine which ones are going to be maintained in future generations and which will be removed from the pool of trials. There are some important differences between the existing EAs. GAs [6] emphasize models of genetic operators as observed in nature, such as crossover (recombination) and mutation, and these re appllied to abstracted chromosomes with different representation schemes according to the problem being solved. Evolution strategies and evolutionary programming only apply to real-valued problems and emphasize mutational transformations that maintain the behavioral linkage between each parent and its off-spring.

As regards GP [8], it constitutes a variant of GAs, based on evolving structures encoding programs such as expression trees. Apart from adapting the crossover and mutation operators to deal with the specific coding scheme considered, the remaining algorithm components remain the same.

2.2. Evolution Strategies

Evolution strategies (ESs) were independently developed by Rechenberg [10], with selection, mutation, and a population of size one. Schwefel [11], introduced recombination and populations with more than one individual, and provided a nice comparison of ESs with more traditional optimization techniques. Evolution strategies typically use real-valued vector representations. Evolution strategies are similar to genetic algorithms in that both attempt to find a (near-)optimal solution to a problem within a search space (all possible solutions to a problem) without exhaustively testing all solutions.

Evolution strategies are based on the principal of strong causality, which states that similar causes have similar effects. That is, a slight change to one encoding of a problem only slightly changes its optimality. The process of evolution strategy can be summarized by a relatively simple algorithm:

1. Generate some random individuals
2. Select the p best individuals based on some selection algorithm (fitness function)
3. Use these p individuals to generate c children (using mutation or recombination)
4. Go to step 2, until the ending condition is satisfied (i.e. little difference between generations, or maximum number of iterations completed).

2.3. Automatic Query Expansion

The automatic query expansion or modification based on term co-occurrence data has been studied for nearly three decades. The various methods proposed in the literature can be classified into the following four groups:

1. Simple use of co-occurrence data. The similarities between terms are first calculated based on the association hypothesis and then used to classify terms by setting a similarity threshold value [12], [13] and [14]. In this way, the set of index terms is subdivided into classes of similar terms. A query is then expanded by adding all the terms of the classes that contain query terms. It turns out that the idea of classifying terms into classes and treating the members of the same class as equivalent is too naive an approach to be useful [13], [15] and [16].
2. Use of document classification. Documents are first classified using a document classification algorithm. Infrequent terms found in a document class are considered similar and clustered in the same term class (thesaurus class) [17]. The indexing of documents and queries is enhanced either by replacing a term by a thesaurus class or by adding a thesaurus class to the index data. However, the retrieval effectiveness depends strongly on some parameters that are hard to determine [18]. Furthermore, commercial databases contain millions of documents and are highly dynamic. The number of documents is much larger than the number of terms in the database. Consequently, document classification is much more expensive and has to be done more often than the simple term classification mentioned in 1.
3. Use of syntactic context. The term relations are generated on the basis of linguistic knowledge and co-occurrence statistics [19], [20]. The method used grammar and a dictionary to extract for each term *t* a list of terms. This list consists of all the terms that modify *t*. The similarities between terms are then calculated by using these modifiers from the list. Subsequently, a query is expanded by adding those terms most similar to any of the query terms. This produces only slightly better results than using the original queries [19].
4. Use of relevance information. Relevance information is used to construct a global information structure, such as a pseudo thesaurus [21], [22] or a minimum spanning tree [23]. A query is expanded by means of this global information structure. The retrieval effectiveness of this method depends heavily on the user's relevance information. Moreover, the experiments in [23] did not yield a consistent performance improvement. On the other hand, the direct use of relevance information, by simply extracting terms from relevant documents, is proved to be effective in interactive information retrieval [24], [25]. However, this approach does not provide any help for queries without relevance information.

In addition to automatic query expansion, semi-automatic query expansion has also been studied [26], [27] and [28]. In contrast to the fully automated methods, the user is involved in the selection of additional search terms during the semi-automatic expansion process. In other words, a list of candidate terms is computed by means of one of the methods mentioned above and presented to the user who makes the final decision. Experiments with semi-automatic query expansion, however, do not result in significant improvement of the retrieval effectiveness [26]. We use a document centroid (DC) as the basis of our query expansion.

3. System Framework

3.1. Building IR System

The proposed system is based on Vector Space Model (VSM) in which both documents and queries are represented as vectors. Firstly, to determine documents terms, we used the following procedure:

- Extraction of all the words from each document.
- Elimination of the stop-words from a stop-word list generated with the frequency dictionary of Kucera and Francis [29].
- Stemming the remaining words using the porter stemmer that is the most commonly used stemmer in English [3], [30].

After using this procedure, the final number of terms was 6385 for the CISI collection, 7126 for CACM and 7772 for NPL. After determining the terms that described all documents of the collection, we assigned the weights by using the formula (1) which proposed by Salton and Buckley [25]:

$$a_{ij} = \frac{\left(0.5 + 0.5\,\frac{tf_{ij}}{\max tf}\right) \times \log \frac{N}{n_i}}{\sqrt{\left(0.5 + 0.5\,\frac{tf_{ij}}{\max tf}\right)^2 \times \left(\log \frac{N}{n_i}\right)^2}} \longrightarrow \qquad (1)$$

where a_{ij} is the weight assigned to the term t_j in document D_i, tf_{ij} is the number of times that term t_j appears in document D_i, n_j is the number of documents indexed by the term t_j and finally, N is the total number of documents in the database.

Finally, we normalize the vectors, dividing them by their Euclidean norm. This is according to the study of Noreault et al. [31], of the best similarity measures which makes angle comparisons between vectors. We carry out a similar procedure with the collection of queries, thereby obtaining the normalized query vectors. Then, for applying ES, we apply the following steps:

- For each collection, each query is compared with all the documents, using the cosine similarity measure. This yields a list giving the similarities of each query with all documents of the collection.
- This list is ranked in decreasing order of similarity degree.
- Make a training data that consists of the top 15 document of the list with a corresponding query.
- Automatically, the keywords (terms) are retrieved from the training data and the terms which are used to form a query vector.
- Adapt the query vector using the ES approach.

3.2. The Evolution Strategy Approach

Once significant keywords are extracted from training data (relevant and irrevelent documents) including weights are assigned to the keywords. We have applied ES to get an optimal or near optimal query vector. Also we have compared the result of the ES approach with both the result obtained of (DC) and the traditional IR system. The (ES) approach will be explained in the following subsections.

Encoding & Fitness Functions

To implement an evolution strategy, the individuals in the population (solutions) need to be represented. Unlike genetic algorithms, which use bit strings, evolution strategies encode these individuals as vectors of real numbers (object parameters). Another vector of parameters, the strategy parameters, affects the mutation of the object parameters. Together, these two vectors constitute the individual's chromosome.

To distinguish whether one solution is more optimal than another, we use the cosine similarity as fitness function (2).

$$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{t} x_i^2 \cdot \sum_{i=1}^{t} y_i^2}} \longrightarrow \qquad (2)$$

where X_i is the real representation weight of term *i* in the chromosome, Y_i is the real representation weight of that term in the query vector and *t* is the total number of terms in the query vector as in a given chromosome.

Forming the Next Generation

One key difference from genetic algorithms is that only the *p* most fit individuals in the population survive until the next generation (this form of selection is known as elitist selection). (Genetic algorithms usually use roulette wheel selection to give the fittest individuals a better chance of survival, but don't, like evolution strategies, guarantee that they will survive.) Using the fitness function as the evaluator, the *p* best individuals from the population are selected to be the parents of the next generation. A large value of *p* prevents bad characteristics from being filtered out of the gene pool (since they will persist from generation to generation), while a small value reduces variation in the gene pool, increasing the need for mutation. These *p* parent individuals produce a total of *c* children using mutation and recombination. The parents can be included in the next generation. Producing more children increases the probability of achieving better solutions, Mutation and recombination are used, but there are some Differences in how they are applied.

Mutation

To simulate mutation, random changes to the chromosome are made. These changes are necessary to add new genes to the gene pool; otherwise an optimal solution could not be reached if a necessary gene is absent.

Recombination

Recombination (also known as crossover) is the process where two or more parent chromosomes are combined to produce a child chromosome. Recombination is necessary in cases where each child is to have multiple parents, since mutation provides no mechanism for the "mixing" of chromosomes. We use a single point recombination, exchanges the weights of sub-vector between two chromosomes, which are candidate for this process.

Evolution Process

Figure 1 outlines a typical evolution strategy (ES). After initialization and evaluation, individuals are selected uniformly randomly to be parents. In the standard recombinative ES, pairs of parents produce children via recombination, which are further perturbed via mutation. Survival is deterministic and is implemented in one of two ways.

```
procedure ES; {
t = 0;
initialize population P(t);
evaluate P(t);
until (done) {
t = t + 1;
parent_selection P(t);
recombine P(t)
mutate P(t);
evaluate P(t);
survive P(t);
}         }
```

Fig. 1. The evolution strategy algorithm

The first allows the best children to survive and replaces the parents with these children. The second allows the N best children and parents to survive. Like EP, considerable effort has focused on adapting mutation as the algorithm runs by allowing each variable within an individual to have an adaptive mutation rate that is normally distributed with a zero expectation. Unlike EP, however, recombination does play an important role in evolution strategies, especially in adapting mutation.

3.3. The Query expansion approach

After using the vector space model (VSM) to represent the user's query and the documents. Each document d_k in the document database is represented by a document vector $\overline{d}_k$, the system calculates the degree of similarity between the query vector $\overline{Q}$ and each document vector $\overline{d}_k$. Then, the system ranks the document according to their degrees of similarity with respect to the user's query from the largest to the smallest. Based on the relevant degree of relevant documents, get the top 15 documents for each query. The system considers each term appearing in any relevant document from the top 15 documents as a relevant term. The weight of each relevant term in each relevant document is calculated using formula (1). The average weight W_{avg} of each relevant term t_i is calculated as follows:

$$W_{avg} = \frac{\sum_{k=1}^{15} w_{ik}}{15} \quad \rightarrow \tag{3}$$

where w_{ik} denotes the weight of relevant term t_i relevant document d_k. The result obtained from equation (3) represents the document Centroid (DC). The original and the additional terms together form the expanded query that is, consequently, used to retrieve documents, to get the result of the DC. We have three types of results, one for Traditional IR (TIR), second from adding the Document Centroid (DC), and third from the Evolution Strategies (ES).

4. Experimental Results

The test databases for our approaches are three well-known test collections, which are: the CISI collection (1460 documents on information science), the CACM collection (3204 documents on Communications), and finally the NPL collection (11,429 documents on electronic engineering). One of the principal reasons for choosing more than one test collection is to emphasize and generalize our results in all alternative test documents collections. The Experiments are applied on 100 queries chosen according to each query which does not retrieve 15 relevant documents for our IR system.

CACM Collection Results for 100 Queries

Table (1), and its corresponding graph represented in figure (2) both are using non-interpolated average Recall – Precision relationship. From this table we notice that ES gives a higher improvement than TIR with 21.35% and higher than DC with 39.6 % respectively as average values. The average number of terms of query vector before applying ES is 160.7 terms; these terms are reduced after applying ES to 16.83 terms, and increasing to 167.8 terms when applying DC approach.

CISI Collection Results For 100 Queries

Table (2), and its corresponding graph represented in figure (3) both are using non-interpolated average Recall – Precision relationship. From this table we notice that ES gives a higher improvement than TIR with 17.66% and less than DC with 20.6% respectively as average values. The average number of terms of query vector before applying ES is 509.61 terms; these terms are reduced after applying ES to 358.84 terms, and increasing to 514.9 when applying DC approach.

NPL Collection Results for 100 Queries:

Table (3), and its corresponding graph represented in graph (3) both are using non-interpolated average Recall – Precision relationship. From this table we notice that ES gives a higher improvement than TIR with 19.82% and higher than DC with 99.82% respectively as average values. The average number of terms of query vector before applying ES is 134.14 terms; these terms are reduced after applying ES to 16.8 terms, and increasing to 142.9 terms when applying DC approach.

Table (1): Experimental results on CACM Collection

Average Recall-Precision Relationship			
	Precision		
Recall	TIR	DC	ES
0.1	0.72267	0.774552	0.81134
0.2	0.41646	0.52837	0.50888
0.3	0.36936	0.442366	0.45776
0.4	0.24673	0.267359	0.31126
0.5	0.21268	0.109873	0.2632
0.6	0.15801	0.005216	0.20032
0.7	0.14291	0.005216	0.17607
0.8	0.10728	0.005216	0.141
0.9	0.08965	0.005216	0.12236
Average	0.27397	0.238154	0.33247

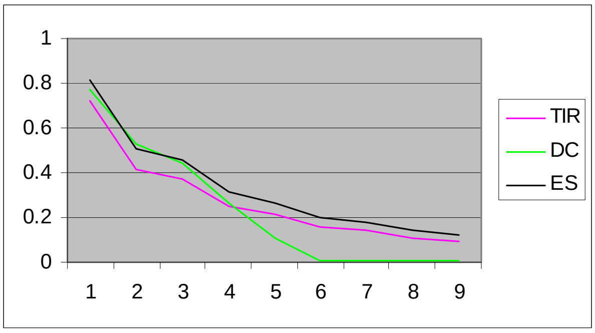

Fig. 2. The relationship between average recall-precision for 100 queries on CACM

Table (2): Experimental results on CISI Collection

Average Recall-Precision Relationship			
	Precision		
Recall	TIR	DC	ES
0.1	0.67935	0.83819	0.84987
0.2	0.55781	0.753699	0.66449
0.3	0.46199	0.732035	0.5962
0.4	0.4007	0.67232	0.46771
0.5	0.34937	0.612855	0.40763
0.6	0.30394	0.454284	0.31125
0.7	0.25167	0.355882	0.27429
0.8	0.19887	0.187812	0.20741
0.9	0.14908	0.150724	0.16604
Average	0.37253	0.528645	0.43832

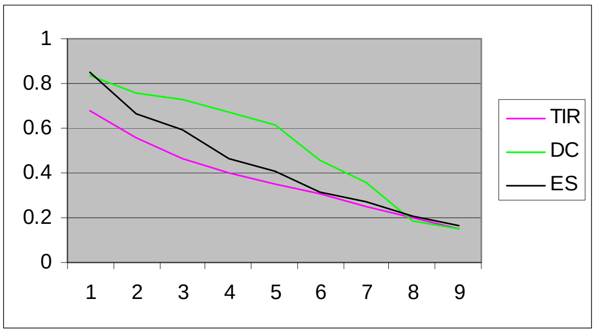

Fig. 3. The relationship between average recall-precision for 100 queries on CISI

Table (3): Experimental results on NPL Collection

Average Recall-Precision Relationship			
	Precision		
Recall	TIR	DC	ES
0.1	0.73292	0.886421	0.80875
0.2	0.50337	0.457369	0.56654
0.3	0.43515	0.303652	0.50423
0.4	0.34047	0.147124	0.4184
0.5	0.31333	0.053247	0.39739
0.6	0.23999	0.00282	0.31045
0.7	0.21539	0.00282	0.27597
0.8	0.17428	0.00282	0.23302
0.9	0.14555	0.00282	0.20027
Average	0.34449	0.206566	0.41278

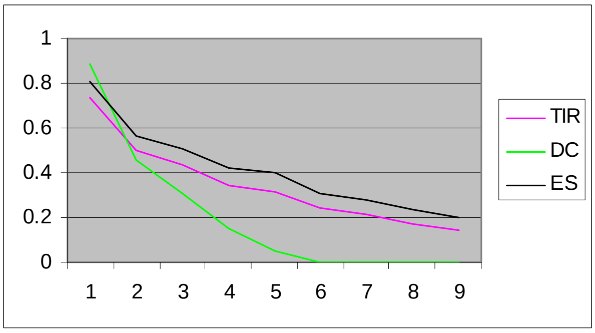

Fig. 4. The relationship between average recall-precision for 100 queries on NPL

5. Conclusion

The goal is to retrieve most relevant documents with less number of non-relevant documents with respect to user's query in information retrieval system using evolution strategies. Our results have been applied on three well-known test collections (CISI, CACM and NPL), and compare the results of three variant methods (TIR, DC and ES). The results demonstrate that evolution strategies are effective optimization technique for Document retrieval.

Bibliography

[1] G. Salton, M.H. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, 1983.

[2] C.J. Van Rijsbergen, Information Retrieval, second ed., Butterworth, 1979.

[3] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison, 1999.

[4] H. Chen et al., "A machine learning approach to inductive query by examples: an experiment using relevance feedback", ID3, genetic algorithms, and simulated annealing, Journal of the American Society for Information Science 49 (8) (1998) 693–705.

[5] T. Bäck, D.B. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation, IOP Publishing and Oxford University Press, 1997.

[6] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, 1996.

[7] H.-P. GenerSchwefel, Evolution and Optimum Seeking, in: Sixth Generation Computer Technology Series, John Wiley and Sons, 1995.

[8] J. Koza, "Genetic Programming", On the Programming of Computers by means of Natural Selection, The MIT Press, 1992.

[9] D.B. Fogel, "System Identification trough Simulated Evolution: A Machine Learning Approach", Ginn Press, USA, 1991.

[10] I., Rechenberg, "Evolutions strategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution", Frommann-Holzboog, Stuttgart (1973).

[11] H.-P., Schwefel, Numerical Optimization of Computer Models, New York: John Wiley & Sons (1981).

[12] M.E., Lesk, "Word-word association in document retrieval systems", American Documentation, 20(1): 27-38, 1969.

[13] J.,Minker, Wilson, G.A., Zimmerman, B.H., "An evaluation of query expansion by the addition of clustered terms for a document retrieval system", Information Storage and Retrieval, 8(6): 329-48, 1972.

[14] K., Sparck-Jones, E.B., Barber, "What makes an automatic keyword classification effective?", Journal of the ASIS, 18: 166-175, 1971.

[15] H.J.,Peat, P., Willett, "The limitations of term co-occurrence data for query expansion in document retrieval systems", Journal of the ASIS, 42(5): 378-83, 1991.

[16] K., Sparck-Jones, "Notes and references on early classification work". SIGIR Forum, 25(1): 10-17, 1991.

[17] C.J., Crouch, "An approach to the automatic construction of global thesauri", Information Processing & Management, 26(5): 629-40, 1990.

[18] C.J.,Crouch, B.,Yong, "Experiments in automatic statistical thesaurus construction", SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval, Copenhagen, Denmark, 77-87, June 1992.

[19] G., Grefenstette, "Use of syntactic context to produce term association lists for retrieval", SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval, Copenhagen, Denmark, 89-97, June 1992.

[20] G.,Ruge, "Experiments on linguistically-based term associations", Information Processing & Management, 28(3): 317-32, 1992.

[21] G., Salton, "Experiments in automatic thesaurus construction for information retrieval", Information Processing 71, 1: 115-123, 1971.

[22] G., Salton, "Automatic term class construction using relevance-a summary of work in automatic pseudo classification", Information Processing & Management, 16(1): 1-15, 1980.

[23] A.F., Smeaton, C.J., van Rijsbergen, "The retrieval effects of query expansion on a feedback document retrieval system", The Computer Journal, 26(3): 239-46, 1983.

[24] Harman, D., "Relevance feedback revisited", SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval, Copenhagen, Denmark, 1-10, June 1992.

[25] G., Salton, C. Buckley, "Improving Retrieval Performance by Relevance Feedback", Journal of the ASIS, 41(4): 288-297, 1990.

[26] F.C., Ekmekcioglu, A.M., Robertson, P., Willett, "Effectiveness of query expansion in ranked-output document retrieval systems", Journal of Information Science, 18(2): 139-47, 1992.

[27] M., Hancock-Beaulieu, "Query expansion: advances in research in on-line catalogues", Journal of Information Science, 18(2): 99-103, 1992.

[28] S.J., Wade, P., Willett, " INSTRUCT: a teaching package for experimental methods in information retrieval". III. Browsing, clustering and query expansion, Program, 22(1): 44-61, 1988.

[29] H. Kucera, N. Francis. “Computational analysis of present-day American English”. Providence, RD: Brown University Press (1967).

[30] M. F. Porter. “An algorithm for suffix stripping. Program”, 14(3), 130–137 (1980).

[31] T. Noreault, M. McGill and M. B. Koll. “A performance evaluation of similarity measures, document term weighting schemes and representation in a Boolean environment”. Information retrieval research. London: Butterworths (1981).

Author's Information

A. A. Aly – Computer Science Department, Minia University, El Minia, Egypt; Email: abdelmgeid@yahoo.com