A TECHNIQUE OF PROBABILITY IN DOCUMENT SIMILARITY COMPARISN IN INFORMATION RETRIEVAL SYSTEM

Poltak Sihombing[1], Abdullah Embong[2], Putra Sumari[3].

[1] *Program Studi Ilmu Komputer, Universitas Sumatera Utara, Medan, Indonesia*
[2,3] *School of Computer Science, Universiti Sains Malaysia, Penang, Malaysia*

[1] *poltakhombing@yahoo.com*; [2] *ae@cs.usm.my*; [3] *putras@cs.usm.my*

Abstract

Nowadays, storing of documents in an information retrieval system is no longer an issue due to the availability of huge storage space, multiple storage devices and different storage media, and the occurrence of various methods of document storage. The challenge is more on the retrieval of the documents since documents stored in a database grow very fast and soon become unmanageable. In this paper we propose a technique of probability to retrieve a document from one or more databases based on a similarity measure. The similarity measure is calculated using Jaccard formulation. Jaccard's score is used to represent a general measurement of document similarity. We have implemented a prototype of an information retrieval system based on genetic algorithm. This algorithm is basically based on natural biological evolution. The parent solution (chromosome) with the higher level of fitness has a bigger probability to reproduce, while those with lower level of fitness have less probability to reproduce. Documents with a higher Jaccard's score reflect a higher probability of similarity. Application of this technique will facilitate searching and retrieval of required document from one or more databases based on the representation of the similarity level.

Keywords: probability technique, similarity level representation, document retrieval, Genetic Algorithm.

1. Introduction

In the past few decades, documents stored in a database in Information Retrieval System (IRS) grow very fast. The most important problem in IRS is to get most relevant document from a database. Some times the user are unable to retrieve the required document. To solve this problem, researchers have implemented some methods such as inverted index, Boolean querying techniques, knowledge-based techniques, neural network, probabilistic retrieval techniques and *machine learning* approach [1]. Probabilistic retrieval techniques have been used to improve the information retrieval performance. The approach is based on two main parameters, the probability of relevance and the probability of irrelevance of a document [2].

Machine learning approach is a new paradigm which has attracted attention of researchers not only in IRS but olso in artificial intelligence, computer science, and other functional

disciplines such as engineering, medicine, and business [3] [4]. In contrast to *performance systems* which acquire knowledge from human experts, *machine learning systems* acquire knowledge automatically from examples, i.e., from source data. The most frequently used techniques include symbolic, inductive learning algorithms, [5], multiple-layered, feed-forward neural networks such as Backpropagation networks [6], and evolution-based **genetic algorithms (GA)** [7] [8]

In this paper, we proposed a technique of probability in document similarity comparisn in information retrieval system by using genetic algorithm. By similarity comparison we can compare the rangking of the relevant documents by taking the similarity level as a result of retrieval. Our objective is to developed the probability technique into the concept of artificial intelligencia (AI) and GA and implemented it in IRS.

2. Genetic Algorithm

GA is one of the branches of *Artificial Intelligence* representing a computational model which is inspired by the evolution theory. In GA, the following steps are repeated until a solution is found [7] [8]:

a. Forming chromosome model.

Before GA can be used to solve a problem, a way of *encoding* a potential solution to the problem must be found. A chromosome is formed by gene which represents bit (0 and 1). In IRS the keywords used in the set of user-selected documents were first identified to represent the underlying bit strings for the initial population. Each bit represents the same unique keyword throughout the complete GA process. When a keyword is present in a document, the bit is set to 1, otherwise it is set to 0. Each document could then be represented in terms of a sequence of 0s and 1s which is called a chromosom model, for examole: 01101010110101101. At the beginning of a run of a genetic algorithm a large population of *random* chromosomes is created. A Chromosome model depend on the case, example given the following equation as a case:

$$a + b * c - d * e = 100$$

At the equation searched a variable score a, b, c, d, and e to be getting 100. Supposing maximum score to each variable is 15, meaning in binary representation there are four bits to each score. Because there are five (5) variables, hence chromosome length is 20 bit. Taking example at one particular solution / chromosome, score of variable a is $10(1010_2)$, variable b is $5(0101_2)$, variable c is $14(1110_2)$, variable d is $2(0010_2)$, and variable e is $9(1001_2)$, hence chromosome at the solution is [7] [8]:

10100101111000101001

b. Forming Early Population At Random.

Population represents corps of chromosome. The ancestors which represent early population is formed randomly. The amount of the early population do not have directive, according to existing problems and ability of computer. Example of early population for the case above is as follows [7] [8]:

- 01101101101010010111(a=6, b=13, c=10, d=9, e=7)
- 01011010110101101010(a=5, b=10, c=13, d=6, e=10)
- 11011011110010100100(a=13, b=11, c=12, d=10, e=4)
- 01011010001010100110(a=5, b=10, c=2, d=10, e=6)
- 11011011110110101111(a=13, b=11, c=13, d=10, e=15)

c. Evaluating Fitness To Each Chromosome.

The objective of this phase is to obtain score of fitness to be used in selecting chromosome for the next generation. Fitness evaluation depends on the case and/or problems. For example, if a = 0, b=0, c=0, d=15, e=15, then the worst score is -325 because it needs a score of 325 to get the expected score of 100, the calculation can be seen below:

$$|0 + 0 * 0| - |15 * 15| = 100$$
$$|0| - |225| = 100$$

Fitness' score is calculated on below:

Fitness'score = | P | - | Q - R | ; P = negation of worst score; Q = expected score; R = result of the equation with substitution for chromosom variable. The calculation fitness'score for the example of previous population can be seen at following tables 1.1 below [7] [8]:

Tables 1. Calculation fitness' score.

Chromosome	Variable score	Result of Equation	Fitness's score
(A) 01101101101010010111	a=6, b=13, c=10, d=9, e=7	6 + 13 * 10 – 9 * 7 = 73	325 - |100-73| = 298
(B) 01011010110101101010	a=5, b=10, c=13, d=6, e=10	5 + 10 * 13 – 6 * 10 = 75	325 - |100 – 75| = 300
(C) 11011011110010100100	a=13, b=11, c=12, d=10, e=4	13 + 11 * 12 – 10 * 4 = 105	325 - |100 – 105 | = 320
(D) 01011010001010100110	a=5, b=10, c=2, d=10, e=6	5 + 10 * 2 – 10 * 6 = -35	325 - |100 – (-35) | = 190
(E) 11011011110110101111	a=13, b=11, c=13, d=10, e=15	13 + 11 * 13 – 10 * 15 = 6	325 - |100 – 6 | = 231

d. Determination Of Population

Determination of population on next generation based on fitness'score. The higher level of fitness has a bigger probability to reproduce, while those with lower level of fitness have less probability to reproduce. Commonly, this probability is selected by Roulette wheel selection method.

In the previous example, total fitness' score is: 298 + 300 + 320 + 190 + 231 = 1339. Hence, level of cutting to each chromosome is[7] [8] [9]:

- 298/1339 * 100 % = 22 %
- 300/1339 * 100 % = 22 %
- 320/1339 * 100 % = 25 %
- 190/1339 * 100 % = 14 %
- 231/1339 * 100 % = 17 %

The roulette wheel selection can be seen at the following figure below:

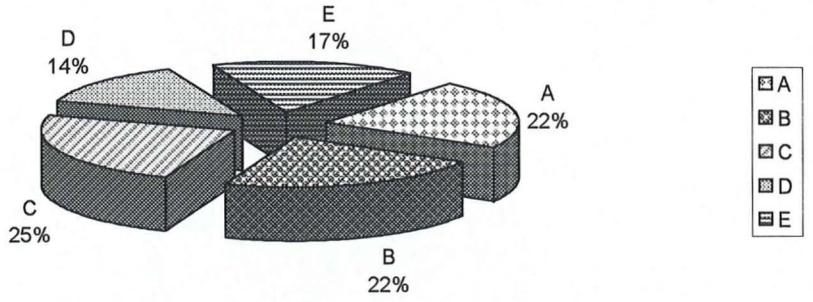


Fig. 1 Roulet/Circle diagram of fitness.

From the roulette/circle its found that score from 0 to 22 is belong to chromosome A, score from 22.1 to 44(44=22+22) belong to chromosome B, score from 44.1 to 69(69 = 44 + 25) belong to chromosomeC, score from 69.1 to 83 (83=69+14) belong to chromosome of D, and score from 83.1 to 100 (100=83+17) belong to chromosome E. By intake of random score with range 0 to 100 counted 5 times, hence:

- 42 : residing in region B
- 83 : residing in region D
- 33 : residing in region B
- 11 : residing in region A
- 60 : residing in region C

Hence, the next generation populations are:

- 01011010110101101010 (chromosome of B)
- 01011010001010100110 (chromosome of D)
- 01011010110101101010 (chromosome of B)
- 01101101101010010111 (chromosome of A)
- 11011011110010100100 (chromosome of C)

e. Crossover.

This is simply the chance that two chromosomes will swap their bits. A good value for this is around 0.7. Crossover is performed by selecting a random gene along the length of the chromosomes and swapping all the genes after that point.

e.g. Given two chromosomes

10001001110010010

01010001001000011

Choose a random bit along the length, say at position 9, and swap all the bits after that point

so the above become [7] [8]:

10001001101000011

01010001010010010

f. Mutation

This is the chance that a bit within a chromosome will be flipped (0 becomes 1, 1 becomes 0). Whether a mutation is done or otherwise is determined by a constant p_m. This constant expresses the opportunity for mutation happening . The value of the constant p_m usually is set to be very low, for example 0.01. Whenever chromosomes are chosen from the population the algorithm first checks to see if crossover should be applied and then the algorithm iterates down the length of each chromosome mutating the bits if applicable. Iteration is done for each chromosome by taking a rondom score of 0 to 1.

If random score yielded is <= p_m, then the gene/bit is inversed, otherwise nothing is done. [7] [8] [9] [14].

Taking example there is a chromosome:

01101101010110101101

Is done by iteration at every bit and done by intake of random score 0 to 1:

- Bit 1 : 0.180979788303375
- Bit 2 : 0.64313793182373
- Bit 3 : 0.517344653606415
- Bit 4 : 0.0501810312271118
- Bit 5 : 0.172802269458771
- Bit 6 : 0.0090474414825439
- Bit 7 : 0.225485235254974
- Bit 8 : 0.128151774406433
- Bit 9 : 0.581712305545807
- Bit 10 : 0.173850536346436
- Bit 11 : 0.2264763712883
- Bit 12 : 0.518977284431458
- Bit 13 : 0.477839291095734
- Bit 14 : 0.529659032821655
- Bit 15 : 0.27648252248764
- Bit 16 : 0.266663908958435
- Bit 17 : 0.791664183139801
- Bit 18 : 0.167530059814453
- Bit 19 : 0.874832332134247
- Bit 20 : 0.0878018701157

Hence chromosome result of mutation shall be as follows:

0110111101011010110

g. Next Generation evaluation.

In this phase, all population is evaluated whether they have reached the expected solution. If not yet, hence returning to step (c) and done repeatedly until got the expected solution.

In the previous example, this algorithm will stop if one of the new chromosome in population get score 100 at the equation. For example, in a new generation if one of the chromosome is 10001010110001000111 (a=8, b=10, c=12, d=4, e=7), result of equation: 8 + 10 * 12 – 4 * 7 = 100. Hence this chromosome is expected solution, and genetic algorithm stop at this phase [7] [8] [9] [14].

3. GA's application in IRS

In this research design, a keyword represents a gene (a bit pattern), a document's list of keywords represents individuals (a bit string), and a collection of documents initially judged relevant by a user represents the initial population. Based on a Jaccard's score matching function (fitness measure), the initial population evolved through generations and eventually converged to an optimal (improved) population - a set of keywords which best described the documents. A similarity approach of document was adopted by Jaccard's score to compute the ``fitness" of subject descriptions for information retrieval [14].

3.1 Algorithm

Let's say there are *N* chromosomes in the initial population. Then, the following steps are repeated until a solution is found

1. Test each chromosome to see how good it is at solving the problem at hand and assign a *fitness score* accordingly. The fitness score is a measure of how good that chromosome is at solving the problem.
2. Select two members from the current population. The chance of being selected is proportional to the chromosomes fitness. Roulette wheel selection is a commonly used method.
3. Dependent on the crossover rate, crossover the bits from each chosen chromosome at a randomly chosen point.
4. Step through the chosen chromosomes bits and flip dependent on the *mutation rate*.
5. Repeat step 2, 3, 4 until a new population of *N* members has been created.

3.2 Implementation

As mentioned before, we have implemented a prototype of Journal Browser to retrieve a similar document from database by using Jaccard formulation as fitness'score.

Jaccard's score is formulated below:

$$= \frac{\#(X \cap Y)}{\#(X \cup Y)}$$

Where #(S) showing number of element in S.

For example:

S = {a, b, c, d, e, f, g, h, i, j}

if X = {a, b, e, g, h, i, j}

and Y = {b, c, d, f, g, j}

$$X \cup Y = \{a, b, c, d, e, f, g, h, i, j\}$$

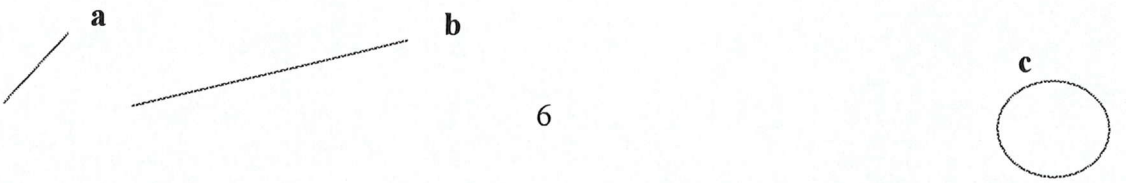
$$X \cap Y = \{b, g, j\}$$

$$\therefore \frac{\#(X \cap Y)}{\#(X \cup Y)} = \frac{3}{10} = 0.3$$

Jaccard's score represents common measurement at genetic algorithm. From this formula it can be seen that if #(X) is equal to #(Y) then Jaccard's score is 1. That mean if X element it is equal to Y fitness'score is 1, meaning X document precisely same with Y document, although this matter is very difficult founded in database.

If a keyword is present in a document, the bit is set 1, otherwise 0. Each document could then be represented in terms of a sequence of 0s and 1s. We have computed the fitness of each document based on its relevance to the documents in the user-selected set. Document with a higher Jaccard's score reflect a higher probability of similarity. Application of this technique will facilitate searching and retrieval of required document from one or more databases on the representation of similarity level [13] [14].

Form Journal Browser



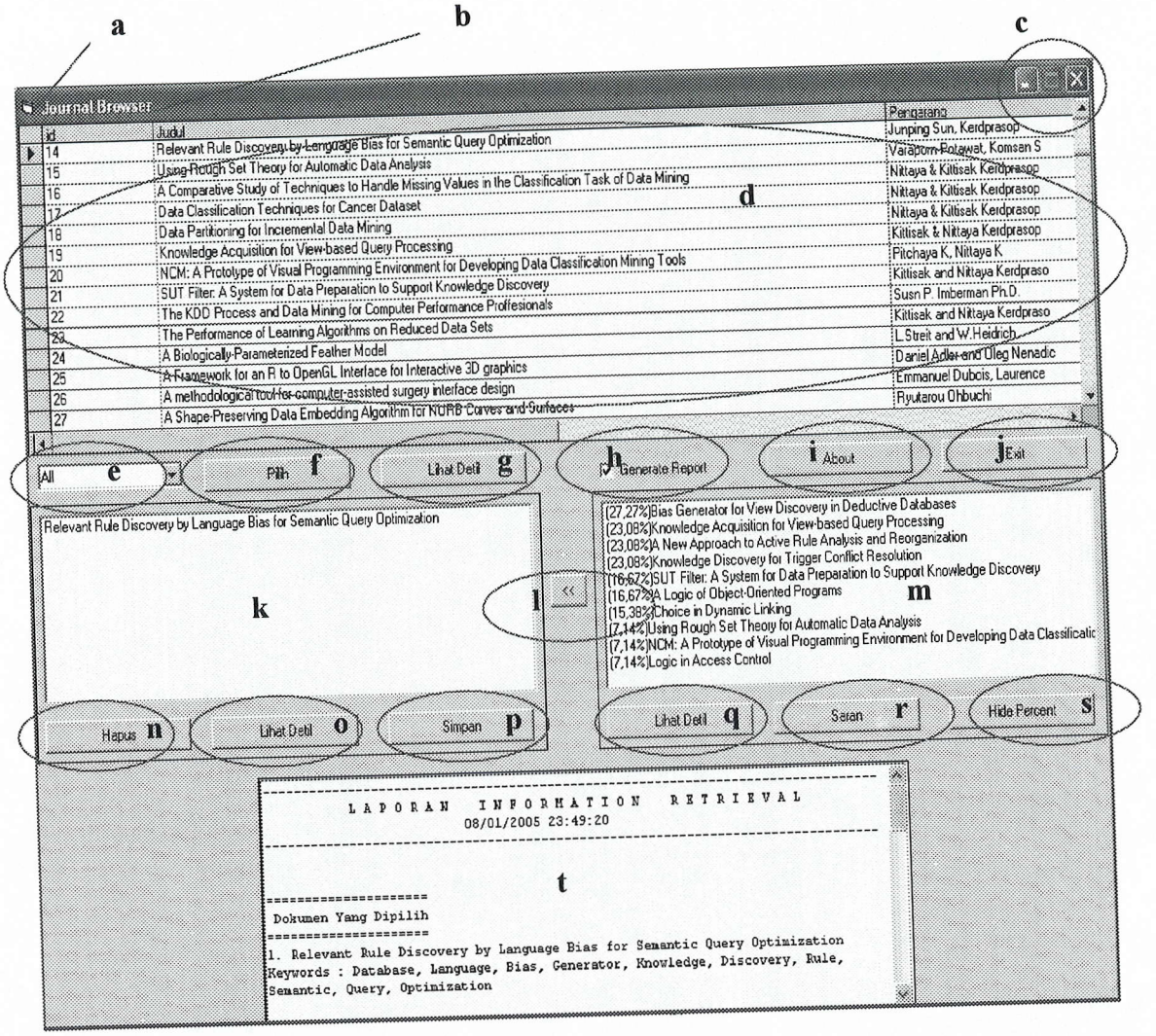


Fig. 2 Interface Form Journal Browser.

Jaccard's score is formulated below:

$$= \frac{\#(X \cap Y)}{\#(X \cup Y)}$$

Where #(S) showing number of element in S.

For example:

S = {a, b, c, d, e, f, g, h, i, j}

if X = {a, b, e, g, h, i, j}

and Y = {b, c, d, f, g, j}

$$X \cup Y = \{a, b, c, d, e, f, g, h, i, j\}$$

$$X \cap Y = \{b, g, j\}$$

$$\therefore \frac{\#(X \cap Y)}{\#(X \cup Y)} = \frac{3}{10} = 0.3$$

Jaccard's score represents common measurement at genetic algorithm. From this formula it can be seen that if #(X) is equal to #(Y) then Jaccard's score is 1. That mean if X element it is equal to Y

fitness'score is 1, meaning X document precisely same with Y document, although this matter is very difficult founded in database.

If a keyword is present in a document, the bit is set 1, otherwise 0. Each document could then be represented in terms of a sequence of 0s and 1s. We have computed the fitness of each document based on its relevance to the documents in the user-selected set. Document with a higher Jaccard's score reflect a higher probability of similarity. Application of this technique will facilitate searching and retrieval of required document from one or more databases on the representation of similarity level [13] [14].

4. The Result

In the testing a query we choose randomly, and then we used classification of query. Experiment indicates that percentage level of document as result of retrieval is consistent at certain range, although rangking value change. We mean that documents residing at top level, remain to have top level at its size measure.

There is tendency a query which envolves of mutation and crossover process will result an better document retrieval (has higher percentage of documents retrieval), while those have less mutation have lower percentage of document retrieval.

The parent solution (chromosome) with the higher level of fitness has a bigger probability to reproduce, while those with lower level of fitness have less probability to reproduce. Documents with a higher Jaccard's score reflect a higher probability of similarity.

5. Conclusion

Research in Information retrieval has been advancing very rapidly over the past few decades. Researchers have experimented with the techniques ranging from probabilistic models and the vector space model to the knowledge-based approach and the recent machine learning techniques. At each stage, significant insights regarding how to design more useful and ``intelligent'' information retrieval systems have been gained. Currently many IRS research are based on machine learning techniques. Symbolic machine learning and genetic algorithms further are two popular candidates for adaptive learning in other applications. This paper has discussed a technique of probability in document similarity comparison in IRS. We hope the system can be developed to improve the success in precise in document retrieval.

Reference

1. Chen, Hsinchun, Machine Learning for Information Retrieval: Neural Networks, Symbolic Learning, and Genetic Algorithms, Artificial Intelligence Lab, Eller College of Management, The University of Arizona, 1995.
   http://ai.bpa.arizona.edu/papers/mlir93/mlir93.html,
2. R.K.Belew.Adaptive information retrieval. In *Proceedings of the Twelfth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 11-20, Cambridge, MA, June 25-28, 1989.
3. R. E. Stepp and R. S. Michalski. Conceptual clustering: Inventing goal-oriented classifications of structured objects. In *Machine Learning, An Artificial Intelligence Approach, Vol. II*, pages 472-498, Pages 463-482, Carbonell, J. G., Michalski, R. S., and Mitchell, T. M., Editors, Morgan Kaufmann, Los Altos, CA, 1987.
4. S. M. Weiss and C. A. Kulikowski. *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Networks, Machine Learning, and Expert Systems*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1991.
5. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kauffmann, Los Altos, CA, 1993.
6. D. E. Rumelhart, B. Widrow, and M. A. Lehr. The basic ideas in neural networks. *Communications of the ACM*, 37(3):87-92, March 1994.
7. Hsiung, Sam, An Introduction to Genetic Algorithms, generation5, 2000,

http://www.generation5.org/content/2000/ga.asp,
8. Anonym, Genetic Algorithm Tutorial, ai-junkie, 2000. http://www.ai-junkie.com/ga/intro/gat1.html,
9. D. E. Goldberg. Genetic and evolutionary algorithms come of age. *Communications of the ACM*, 37(3):113-119, March 1994.
10. H. Chen and K. J. Lynch. Automatic construction of networks of concepts characterizing document databases.*IEEE Transactions on Systems, Man and Cybernetics*, 22(5):885-902, September/October 1992.
11. H. Chen, K. J. Lynch, K. Basu, and D. T. Ng. Generating, integrating, and activating thesauri for concept-based document retrieval. *IEEE EXPERT, Special Series on Artificial Intelligence in Text-based Information Systems*, 8(2):25-34, April 1993.
12. H. Chen and J. Kim. GANNET: a machine learning approach to document retrieval. *Journal of Management Information Systems*, 11(3):7-41, Winter 1994-95.
13. M. Gordon. Probabilistic and genetic algorithms for document retrieval. *Communications of the ACM*, 31(10):1208-1218, October 1988.
14. M. D. Gordon. User-based document clustering by redescribing subject descriptions with a genetic algorithm. *Journal of the American Society for Information Science*, 42(5):311-322, June 1991.