Web Document Query Optimization Based on Memetic Algorithm

Ziqiang Wang, Xia Sun, Dexian Zhang
School of Information Science and Engineering,
Henan University of Technology, Zhengzhou, Henan, 450001, China
wzqagent@126.com

Abstract

To efficiently retrieve relevant documents from the explosive growth of the Internet and other sources of information access, an efficient document query optimization approach based on memetic algorithm(MA) is proposed. Experimental results show that the proposed algorithm can improve the precision of document retrieval compared with other conventional query optimization algorithm.

1. Introduction

With the rapid development of Internet, information on the Internet is increasing exponentially. As a consequence, the role of information retrieval (IR) systems is becoming more important. One of the most important and difficult operations in information retrieval is to generate queries that can succinctly identify relevant documents and reject irrelevant documents. In order to get good retrieval performance, there has been a growing interest in applying genetic algorithm(GA) to the information retrieval domain with the purpose of optimizing document descriptions and improving query formulation[1,2].The main advantages of GA lie in its global convergence, inherent parallel search nature, and great robustness. However, owing to the slow convergence for each generation, a revised evolutionary algorithm to improve the convergence efficiency is needed for superior Web document query optimization.

Recently, Moscato introduced the term "memetic algorithm" (MA)[3] which combines evolutionary algorithms with the intensification power of a local search, and has a pragmatic perspective for better effects than GA. As such MA, a local optimizer is applied to each offspring before it is inserted into the population in order to make it towards optimum and then GA platform as a means to accomplish global exploration within a population. MAs have been successfully applied to several NP optimization problems such as scheduling problem, cell formation problem and TSP problem. Nevertheless, the use of the algorithm for document query optimization is still a research area where few people have tried to explore. In this paper, the objective is to investigate the capability of the MA for Web document query optimization in the context of information retrieval. Experimental results show that the proposed algorithm can improve the precision of document retrieval markedly compared with genetic algorithm.

The paper is organized as follows: The memetic algorithm(MA) is described in section 2. The memetic algorithm for document query optimization in information retrieval is introduced in section 3. Experimental results are given in section 4. Conclusions are presented in section 5.

2. Brief review of memetic algorithm (MA)

Memetic algorithms(MAs) are inspired by Dawkins' notion of a meme [3,4]. MAs are similar to GA but the elements that form a chromosome are called memes, not genes. The unique aspect of the MAs algorithm is that all chromosomes and offsprings are allowed to gain some experience, through a local search, before being involved in the evolutionary process[5]. A pseudocode for a MA procedure is given in as follows:

Begin;
Generate random population of P solutions (chromosomes);
For each individual $i \in P$: calculate fitness (i);
For each individual $i \in P$: do local-search(i);
For i=1 to number of generations;
Randomly select an operation (crossover or mutation);
If crossover;
Select two parents at random i_a and i_b;
Generate on offspring $i_c = crossover(i_a, i_b)$;
$i_c = local - search(i_c)$;
Else If mutation;
Select one chromosome *i* at random;
Generate an offspring $i_c = mutate(i)$;
$i_c = local - search(i_c)$;
End if;
Calculate the fitness of the offspring;

978-0-7695-3490-9/08 $25.00 © 2008 IEEE
DOI 10.1109/PACIIA.2008.100

If i_c is better than the worst chromosome then replace the worst chromosome by i_c;

Next *i*;

Check if termination=true;

End.

From the above pseudocode of memetic algorithm (MA), we can observe that the parameters involved in MAs are the same four parameters used in GA: population size, number of generations, crossover rate, and mutation rate in addition to a local-search mechanism.

3. MA-based query optimization algorithm

The proposed method is based on a vector space model[6] in which both documents and queries are represented as vectors. The goal of our MA is to find an optimal set of documents which best match the user's need by exploring different regions of the document space simultaneously. The system ranks the documents according to the degree of similarity between the documents and the query vector. The higher the value of the similarity measure is, the closer to the query vector the document is. If the value of the similarity measure is sufficiently high, the document will be retrieved. The MA attempts to involve, generation by generation, a population of queries towards those improving the outcome of the system. The memtic algorithm (MA) for document query optimization is described as follows:

Step1: Encoding of query individual. The first step toward implementation of the MA is the definition the chromosomes to be evolved (i.e., solution space).In this paper, the chromosome is represented by query vector space. Each query individual representing a query is of the form:

$$Q_u(q_{u1}, q_{u2}, \cdots, q_{uT}) \tag{1}$$

where T is total number of stemmed terms automatically extracted from the documents, q_{ui} is the weight of the ith term in Q_u, Initially, a term weight is computed as the following formula[7]:

$$q_{ui} = \frac{(1+\log(tf_{ui}))\cdot\log(\frac{N}{n_i})}{\sqrt{\sum_{k=1}^{T}((1+\log(tf_{uk}))\cdot\log(\frac{N}{n_k}))^2}} \tag{2}$$

where tf_{ui} is the frequency of term t_i in document d_u, N is the total number of documents, n_i is the number of documents containing the term i.

Step2: Fitness function computation. The fitness represents the effectiveness of a query during the retrieving stage. It is computed according to the relevance of the retrieved documents. The formula is as follows:

$$Fitness(Q_u^{(s)}) = \frac{\sum_{d_j \in D_r^{(s)}} Sim(d_j, Q_u^{(s)})}{\sum_{d_j \in D_{nr}^{(s)}} Sim(d_j, Q_u^{(s)})} \tag{3}$$

where N is the total number of documents, $D_r^{(s)}$ is the set of relevant documents retrieved at the generation(s) of the MA, d_j is the *jth* document , $D_{nr}^{(s)}$ is the set of non-relevant documents retrieved at the generation(s) of the MA, $Sim(d_j, Q_u^{(s)})$ is a similar measure function defined as follows:

$$\begin{aligned} Sim(d_j, Q_u^{(s)}) &= \cos(d_j, Q_u^{(s)}) \\ &= \frac{\sum_{i=1}^{T}(q_{ui}^{(s)} \cdot d_{ji})}{\sqrt{\sum_{i=1}^{T} q_{ui}^2} \cdot \sqrt{\sum_{i=1}^{T} d_{ui}^2}} \end{aligned} \tag{4}$$

Step3: Local search. For each individual $Q_u \in P$: do local-search(Q_u);

Step4: For *i*=1 to number of generations;

Select two parents at random Q_a and Q_b;

Generate on offspring $Q_c = crossover(Q_a, Q_b)$;

$Q_c = local-search(Q_c)$;

Generate an offspring $Q_c = mutate(Q_c)$;

Calculate the fitness of the offspring in terms of Equation(2);

If Q_c is better than the worst chromosome

Then replace the worst chromosome by Q_c;

Next *i*;

Step5: Relevant documents merging. At each generation of MA, these retrieved relevant documents by all the individual queries of the query population are merged to a single document lists, and presented to user. Our adopted merging methods according to the following formula:

$$\mathrm{Re}l^{(s)}(d_j) = \sum_{Q_u^{(s)} \in Pop^{(s)}} Fitness(Q_u^{(s)}) \cdot RSV(Q_u^{(s)}, d_j)$$

where $Pop^{(s)}$ is the population at the generation(s) of the MA; $RSV(Q_u^{(s)}, d_j)$ is the retrieval status value(RSV)[8] of the document d_j for the query $Q_u^{(s)}$ at the generation(s) of the MA.

The main characteristic of this merging method is the use of the real fitness values and retrieval status value of retrieved relevant documents, thus, the merged relevant documents are more fit to user query requirement.

Step6: Check if the iteration number approaches to the predefined maximum iteration;

Step7: End.

The pseudocode for the above local-search procedure is described as follows:

Begin;

Select an incremental value $d = a * Rand()$, where a is a constant that suits the variable values, and $Rand()$ is random generator in [0,1];

For a given chromosome $Q_u \in P$: calculate fitness $Fitness(Q_u)$;

For j=1 to number of memes in chromosome Q_u;

$$q_{uj} = q_{uj} + d ;$$

If chromosome fitness not improved then

$$q_{uj} = q_{uj} - d ;$$

If chromosome fitness not improved then retain the original value q_{uj};

Next j;

End.

4. Experimental results

In order to verify the performance of our method, we have evaluated the performance of memtic algorithm (MA) by comparing it with genetic algorithm(GA)[2] and particle swarm optimization(PSO) algorithm[9]. The simulation has been carried out on a PC Pentium machine with 512M main memory, running on Microsoft Windows XP.

our experiment used the best known TREC collections, namely TREC D1&D2[10],which contain about 742,600 documents, One of the principal reasons for the choice of these collections was that they had been used elsewhere for query optimization experiments. We retrieved the top-ranked 1000 documents for 50 queries, and evaluated the results of the retrieval via the classical measures of recall and precision.

The parameter values of memetic algorithm(MA) is set as follows: population size=50,maximum number of generation=5,selection rate=0.75, crossover rate=0.7, mutation rate=0.05, and constant a=0.5.

The comparison of our proposed memtic algorithm (MA) with genetic algorithm(GA) and particle swarm optimization(PSO) algorithm is shown in Figure 1. From results of Figure 1, we can see that our query optimization algorithm can improve the precision of document retrieval markedly compared with GA and PSO. The reason is that we designed corresponding crossover, mutation operator according to itself characteristics of information retrieval, and used local search method to speed up finding a better query vector near the original query vector after applying the genetic operator. Then, our proposed query optimization algorithm markedly improved the precision of document retrieval.

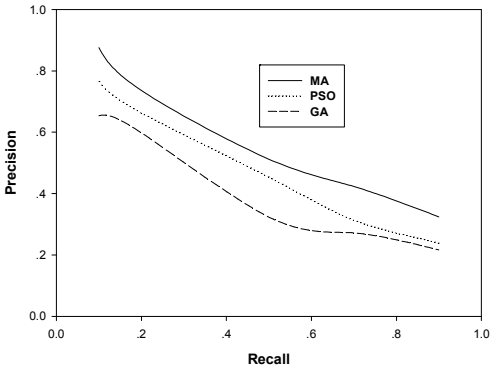

Figure 1 Performance comparison different algorithms

Table 1 Comparison the number of relevant document

Iterations	MA	PSO	GA
Iter-1	112	110	97
Iter-2	73	68	64
Iter-3	55	52	51
Iter-4	58	57	53
Iter-5	46	45	32

In addition, we also compare the number of relevant document retrieved using MA, PSO and GA. Table 1 gives the number of relevant document retrieved at each iteration of the three optimization algorithm. We can clearly see that our MA more effective than GA and PSO in retrieving relevant documents. Indeed the cumulative total number of relevant documents using MA through all the iterations is higher than using PSO and GA. Therefore, our proposed document query optimization algorithm efficiently improves the performance of the query search.

5. Conclusions

In this paper, we have presented a memetic algorithm for document query optimization. Extensive experiments clearly demonstrate its effectiveness. For further research, we plan to combine statistical learning theory to further improve the document retrieval performance.

Acknowledgement

This work is supported by the National Natural Science Foundation of China under Grant No.70701013 and the Natural Science Foundation of Henan Province under Grant No. 0611030100.

References

[1] Chen Hsinchun(1995).“Machine learning for information retrieval: neural networks, symbolic learning and genetic algorithms”. *Journal of the American Society for Information Science*, Vol.46, No.3, pp.194-216.

[2] Horng Jorng-Tzong,Yeh Ching-Chang(2000). “Applying genetic algorithms to query optimization in document retrieval”. *Information Processing & Management*, Vol.36, No.5, pp.737-759.

[3] Moscato Pablo(1989). “On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms”. *Technical Report Caltech Concurrent Computation Program Report 826*, California Institute of Technology, Pasadena, pp.1-68.

[4] Dawkins Richard. *The Selfish Gene*. Oxford University Press, Cary,1990.

[5] Elbeltagia Emad, Hegazyb Tarek and Grierson Donald (2005). “Comparison among five evolutionary-based optimization algorithms”. *Advanced Engineering Informatics*, Vol.19, No.1, pp.43-53.

[6] Salton Gerard, Wong A. and Yang C.S.(1975). “A vector space model for information retrieval”. *Communications of the ACM*, Vol.18, No.11,pp.613-620.

[7] Singhal Amit, Buckley Chris and Mitra Mandar(1996). “Pivoted document length normalisation”. *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.21-29.

[8] Aguilera A.I., Tineo L.J.(2000). “Flexible query processor for information”. *Proceedings of the Ninth IEEE International Conference on Fuzzy Systems*, Vol.2, pp. 1009-1012.

[9] Eberhart R., Kennedy J.(1995). “A new optimizer using particle swarm theory”. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp.39-43.

[10] Harman Donna(1993). “Overview of the first text retrieval conference”. *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.36-47.